А. А. Молдовян, Н. А. Молдовян, Н. Д. Гуц, Б. В. Изотов

Криптография
Скоростные шифры

НАУЧНОЕ ИЗДАНИЕ

А. А. Молдовян
Н. А. Молдовян
Н. Д. Гуц
Б. В. Изотов

Криптография
Скоростные шифры

Санкт-Петербург
«БХВ-Петербург»
2002

УДК 681.3
ББК 32.81
М75

Рецензенты:
доктор технических наук, профессор В. И. Воробьев
доктор технических наук, профессор А. М. Чуднов

Молдовян А. А. и др.

М75 Криптография: скоростные шифры. — СПб.: БХВ-Петербург, 2002. — 496 с.: ил.

ISBN 5-94157-214-X

В книге рассматривается широкий круг вопросов, связанных с использованием криптографических методов защиты информации в компьютерных системах. Впервые излагается разработанная авторами концепция управляемых преобразований, являющаяся новым направлением прикладной криптографии. Представлены варианты построения управляемых операций и анализ их основных криптографических свойств. Дается описание ряда новых криптографических примитивов и скоростных криптосистем с оценкой их стойкости к дифференциальному, линейному и другим методам криптоанализа. Показана возможность построения операционных блоков, реализующих уникальные модификации операций для каждого значения управляющего кода. Отражены вопросы построения управляемых перестановок и управляемых подстановочных операций, в частности, управляемых сумматоров специального типа, а также представлены методы конструирования скоростных итеративных шифров на их основе.

Для специалистов в области безопасности информации, криптографии, прикладной математики, информатики и электроники, а также для преподавателей, студентов и аспирантов инженерно-технических вузов

УДК 681.3
ББК 32.81

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Евгений Рыбаков*
Зав. редакцией	*Анна Кузьмина*
Редактор	*Григорий Добин*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 19.09.02.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 39,99.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 198005, Санкт-Петербург, Измайловский пр., 29.

Гигиеническое заключение на продукцию, товар № 77.99.02.953.Д.001537.03.02
от 13.03.2002 г. выдано Департаментом ГСЭН Минздрава России.

Отпечатано с готовых диапозитивов
в Академической типографии "Наука" РАН
199034, Санкт-Петербург, 9 линия, 12.

ISBN 5-94157-214-X

© Молдовян А. А., Молдовян Н. А., Гуц Н. Д., Изотов Б. В., 2002
© Оформление, издательство "БХВ-Петербург", 2002

Содержание

Введение **11**

Глава 1. Криптография в информационном обществе **17**

1.1. Проблемы защиты информации в компьютерных системах 17
1.2. Проблематика криптографии 23
1.2.1. Традиционные вопросы криптографии 23
1.2.2. Современные приложения 29
1.2.3. Защита информации на технологическом уровне 37
1.3. Основы одноключевой криптографии 42
1.3.1. Условная и безусловная секретность 42
1.3.2. Общие вопросы разработки шифров 46
1.3.3. Композиционные и итеративные блочные шифры 48
1.3.4. Управляемые операции — новый криптографический примитив 53
Общая характеристика управляемых операций 53
1.4. Двухключевые криптосистемы 58
1.4.1. Система открытого распределения ключей 58
1.4.2. Понятие криптографического протокола 61
1.4.3. Цифровая электронная подпись 64
1.4.4. Криптосистема RSA 65
1.4.5. Цифровая подпись Эль-Гамаля 68
1.4.6. Слепая подпись Чаума 70
1.4.7. Виды нападений на цифровую подпись 71
1.5. Вероятностные шифры 73
1.5.1. Гомофонические шифры 73
1.5.2. Шифры с простым вероятностным механизмом 74
1.5.3. Вероятностное объединение битов данных и случайных битов 76
1.5.4. Вероятностные механизмы в двухключевых шифрах 79
1.5.5. Открытый шифр Эль-Гамаля 80

1.6. Применение шифрования на практике ... 81
1.6.1. Алгоритмы шифрования в средствах защиты ЭВМ ... 81
1.6.2. Особенности приложений ... 86
Длина ключа и стойкость ... 86
Шифрование и архивирование ... 88
Шифрование и кодирование ... 89
1.6.3. Стандартизация алгоритмов шифрования ... 90
1.6.4. Вопросы встраивания потайных лазеек ... 93
1.6.5. Криптография и стеганография ... 94

Глава 2. Гибкие программные шифры ... 97

2.1. Секретность алгоритма и стойкость ... 97
2.2. Принципы построения программных шифров ... 100
2.3. Инициализация программных шифров ... 103
2.4. Недетерминированные программные шифры ... 106
2.4.1. Принципы построения гибких шифров ... 106
2.4.2. Криптосхема с перестановкой фиксированных процедур ... 109
2.4.3. Многопроходные криптосхемы с гибким алгоритмом ... 109
2.4.4. Криптосистема с настройкой операций преобразования ... 110
2.4.5. Псевдовероятностный недетерминированный шифр ... 111
2.4.6. Гибкие шифры с доказуемой неэквивалентностью модификаций криптоалгоритма ... 114
2.4.7. Комбинаторно-вероятностная модель ... 116
2.5. Скоростные программные шифры. Обозначения и термины ... 118
2.6. Шифры на основе выборки подключей в зависимости от данных ... 119
2.7. Алгоритмы шифрования в современных системах компьютерной безопасности ... 120
2.7.1. Скоростное шифрование дисковых данных ... 121
Критерии построения ... 121
Общая схема преобразований ... 123
2.7.2. Предвычисления ... 124
Процедура *Form_Q* ... 126
2.7.3. Алгоритмы дискового шифрования ... 127
Процедура *Encrypt_Z* (зашифрование 512-байтового блока данных) ... 128
Процедура *Decrypt_Z* ... 129
2.7.4. Оценка стойкости ... 132
2.7.5. Алгоритм файлового шифрования ... 137
Формирование локального ключа ... 138
Шифрование текущего байта ... 139
2.7.6. Преобразование данных в загрузочном секторе ... 140
Процедура *D* (мини-алгоритм расшифрования) ... 140
2.8. Программная хэш-функция с гибким входом ... 141
2.8.1. Схемы хэш-функций ... 143
2.8.2. Построение функции $f(x)$ ... 144
Процедура *Table_Q* ... 144
Раундовая хэш-функция ... 145

2.9. Программный шифр с гибким входом ... 147
2.9.1. Предвычисления ... 148
Процедура *Table_Z* ... 148
Процедура *FormKey* ... 148
2.9.2. Алгоритмы преобразования ... 149
Процедура *Initialize* ... 149
Процедура *Change_NVYU* ... 149
Процедура зашифрования в четырех сокращенных раундах ... 149
Процедура расшифрования в сокращенных раундах ... 150
Процедура *Encrypt512*: первый (полный) раунд зашифрования ... 150
Процедура шестого (полного) раунда (зашифрование) ... 151
Процедура первого (полного) раунда (расшифрование) ... 151
Процедура шестого (полного) раунда (расшифрование) ... 152

Глава 3. Управляемые перестановочные операции ... 157

3.1. Понятие управляемой перестановочной операции ... 159
3.2. Разработка операционных блоков управляемых перестановок с заданными параметрами скорости и схемотехнической сложности ... 170
3.2.1. Первый *(формальный)* вариант синтеза блока управляемых перестановок $P_{n/m}$ ... 171
3.2.2. Второй *(экономичный)* вариант реализации блока управляемых перестановок $P_{n/m}$ ... 174
3.2.3. Построение обратного преобразования $P_{n/m}^{-1}$ ... 176
3.3. Синтез блоков управляемых перестановок с заданными алгебраическими и вероятностно-статистическими свойствами ... 177
3.3.1. Блоки равновероятного смещения и блоки порядка *h* ... 177
Сети Клоса ... 182
Рекурсивная модель блоков равновероятного смещения и блоков первого порядка с послойной структурой ... 184
Представление блока R_k двухуровневой сетью Клоса ... 187
Эквивалентность моделей R_k и R_k^{-1} ... 190
3.3.2. Схемы реализации блоков максимального порядка ... 192
Универсальная схема ... 192
Схема Бенеша ... 194
Модернизированный вариант схемы Бенеша ... 195
Схема Ваксмана ... 196
3.3.3. Рекурсивная схема реализации блоков порядка *h* ... 197
3.3.4. Реализация полноцикловых перестановок ... 201
3.3.5. Синтез блоков с заданной цикловой структурой ... 202
Инволюции ... 204
3.4. Линейный криптоанализ управляемых перестановок ... 206
3.4.1. Основные понятия линейного криптоанализа ... 206
Композиция и декомпозиция входных и выходных векторов ... 209
Согласование линейных характеристик ... 210

3.4.2. Линейные характеристики блоков управляемых перестановок 211
3.4.3. Способы и практические рекомендации вычисления смещений для линейных характеристик блоков управляемых перестановок 218
Разбиение множества объединенных масок на классы эквивалентности 222
3.5. Формирование управляющих векторов 224
3.5.1. Процедуры формирования управляющих векторов для блоков $P_{n/m}$ 224
3.6. Основные результаты 228

Глава 4. Управляемые подстановочные операции 231

4.1. Характеристика управляемых подстановочных преобразований 231
4.2. Математические свойства криптографических преобразований 237
4.2.1. Представление криптографических преобразований в виде отображений и подстановок 238
4.2.2. Булевы функции в криптографических преобразованиях 239
4.2.3. Роль дискретного преобразования Фурье в описании свойств криптографических преобразований 241
4.2.4. Характеристики нелинейности булевых функций 243
4.2.5. Автокорреляция булевых функций 246
4.2.6. Критерий строгого лавинного эффекта и критерий распространения изменений 247
4.2.7. Корреляционная иммунность и эластичность булевых функций 249
4.2.8. Роль бент-функций в криптографических преобразованиях 251
4.2.9. Линейные структуры булевых функций 253
4.2.10. Обобщенные свойства нелинейности криптографических преобразований 255
4.2.11. Общие требования, предъявляемые к криптографическим примитивам 256
4.3. Модели управляемых подстановочных операций 257
4.3.1. Определение управляемой подстановочной операции 257
4.3.2. Общая модель управляемой подстановочной операции 259
4.3.3. Последовательная модель управляемой подстановочной операции 263
4.3.4. Рекурсивная модель управляемой подстановочной операции 267
4.3.5. Синтез элементарных управляемых сумматоров 275
4.3.6. Небиективная модель управляемой подстановочной операции 281
4.3.7. Синтез комбинированных управляемых сумматоров и управляемых схемных преобразователей 281
4.3.8. Свойства булевых функций, образующих управляемые подстановочные операции 284
4.3.9. Оптимизация управляемых подстановочных операций на базе специальных булевых функций 285
4.3.10. Общие замечания относительно управляемых подстановочных операций 287
4.3.11. Выводы 289
4.4. Влияние управляемых подстановочных операций на стойкость блочных шифров к криптоанализу 291
4.4.1. Общие сведения об аналитических и технических атаках на блочные шифры 291
4.4.2. Теоретические основы дифференциального криптоанализа 294

4.4.3. Влияние управляемых подстановочных операций на стойкость блочных шифров к дифференциальному криптоанализу ... 300
4.4.4. Теоретические основы линейного криптоанализа ... 304
4.4.5. Влияние управляемых подстановочных операций на стойкость блочных шифров к линейному криптоанализу ... 308
4.4.6. Влияние управляемых операций на стойкость блочных шифров к техническим атакам ... 311
4.4.7. Обзор результатов криптоанализа блочного шифра ГОСТ 28147-89 по открытым публикациям ... 314
4.5. Практическое использование управляемых подстановочных операций ... 316
4.5.1. Общие требования к блочным шифрам и управляемые подстановочные операции ... 316
4.5.2. Оценки сложности реализации управляемых подстановочных операций ... 318
4.5.3. Итеративные блочные шифры на основе управляемых подстановочных операций ... 321
4.5.4. Варианты синтеза блочных шифров на основе управляемых операций ... 323
4.5.5. Синтез комбинированных раундовых функций на основе управляемых подстановочных операций ... 327
4.5.6. Практическая реализация управляемых подстановочных операций в блочных шифрах ... 328
4.6. Заключение ... 332

Глава 5. Проектирование скоростных шифров на основе управляемых операций ... 333

5.1. Базовые криптосхемы и ключевая система ... 333
5.1.1. Итеративный (композиционный) блочный шифр ... 335
5.1.2. Процедура генерации расширенного ключа ... 336
5.1.3. Практика использования расписания ключей ... 338
Начальное и заключительное преобразования ... 339
5.1.4. Два основных типа схем шифрования блочных шифров ... 342
5.1.5. Усовершенствованная схема блочного шифрования одного раунда (патент РФ № 2140714) ... 343
5.1.6. Общие элементы алгоритмов блочного шифрования на основе управляемых операций ... 347
Структура базовой модели криптосхемы итеративного *r*-раундового алгоритма блочного шифрования ... 347
Начальное и заключительное преобразования ... 349
5.1.7. Универсальные схемы блочного шифрования на основе взаимно обратных управляемых операций ... 350
Модель 1 ... 352
Модель 2 ... 356
5.2. Блочный шифр SPECTR-H64 ... 360
5.2.1. Общая схема блочного шифрования ... 360
5.2.2. Алгоритм шифрования ... 360
Начальное преобразование *IT* ... 361

Процедура *Crypt* ... 361
Заключительное преобразование *FT* ... 367
5.2.3. Расписание использования раундовых ключей ... 367
5.3. Шифр (алгоритм) SPECTR-128 ... 369
5.3.1. Общая схема блочного шифрования ... 369
5.3.2. Алгоритм шифрования ... 369
Начальное преобразование *IT* ... 370
Процедура *Crypt* ... 370
Заключительное преобразование *FT* ... 376
5.3.3. Расписание использования раундовых ключей ... 376
5.4. Шифр (алгоритм) CIKS-128 ... 378
5.4.1. Общая схема блочного шифрования ... 378
5.4.2. Алгоритм шифрования ... 379
Начальное *(IT)* и заключительное *(FT)* преобразования ... 379
Процедура *Crypt* ... 379
5.4.3. Расписание использования раундовых ключей ... 385
Универсальность схемы ... 387
5.5. Перспективные программные шифры на основе управляемых перестановок ... 388
5.5.1. Описание гипотетической команды *DDP32* ... 389
5.5.2. Программный шифр SPECTR-SZ ... 391
Обозначения и входные данные ... 392
Структура шифра ... 393
Процедура генерации расширенного ключа ... 394
Процедура *Table_H* ... 394
Процедура *FormKey* ... 395
Процедура зашифрования *Encrypt* ... 395
Процедура расшифрования *Decrypt* ... 398
Скоростные параметры и криптографическая стойкость шифра ... 398
5.5.3. Блочные шифры COBRA-F64a и COBRA-F64b ... 400
Общая схема шифрования ... 400
Расписание использования раундовых ключей ... 402
Алгоритм шифрования ... 403
Скоростные параметры и криптографическая стойкость шифров ... 404
5.5.4. Алгоритмы DDP-S64 и DDP-S128 ... 406
Общая схема шифрования ... 406
64-битовый шифр DDP-S64 ... 407
128-битовый шифр DDP-S128 ... 409
5.6. Статистические свойства алгоритмов ... 412
5.6.1. Критерии оценки свойств "лавинного эффекта" ... 412
5.6.2. Оценка влияния битов исходного текста на преобразованный текст ... 414
5.6.3. Оценка влияния битов ключа на преобразованный текст ... 415
5.6.4. Использование критерия χ^2 для уточнения интегральных оценок ... 416
Основные результаты ... 417

Глава 6. Элементы криптоанализа шифров на базе управляемых операций ... 419

6.1. Об оценке гибких шифров ... 419
6.2. Дифференциальные характеристики блоков управляемых перестановок ... 423
6.3. Анализ криптосистемы SPECTR-H64 ... 429
6.4. Дифференциальный криптоанализ шифра SPECTR-128 ... 437
6.4.1. Стойкость криптосистемы SPECTR-128 с модифицированным блоком расширения ... 444
6.5. Основные дифференциальные характеристики шифров DDP-S64 и DDP-S128 ... 447
6.5.1. Анализ шифра DDP-S64 ... 448
6.5.2. Анализ шифра DDP-S128 ... 451
6.6. Оценка стойкости шифров COBRA-F64a и COBRA-F64b ... 453
6.7. Атака на основе аппаратных ошибок ... 458
6.7.1. Криптоанализ шифра RC5 ... 461
6.7.2. Криптоанализ гибкого шифра ... 462
6.7.3. Стойкость алгоритма ГОСТ 28147-89 ... 466
6.7.4. Стойкость шифров на основе псевдослучайной выборки подключей ... 466

Заключение ... 469

Глоссарий ... 471

Список литературы ... 477

Предметный указатель ... 489

Введение

В настоящее время криптографические методы нашли широкое применение не только для защиты информации от несанкционированного доступа, но и в качестве основы многих новых электронных информационных технологий — электронного документооборота, электронных денег, тайного электронного голосования и др. Современная криптография решает следующие три основные проблемы:

- обеспечение конфиденциальности (секретности);
- обеспечение аутентификации информации и источника сообщений;
- обеспечение анонимности (например, сокрытие перемещения электронных денег от одного субъекта к другому).

Первая проблема известна тысячи лет, последние же две являются относительно новыми, и с их решением связан ряд перспективных направлений теоретической и практической криптографии. Тем не менее, традиционная криптографическая задача обеспечения секретности информации не утратила своей остроты и в настоящее время. Это связано, главным образом, с тем, что в эпоху массового применения компьютерных технологий задача защиты электронной информации приобрела характер широкомасштабной проблемы. При этом к алгоритмам шифрования предъявляются жесткие технологические требования, которые продиктованы их использованием в различных электронных устройствах (телекоммуникационных системах, ЭВМ, компьютерных сетях, интеллектуальных электронных карточках и др.). Характерным для технологических применений криптографических средств является возрастание требований к шифрам одновременно по стойкости, скорости и по простоте реализации. Ужесточение требований по стойкости обусловлено тем, что разностороннее использование криптографии связано с более широкими возможностями для атакующего следовать особенностям конкретных условий, в которых функционирует криптосистема (например, имеются возможности: первая — осуществить внешнее воздействие на устройство шифрования с целью вызвать случайные аппаратные сбои, вторая — выпол-

нить замер потребляемой мощности, третья — определить время вычислений и т. п.). Возросшие требования по скорости связаны с необходимостью сохранения высокой производительности автоматизированных систем после встраивания в них механизмов защиты. Простота аппаратной реализации необходима для снижения стоимости средств шифрования, что будет способствовать их массовому применению и более широким возможностям встраивания в портативную аппаратуру. В силу специфики представления информации в цифровых устройствах наибольший интерес представляют блочные шифры, которые могут, например, обеспечить произвольный доступ к данным на зашифрованных магнитных носителях.

Таким образом, разработка скоростных блочных шифров является важной задачей прикладной криптографии. В этом направлении сделано большое число предложений со стороны российских и зарубежных криптографов. Характерным для большинства новых скоростных шифров является использование предвычислений, осуществляющих расширение секретного ключа. При создании программных шифров применение сложных алгоритмов предвычислений с целью упрощения шифрующих преобразований во многих приложениях является оправданным. При разработке шифров двойной (программной и аппаратной) ориентации, например нового американского стандарта AES, криптографы также в первую очередь учитывают возможности современных массовых процессоров, полагая, что существующая микроэлектронная технология легко может обеспечить аппаратную реализацию весьма сложных алгоритмов преобразования. Однако при массовом использовании устройств шифрования в компьютерных и телекоммуникационных системах, интеллектуальных электронных карточках экономия аппаратных ресурсов является крайне важной. Наиболее экономичные решения позволяют изготовить более дешевые и более надежные устройства, а также снизить потребляемую мощность.

В ряде сетевых приложений требуется обеспечить высокую скорость шифрования при частой смене ключей, что ограничивает применение шифров с предвычислениями, поскольку последние вносят существенные ограничения по быстродействию. В связи с этим весьма важным становится уменьшение сложности предвычислений при сохранении высокой криптостойкости преобразования. В этом плане удачным решением представляется полный отказ от предварительного преобразования секретного ключа путем замены этой процедуры операциями преобразования подключей в зависимости от преобразуемых данных, которые выполняются одновременно с операциями преобразования данных. Таким образом, актуальной является разработка скоростных шифров нового поколения, допускающих экономичную аппаратную реализацию и сохраняющих высокую скорость шифрования при частой смене ключей. При этом криптосистемы должны обладать высокой стойкостью к классическим вариантам атак на основе известных и специально подобранных текстов, а также к новым атакам, например, на основе генерации случайных аппаратных ошибок.

Одним из перспективных направлений построения скоростных шифров является использование гибких операций и/или процедур преобразования. В случае программной реализации, например, легко реализуются криптосистемы, в которых на этапе предвычислений формируется алгоритм шифрования в зависимости от секретного ключа, что может быть сделано путем настройки операций преобразования в зарезервированных местах алгоритма или путем комбинирования процедур преобразования из некоторого специфицированного подмножества шифрующих функций (библиотеки процедур преобразования). Такие шифры можно назвать *гибкими*. В действительности, гибкие блочные алгоритмы с размером входа, равным b битов, формально описываются так же, как и любой другой шифр, — одноалфавитной подстановкой на множестве возможных входных текстов длиной b. Однако, с учетом ориентации на стандартные операции микропроцессора, оправданным и удобным является описание таких шифров в терминах настройки алгоритма. Данный подход к построению гибких аппаратных шифров может быть применен при использовании микропроцессорных устройств шифрования. Недостаток такого подхода выражается в том, что требуется выполнение достаточно сложных процедур предвычислений, существенно замедляющих скорость шифрования в случае частой смены ключей.

Одним из интересных способов синтеза блочных шифров является построение аппаратно ориентированных шифров на основе управляемых операций. Под гибкой (управляемой) операцией преобразования понимается операция, управляемая некоторым двоичным вектором. При фиксированном значении управляющего вектора реализуются преобразования, относящиеся к одному из вариантов (модификаций) гибкой операции. Таким образом, управляемую операцию можно охарактеризовать как множество различных модификаций, каждая из которых соответствует конкретному значению управляющего вектора. В общем случае, различным значениям управляющего вектора может соответствовать одна и та же модификация. Однако при проектировании управляемых операций разумно ориентироваться на следующие конструктивные критерии:

- уникальность всех модификаций;
- большое число модификаций;
- существенные различия модификаций.

Управляемые операции преобразования, по сути, являются новым криптографическим примитивом, задающим для данного секретного ключа фиксированные процедуры зашифрования и расшифрования, в которых управляющий вектор формируется по секретному ключу и/или по преобразуемому блоку данных. Построенные на основе управляемых операций криптоалгоритмы являются гибкими в том смысле, что их описание на основе простых, традиционно используемых операций, приводит к необходимости учитывать зависимость выбора конкретных модификаций таких операций от переменных параметров криптосистемы (ключа, шифруемого текста).

Если управляющий вектор формируется в зависимости только от преобразуемого блока данных, то гибкую операцию будем называть *операцией*, *зависящей от преобразуемых данных* (ОЗПД). Если управляющий вектор формируется по ключу и по данным, то в подобной гибкой операции в зависимости от ключа задается тот или иной набор модификаций, из которого в зависимости от значения текущего управляющего подблока данных осуществляется выбор конкретной текущей модификации. Таким образом, в этом случае имеем некоторую ОЗПД, зависящую от ключа. Такие примитивы расширяют возможности разработчиков при проектировании гибких шифров.

ОЗПД обычно задают нелинейные преобразования, даже если все соответствующие ей модификации являются линейными. В общем плане эффективность ОЗПД связана с увеличением размера преобразуемого подблока данных, над которым они задают некоторую подстановку. Если модификации данной ОЗПД задают преобразование n-битовых двоичных векторов и число этих модификаций M, то сама ОЗПД определяет преобразование над N-битовыми векторами, где $N = n + \log_2 M$. Из последнего соотношения видно, что для повышения эффективности ОЗПД необходимо существенно увеличить число модификаций. Другим способом является разработка ОЗПД, включающих модификации специального типа. Использование ОЗПД на настоящий момент прошло достаточную практическую апробацию на примере блочных шифров RC5, RC6 и MARS, которые детально исследовались в последние годы. Было показано, что операции циклического сдвига, зависящие от преобразуемых данных и включающие всего лишь 32 модификации, являются эффективным средством противодействия самым мощным современным криптоатакам — дифференциальному и линейному криптоанализу.

Более эффективными способами шифрования на базе ОЗПД являются запатентованные в России технические решения, базирующиеся на управляемых перестановках (патенты РФ № 2127024, 2140710, 2140713, 2140715) и управляемых двухместных операциях (патент № 2140716). Число модификаций, соответствующих некоторой операции управляемой перестановки, может иметь значение до $n!$ ($\approx 2^{n(\log 2n-1)}$). Для управляемых двухместных операций могут быть разработаны быстродействующие операционные блоки, задающие значение M от 2^n до 2^{kn}, где $k \geq 2$ — натуральное число.

В данной книге рассматривается применение управляемых операций для построения скоростных шифров, сочетающих в себе высокую стойкость, высокую скорость преобразования данных и низкую сложность схемотехнической реализации. Показана возможность построения операционных блоков, реализующих уникальные модификации операций для каждого значения управляющего кода. Формирование управляющего кода в зависимости от преобразуемых данных и от секретного ключа дает возможность эффективной реализации нового технического решения (патент № 2124814), связанного с построением скоростных шифров, в которых алгоритм шифрования формируется в зависимости от секретного ключа путем настройки ОЗПД. Замечательной особенностью таких шифров яв-

ляется тот факт, что в них не используются предвычисления. Благодаря этому упрощается схемная реализация и легко обеспечивается скорость шифрования 1 Гбит/с и более при произвольной частоте смены секретных ключей. Отражены вопросы построения управляемых перестановок и управляемых подстановочных операций, в частности, управляемых сумматоров специального типа, а также представлены методы конструирования скоростных итеративных шифров на их основе.

Шифры на базе управляемых операций обладают следующими свойствами, обеспечивающими высокую стойкость к линейному и дифференциальному криптоанализу:

1. Нелинейные преобразования выполняются над подблоками большого размера (32, 64, 128, 256 бит).
2. ОЗПД зависят от секретного ключа.
3. Раундовые ключи преобразуются в зависимости от преобразуемых данных.

При расширении элементарных команд микропроцессоров инструкциями, реализующими операцию управляемой перестановки, легко могут быть разработаны программные шифры, обладающие скоростью более 1 Гбит/с при использовании массовых микропроцессоров типа Pentium. Такое расширение команд практически не изменит стоимость процессоров ввиду малой схемотехнической сложности операционных блоков, реализующих управляемую перестановку. При этом подобная новая команда может выполнить операцию циклического сдвига над двоичными векторами произвольной длины (например, над 32-битовыми словами, одновременно над двумя 16-битовыми словами и одновременно над четырьмя байтами).

Особый интерес представляет использование гибких шифров без предвычислений для построения стойких скоростных хэш-функций. Такие хэш-функции характеризуются зависимостью процедур преобразования (трактуемых в стандартных операциях) от хэшируемых данных, что позволяет назвать их гибкими хэш-функциями. Следует, однако, отметить, что термин "гибкая операция", также как и термин "ОЗПД", является условным. Он только подчеркивает способ описания некоторой более сложной операции над двоичными векторами большей длины с использованием более простых стандартных операций. Такой способ является достаточно признанным. Он дает более простое описание новых примитивов, которые при формальном описании теряли бы свою наглядность и простоту восприятия.

При написании данной книги авторы имели намерение дать общую картину современной криптографии в таком изложении, которое было бы легко доступно широкому кругу читателей, а также рассмотреть специальные вопросы, связанные с разработкой скоростных блочных шифров, ориентированных на программную и/или аппаратную реализацию. К вопросам общего характера относится материал, посвященный значению криптографии в информационном обществе,

двухключевой криптографии, криптографическим протоколам, электронной цифровой подписи и ряду хорошо известных одноключевых шифров. Подробное рассмотрение проблемы и подходов к построению скоростных блочных шифров также представляет достаточно широкий интерес ввиду их большого значения в современных технологиях защиты информации и широкого применения. Если не вдаваться в детальные математические выкладки и доказательства, то и эта часть материала является доступной широким слоям читателей, поскольку излагаемые варианты построения скоростных шифров проиллюстрированы наглядными рисунками и схемами. Приведенная в книге библиография отражает уровень современных достижений теоретической и прикладной криптографии в области блочных шифров. Читатели, заинтересованные в глубоком освоении концепции управляемых преобразований как подхода к построению скоростных блочных шифров, найдут ее математическое обоснование.

ГЛАВА 1

Криптография в информационном обществе

1.1. Проблемы защиты информации в компьютерных системах

Криптография исторически зародилась из потребности передачи секретной информации. Длительное время она была связана только с разработкой специальных методов преобразования информации с целью ее представления в форме, недоступной для потенциального злоумышленника. С началом применения электронных способов передачи и обработки информации задачи криптографии начали расширяться. В настоящее время, когда компьютерные информационные технологии нашли массовое применение, проблематика криптографии включает многочисленные задачи, которые не связаны непосредственно с засекречиванием информации. Современные проблемы криптографии включают разработку систем электронной цифровой подписи и тайного электронного голосования, протоколов электронной жеребьевки и аутентификации удаленных пользователей, методов защиты от навязывания ложных сообщений и т. п.

Многие задачи практической информатики эффективно решаются с использованием криптографических методов. В криптографии рассматривается некоторый *злоумышленник* (оппонент, криптоаналитик противника, нарушитель, нелегальный пользователь), который осведомлен об используемых криптографических алгоритмах, протоколах, методах и пытается вскрыть их. Вскрытие криптосистемы может заключаться, например, в несанкционированном чтении информации, формировании чужой подписи, изменении результатов голосования, нарушении тайны голосования, модифицировании данных, которое не будет обнаружено законным получателем. Разнообразные действия оппонента в общем случае называются *криптографической атакой* (нападением). Специфика криптографии состоит в том, что она направлена на разработку методов, обеспечивающих стойкость к любым действиям злоумышленника, хотя на момент разработки криптосистемы нереально предусмотреть все возможные способы атаки, которые могут

быть изобретены в будущем на основе новых достижений теории и технологического прогресса. Центральным является вопрос, насколько надежно решается та или иная криптографическая проблема. Ответ на этот вопрос непосредственно связан с оценкой трудоемкости каждой конкретной атаки на криптосистему. Решение такой задачи, как правило, чрезвычайно сложно и составляет самостоятельный предмет исследований, называемый *криптоанализом*. *Криптография и криптоанализ* образуют единую область науки — *криптологию*, которая в настоящее время является новым разделом математики, имеющим важные приложения в современных информационных технологиях.

Широкое применение компьютерных технологий в системах обработки данных и управления привело к обострению проблемы защиты информации от несанкционированного доступа. Защита информации в компьютерных системах обладает рядом специфических особенностей, связанных с тем, что информация не является жестко связанной с носителем, может легко и быстро копироваться и передаваться по каналам связи. Известно большое число угроз информации, которые могут быть реализованы как со стороны внешних, так и внутренних нарушителей [9].

Радикальное решение проблем защиты информации, циркулирующей в высокопроизводительных автоматизированных системах, может быть получено на базе использования криптографических методов. При этом важным является применение скоростных алгоритмов шифрования, которые не приводят к снижению производительности компьютерных и телекоммуникационных систем. Криптографические преобразования данных являются гибким и эффективным средством обеспечения их конфиденциальности, целостности и подлинности. Использование методов криптографии в совокупности с необходимыми техническими и организационными мероприятиями может обеспечить защиту от широкого спектра потенциальных угроз.

Потребности современной практической информатики привели к возникновению нетрадиционных задач защиты электронных данных, одной из которых является аутентификация электронной информации в условиях, когда обменивающиеся этой информацией стороны не доверяют друг другу. Эта проблема связана с созданием систем электронной цифровой подписи. Теоретической базой для решения этой проблемы стало открытие в середине 1970-х годов американскими исследователями У. Диффи и М. Е. Хеллманом *двухключевой криптографии* [62], которое явилось блестящим достижением многовекового эволюционного развития криптографии. Революционные идеи двухключевой криптографии привели к резкому росту числа открытых исследований в этой области, показали новые пути совершенствования криптографии, ее далеко не исчерпанные возможности и уникальное значение в современных условиях бурного развития электронных информационных технологий.

Технической основой перехода в информационное общество являются современные микроэлектронные технологии, которые обеспечивают непрерывный рост

качества средств вычислительной техники и служат базой для сохранения основных тенденций ее развития:

- ❒ миниатюризации и снижения энергопотребления оборудования;
- ❒ увеличения объема оперативной памяти (ОП) и емкости встроенных и съемных накопителей информации;
- ❒ роста производительности и надежности систем;
- ❒ расширения сфер и масштабов применения ЭВМ.

Данные тенденции развития средств вычислительной техники привели к тому, что на современном этапе защита компьютерных систем от несанкционированного доступа характеризуется применением *программных криптографических методов защиты*.

Как показывает практика последних лет, расширяются масштабы использования и *аппаратных шифров*. Одной из актуальных проблем практической криптографии является построение алгоритмов, обеспечивающих скорость шифрования более 1500 Мбит/с при их реализации в виде недорогих микроэлектронных схем — *крипточипов*. В первую очередь это связано с широким применением шифрования в коммерческом телевидении. Другой массовой областью использования крипточипов является мобильная телефония. В последнее время специалистами по защите информации и пользователями осознана необходимость криптографической защиты и при передаче охранной информации с видеокамер внешнего наблюдения, а также других устройств сигнализации. Эта область применения криптографии требует разработки устройств шифрования низкой схемотехнической сложности.

Одной из важных социально-этических проблем, порожденных все более расширяющимся применением методов криптографической защиты информации, является противоречие между желанием пользователей защитить свою информацию и передачу сообщений и желанием специальных государственных служб иметь возможность доступа к информации организаций и отдельных лиц с целью пресечения незаконной деятельности. В развитых странах наблюдается широкий спектр мнений о подходах к вопросу о регламентации использования алгоритмов шифрования. Высказываются предложения от полного запрета широкого применения криптографических методов до полной свободы их использования. Некоторые предложения относятся к разрешению использования только ослабленных алгоритмов или к установлению порядка обязательной регистрации ключей шифрования.

Чрезвычайно трудно найти оптимальное решение этой проблемы. Как оценить соотношение потерь законопослушных граждан и организаций от незаконного использования их информации, с одной стороны, и убытков государства от невозможности получения доступа к зашифрованной информации отдельных групп, скрывающих свою незаконную деятельность, с другой стороны? Как исключить незаконное использование криптоалгоритмов лицами, которые нару-

шают и другие законы? Кроме того, всегда существуют способы скрытого хранения и передачи информации. Эти вопросы еще предстоит решать социологам, психологам, правоведам и политикам.

Что касается исследовательских работ в области криптографии, которые могут привести к удобным и практически абсолютно стойким алгоритмам, то их ограничение вряд ли представляется разумным. Видимо, надо предоставить законопослушным гражданам и организациям одинаковые возможности для защиты своей информации по сравнению с возможностями злоумышленников, которые так или иначе получают доступ к достижениям криптографии и поэтому могут оказаться в существенно более выгодных условиях.

В дополнение к этому, ограничение исследовательских работ в области криптографии быстрее всего приведет к отставанию в ее развитии, но никак не к предотвращению доступа криминальных структур к современным криптографическим методам, полученным, например, из других стран. В результате наиболее серьезно будут ущемляться интересы именно законопослушных организаций. Во многих странах мира начали понимать сущность данной проблемы, что привело в настоящее время к возрастанию числа развитых стран, в которых снимаются жесткие ограничения на применение шифрования.

Независимо от прогресса в области разработки криптографических методов защиты информации государство всегда имеет возможность законодательно потребовать, чтобы все пользователи шифров регистрировали свои ключи (или необходимую долю ключевых данных) в специально созданных учреждениях. В этом случае контроль информации обеспечивается независимо от уровня стойкости используемых алгоритмов. Эти и другие аспекты показывают, что торможение исследовательских работ в области криптографии не является объективно обоснованным. Еще в начале 1970-х годов в развитых западных странах потребности практики вызвали интерес к криптографии со стороны широких кругов исследователей и дали толчок к началу открытых исследований в этой области, которая прежде считалась элитарной и входила в сферу интересов сугубо секретных служб.

Известен ряд примеров, когда секретность в области криптографии приводила к существенным провалам при изготовлении засекречивающей аппаратуры и даже к отставанию от научно-технического прогресса. Широкая деятельность в столь животрепещущей сфере науки создали условия для повышения и самого качества исследований в криптологии, что позволило У. Диффи и М. Е. Хеллману открыть двухключевую криптографию, идеи которой привели к возникновению новых нетрадиционных разделов криптографии и превратили ее в одно из бурно развивающихся направлений современной математики. Открытие двухключевой криптографии является ярким примером взаимодействия теории и практики, а также примером влияния политики на достижения теории.

Сдерживание открытых исследований в области криптографии упрощает некоторые проблемы спецслужб, однако для государства, в целом, отрицательный эф-

фект от такого сдерживания крайне велик и связан с отставанием в области разработки современных систем защиты информации, распространенностью компьютерных преступлений и другими проблемами. Примером может служить глобальная компьютерная сеть Internet, являющаяся грандиозным достижением компьютерных технологий, внедрение которой, однако, связано с массой правовых нарушений и преступлений.

В результате опыта работы в Internet были выявлены слабости и недостатки традиционных административных и системных механизмов защиты информации. Криптография предоставляет принципиально новые возможности обеспечения безопасности информации в компьютерных сетях, и сейчас ее методы активно внедряются в глобальные сетевые технологии. Не отказ от прогресса в информатизации, а использование современных достижений криптографии — вот стратегически правильное решение, подтвержденное практикой. Возможность широкого применения криптографии в компьютерных сетях является большим достижением и признаком демократического общества.

Владение основами криптографии в информационном обществе объективно не может быть привилегией отдельных государственных служб, а является насущной необходимостью для самых широких слоев научно-технических работников, применяющих компьютерную обработку данных или разрабатывающих информационные системы, сотрудников служб безопасности и руководящего состава различных организаций и предприятий. Только такой подход может служить базой для повышения эффективности внедрения и эксплуатации высококачественных систем защиты электронной информации.

Одна отдельно взятая организация не может обеспечить достаточно полный контроль над информационными потоками в пределах всего государства и надлежащую защиту пространственно распределенного национального информационного ресурса. Однако определенные государственные организации могут создать условия для формирования рынка качественных средств защиты, подготовки достаточного количества специалистов и овладения массовыми пользователями основами защиты информации и криптографии.

В России и других странах СНГ в начале 1990-х годов отчетливо прослеживалась тенденция опережения расширения масштабов и областей применения информационных технологий над развитием систем защиты данных. Такая ситуация в определенной степени являлась и является типичной и для ряда развитых в промышленном отношении стран. Это закономерно — сначала должны возникнуть практические проблемы, а затем будут найдены их решения. Начало политических изменений в ситуации сильного отставания стран СНГ в области информатизации в конце 1980-х годов создало благодатную почву для резкого преодоления указанной тенденции.

Пример развитых стран, возможность приобретения системного программного обеспечения и компьютерной техники вдохновили российских пользователей на проведение активной работы по внедрению новых информационных технологий.

Включение массового потребителя, заинтересованного в оперативной обработке данных и других достоинствах современных информационно-вычислительных систем, в решение проблемы компьютеризации привело к очень высоким темпам развития этой области в России и других странах СНГ. Однако естественное совместное совершенствование средств автоматизации обработки информации и средств защиты информации в значительной степени нарушилось, что стало причиной массовых компьютерных преступлений. Ни для кого не секрет, что такие преступления в настоящее время составляют одну из актуальных проблем.

Использование средств защиты информации зарубежного производства не могло исправить данный перекос, поскольку поступавшие на рынок России продукты этого типа не соответствовали современным требованиям из-за существовавших экспортных ограничений, принятых в США — основном производителе средств защиты информации. Другим аспектом, имеющим первостепенное значение, является то, что подобная продукция должна пройти установленную процедуру сертифицирования в уполномоченных на проведение таких работ организациях. Проверка алгоритмов шифрования, программного обеспечения и электронных устройств на предмет различного рода "закладок" представляет собой чрезвычайно трудоемкую задачу. Последние исследования криптографов показали возможность разработки алгоритмов шифрования с "потайными дверями", которые практически невозможно обнаружить за разумное время даже высокопрофессиональным специалистам.

Сертификаты иностранных фирм и организаций никак не могут быть заменой отечественным. Сам факт использования зарубежного системного и прикладного программного обеспечения в критических областях создает потенциальную угрозу информационным ресурсам. Применение иностранных средств защиты без должного анализа соответствия выполняемым функциям и уровню обеспечиваемой защиты может многократно осложнить ситуацию.

Форсирование процесса информатизации требует адекватного обеспечения потребителей средствами защиты. Отсутствие на внутреннем рынке достаточного количества средств защиты информации, циркулирующей в компьютерных системах, значительное время не позволяло в необходимых масштабах осуществлять мероприятия по качественной защите данных. Ситуация усугублялась отсутствием достаточного количества специалистов в области защиты информации, поскольку они, как правило, готовились только для специальных организаций. Реструктурирование последних, связанное с изменениями, протекающими в России, привело к образованию независимых организаций, специализирующихся в области защиты информации, и увеличению потребности в специалистах по защите информации. Как следствие, возникла конкуренция между организациями, приведшая к появлению весьма большого количества сертифицированных отечественных средств защиты электронной информации.

Одной из важных особенностей массового использования информационных технологий является то обстоятельство, что для эффективного решения проблемы

защиты национального информационного ресурса необходимо рассредоточение мероприятий по защите данных среди массовых пользователей. Информация должна быть защищена в первую очередь там, где она создается, собирается, перерабатывается и теми организациями, которые несут непосредственный урон от несанкционированного доступа к своим данным. Указанный принцип рационален и эффективен: защита интересов отдельных организаций — это базовая составляющая защиты интересов государства в целом.

1.2. Проблематика криптографии

1.2.1. Традиционные вопросы криптографии

Слово "криптография" в переводе с греческого языка означает "тайнопись", что вполне отражает ее первоначальное предназначение. Примитивные с позиций сегодняшнего дня криптографические методы известны с древнейших времен и длительное время рассматривались скорее как некоторые ухищрения, чем строгая научная дисциплина. Классической задачей криптографии является обеспечение обратимости преобразования некоторого *исходного текста (открытого текста)* в кажущуюся случайной последовательность знаков, называемую *шифртекстом (закрытым текстом)*, или *криптограммой*. При этом шифртекст может содержать как новые знаки, так и уже имеющиеся в исходном сообщении. Количество знаков в криптограмме и в исходном тексте в общем случае может различаться. Непременным требованием является возможность однозначного и в полном объеме восстановления исходного текста, используя лишь некоторые логические действия с символами шифртекста. В далекие времена надежность сохранения информации в тайне определялась секретностью самого метода преобразования.

Однако секретность алгоритма принципиально не может обеспечить его *безусловную стойкость*, т. е. невозможность чтения криптограммы противником, обладающим бесконечными вычислительными ресурсами. Поскольку секретные алгоритмы не доступны для проведения широкомасштабных криптоаналитических исследований, то по сравнению с открытыми алгоритмами имеется значительно более высокая вероятность того, что впоследствии будут найдены уязвимые места и эффективные способы их взлома. В связи с этими обстоятельствами в настоящее время наиболее широко используются открытые алгоритмы, прошедшие длительное тестирование и обсуждение в открытой криптографической литературе.

Стойкость современных криптосистем основывается не на секретности алгоритма, а на секретности некоторой информации сравнительно малого размера, называемой *секретным ключом*. Ключ используется для управления процессом криптографического преобразования (шифрования) и является легко сменяемым элементом криптосистемы. Ключ может быть заменен пользователями в произ-

вольный момент времени, тогда как сам алгоритм шифрования является долговременным элементом криптосистемы и связан с длительным этапом разработки и тестирования.

При прочих равных условиях отсутствие полных данных об алгоритме шифрования существенно (при адекватной его реализации) затрудняет проведение криптоаналитической атаки. Поэтому были предложены современные шифры, в которых непосредственно действующий алгоритм шифрования является легко сменяемым элементом и выбирается случайно. При этом общая структура криптосистемы открыта для детального обсуждения, что позволяет оценить ее стойкость в целом. Такие шифры реализуются как гибкие криптосистемы, в которых алгоритм, действующий в сеансе шифрования, формируется по специальному *алгоритму инициализации (предвычислений)*. Этот алгоритм является открытым, а сам действующий алгоритм шифрования — неизвестным и зависимым от секретного ключа пользователя.

Прошли многие века, в течение которых криптография была предметом избранных — жрецов, правителей, крупных военачальников и дипломатов. Несмотря на малую распространенность, использование криптографических методов вскрытия шифров противника оказывало существенное воздействие на исход важных исторических событий. Известен не один пример, когда переоценка используемых средств шифрования приводила к военным и дипломатическим поражениям. Несмотря на применение криптографических методов в важных областях, эпизодическое использование криптографии не могло даже близко подвести ее к той роли, которую она имеет в современном обществе. Своим превращением в научную дисциплину криптография обязана потребностям практики и развитию электронных информационных технологий.

Пробуждение значительного интереса к криптографии и ее последующее развитие началось в XIX веке, что связано с зарождением электросвязи. В XX столетии секретные службы большинства развитых стран стали относиться к этой дисциплине как к обязательному инструменту своей деятельности.

Говоря об исторических аспектах научных исследований в области криптографии, необходимо отметить тот факт, что весь период с древних времен до 1949 года можно назвать донаучным периодом, поскольку средства закрытия письменной информации не имели строгого математического обоснования. Поворотным моментом, придавшим криптографии научность и выделившим ее в отдельное направление математики, явилась публикация в 1949 году статьи К. Э. Шеннона "Теория связи в секретных системах" [126]. Указанная работа послужила основой развития *одноключевых симметричных криптосистем*, в которых предполагается обмен секретными ключами между корреспондентами. Впоследствии с учетом особенностей построения симметричные шифры были разделены на две криптосистемы: *поточные* и *блочные шифры*. Отличительная особенность первых состоит в преобразовании каждого символа в потоке исходных данных, тогда как вторые осуществляют последовательное преобразование целых блоков данных.

Фундаментальным выводом из работы Шеннона стало определение зависимости надежности алгоритма от размера и качества секретного ключа, а также от *информационной избыточности* исходного текста. Шеннон ввел формальное определение информации и функции ненадежности ключа как его неопределенности при заданном количестве известных битов закрытого текста. Кроме того, им было введено важное понятие *расстояния единственности* как минимального размера текста, для которого еще возможно однозначное раскрытие исходного текста. Было показано, что расстояние единственности прямо пропорционально длине ключа и обратно пропорционально избыточности исходного текста. Следствием работы К. Э. Шеннона стало доказательство наличия теоретически стойких шифров, как, например, шифр Вернама.

Другим фундаментальным толчком развития криптографии явилась публикация в 1976 году статьи У. Диффи и М. Е. Хеллмана "Новые направления в криптографии" [62]. В этой работе впервые было показано, что секретность передачи информации может обеспечиваться без обмена секретными ключами. Тем самым была открыта эпоха *двухключевых (асимметричных) криптосистем*, разновидностями которых являются системы электронной цифровой подписи, тайного электронного голосования, защиты от навязывания ложных сообщений, электронной жеребьевки, идентификации и аутентификации удаленных пользователей и ряд других систем.

Последние годы на базе совершенствования электронных технологий появились новые теоретические разработки в области *квантовой криптографии* [41], основанной на принципах неопределенности Гейзенберга.

Наряду с развитием криптографических систем совершенствовались и методы, позволяющие восстанавливать исходное сообщение, исходя из шифртекста и другой известной информации, получившие название *криптоанализа*. Успехи криптоанализа приводили к ужесточению требований к криптографическим алгоритмам. Принципиально важным вопросом криптографии всегда была надежность криптосистем. Эта проблема допускала различное трактование на протяжении всей истории криптографии.

Голландский криптограф Керкхофф (1835—1903) впервые сформулировал *правило стойкости шифра*, в соответствии с которым:

- ❒ весь механизм преобразований считается известным злоумышленнику;
- ❒ надежность алгоритма должна определяться только неизвестным значением секретного ключа.

Второе требование означает, что оппонент не сможет разработать методы, позволяющие снять защиту или определить истинный ключ, за время существенно меньшее, чем время *полного (тотального) перебора* всего множества возможных секретных ключей.

По всей видимости, одной из задач оценки стойкости шифра по Керкхоффу было осознание необходимости испытания криптосхем в условиях, более благоприят-

ных для атаки, по сравнению с условиями, в которых, как правило, может действовать потенциальный нарушитель. Правило Керкхоффа стимулировало появление более качественных шифрующих алгоритмов. Можно сказать, что в нем содержится первый элемент стандартизации в области криптографии, поскольку предполагается разработка открытых способов преобразований. В настоящее время это правило интерпретируется более широко: *все долговременные элементы системы защиты должны предполагаться известными потенциальному злоумышленнику*. В последнюю формулировку криптосистемы входят как частный случай систем защиты. В расширенном понимании правила Керкхоффа предполагается, что все элементы криптосистем защиты подразделяются на две категории — долговременные и легко сменяемые элементы. К *долговременным* относятся те элементы, которые связаны со структурой криптосистем защиты и заменяются только специалистами. К *легко сменяемым* относятся элементы криптосистемы, которые предназначены для частого модифицирования в соответствии с заданным порядком. Легко сменяемыми элементами шифра являются, например, секретный ключ, пароль, идентификатор и т. п. Правило Керкхоффа отражает тот факт, что надлежащий уровень секретности зашифрованной информации должен быть обеспечен только за счет неизвестных легко сменяемых элементов шифра. Действительно, долговременные элементы системы защиты трудно сохранить в тайне, поэтому система должна быть стойкой в случае, когда они являются известными атакующему.

Согласно современным требованиям, криптосистемы с секретным ключом, включая шифры с ключом ограниченного размера (128—256 бит), должны быть стойкими к криптоанализу на основе известного алгоритма, большого объема известного открытого текста и соответствующего ему шифртекста. Несмотря на эти общие требования, шифры, используемые специальными службами, сохраняются, как правило, в секрете. Это обусловлено необходимостью иметь дополнительный запас прочности защиты секретной информации, поскольку в настоящее время создание криптосистем с доказуемой стойкостью является предметом развивающейся теории и представляет собой достаточно сложную проблему. Чтобы избежать возможных слабостей, алгоритм шифрования может быть построен на основе хорошо изученных и проверенных на практике принципов и способов преобразования. Ни один серьезный современный пользователь не будет полагаться только на надежность сохранения в секрете своего алгоритма, поскольку крайне сложно гарантировать отсутствие вероятности того, что информация об алгоритме станет известной злоумышленнику.

Обоснование надежности используемых систем осуществляется как теоретически, так и экспериментально при моделировании криптоаналитических атак с привлечением группы опытных специалистов, которым предоставляются значительно более благоприятные условия по сравнению с условиями, возникающими на практике в предполагаемых областях применения криптоалгоритма. Например, кроме шифртекста и алгоритма преобразований криптоаналитикам предоставляется весь или часть исходного текста, несколько независимых шифртекстов,

полученных с помощью одного и того же ключа, или шифртексты, полученные из данного открытого текста с помощью различных ключей. Оценивается стойкость испытываемой криптосистемы ко всем известным методам криптоанализа, и, по возможности, разрабатываются новые подходы к ее вскрытию. Если в этих условиях криптосистема оказывается стойкой, то она рекомендуется для конкретного применения.

В современном криптоанализе рассматриваются атаки на засекречивающие системы на основе следующих известных данных:

- шифртекста;
- открытого текста и соответствующего ему шифртекста;
- выбранного открытого текста;
- выбранного шифртекста;
- адаптированного открытого текста;
- адаптированного шифртекста.

Кроме того, рассматриваются следующие методы инженерного (технического) криптоанализа с использованием перечисленных выше известных данных:

- преднамеренно генерируемых аппаратных ошибок;
- замеров потребляемой мощности;
- замеров времени вычислений.

Мы детально перечислили типы атак на криптосистемы, предназначенные для шифрования данных с целью защиты от несанкционированного чтения. По отношению к иным видам криптосистем существует ряд других атак, которые будут рассмотрены ниже. А сейчас рассмотрим перечисленные типы атак подробнее.

- В случае *криптоанализа на основе известного шифртекста* считается, что противник знает механизм шифрования и ему доступен только шифртекст. Это соответствует модели внешнего нарушителя, который имеет физический доступ к линии связи, но не имеет доступ к аппаратуре зашифрования и расшифрования.
- При *криптоанализе на основе известного открытого текста* предполагается, что криптоаналитику известен шифртекст и некоторая часть исходного текста, а, в частных случаях, и соответствие между шифртекстом и исходным текстом. Возможность проведения такой атаки складывается при зашифровании документов, подготовленных с использованием стандартных форм, в условиях, когда определенные блоки данных известны и повторяются. В ряде современных средств защиты компьютерной информации используется режим глобального шифрования, в котором вся информации на встроенном магнитном носителе записывается в виде шифртекста, включая главную корневую запись, загрузочный сектор, системные программы и пр. При хищении

такого носителя (или компьютера) легко установить, какая часть криптограммы соответствует стандартной системной информации, и получить большой объем известного исходного текста для выполнения криптоанализа.

- В нападениях *на основе выбранного открытого текста* предполагается, что криптоаналитик противника может ввести специально подобранный им текст в шифрующее устройство и получить криптограмму, образованную под управлением секретного ключа. Это соответствует модели внутреннего нарушителя. На практике такая ситуация может возникнуть, когда в атаку на шифр вовлекаются лица, которые не знают секретного ключа, но в соответствии со своими служебными полномочиями имеют возможность использовать шифратор для закрытия передаваемых сообщений. Для осуществления такой атаки могут быть использованы также технические работники, готовящие формы документов, электронные бланки и др.
- *Криптоанализ на основе выбранного шифртекста* предполагает, что противник (оппонент) имеет возможность использовать для расшифрования сформированные им самим шифртексты, которые выбираются специальным образом, чтобы по полученным на выходе дешифратора текстам он мог вычислить секретный ключ шифрования с минимальной трудоемкостью.
- *Атака на основе адаптированных текстов* соответствует случаю, когда атакующий многократно подставляет тексты для зашифрования (или расшифрования), причем каждую новую порцию данных выбирает в зависимости от полученного ранее результата преобразования. Этот вид атаки является наиболее благоприятным для нападающего.

В настоящее время к наиболее мощным типам криптоаналитических атак на основе выбранных или адаптированных текстов относятся *дифференциальный (разностный) криптоанализ* (ДКА) [44] и *линейный криптоанализ* (ЛКА) [86], а также производные от них методы.

При тестировании новых криптосистем особый интерес представляют нападения на основе известного *секретного* ключа или *расширенного (рабочего)* ключа шифрования. Будем делать различие между секретным ключом и расширенным ключом, поскольку секретный ключ не всегда непосредственно используется в преобразованиях шифруемого текста, а часто служит только для выработки расширенного ключа, который и используется в процессе шифрования. Существуют шифры (например, блочный шифр ГОСТ 28147-89), в которых секретный ключ непосредственно используется при шифровании данных, т. е. секретный ключ служит и рабочим ключом шифрования. Очевидно, что расширенный ключ является секретным элементом шифра. При проведении криптоанализа на основе известных элементов ключа (секретного или расширенного) предполагается, что криптоаналитик имеет информацию о некоторой части рабочего ключа. Чем больше известная доля ключа, при которой шифр оказывается стойким, тем меньше опасений будет вызывать шифр в реальных условиях атаки, когда ключ атакующему неизвестен, но осуществляется попытка восстановить его элементы.

При сравнении двух шифров предпочтение можно дать тому шифру, который имеет лучшие показатели по указанному критерию.

Одним из направлений построения скоростных программных шифров является использование зависимости алгоритма шифрования от секретного ключа. В таких криптосистемах конкретная модификация алгоритма шифрования сменяется одновременно с заменой секретного ключа и атакующему неизвестна. Подобные шифры получили названия *недетерминированных*, или *гибких* шифров. При тестировании гибких шифров представляется разумным проводить анализ их стойкости с использованием атаки на основе *выбранной модификации алгоритма* шифрования. В этом варианте криптоаналитику предоставляется возможность анализировать самую слабую, по его мнению, модификацию из числа потенциально реализуемых модификаций криптоалгоритма. Затем для выбранной модификации проводится криптоанализ на основе специально подобранных текстов, в том числе может рассматриваться и вариант *атаки с частично известным ключом* шифрования. Если не удается найти слабую модификацию криптоалгоритма, то анализируемый гибкий шифр можно признать криптографически стойким.

1.2.2. Современные приложения

Значение криптографии выходит далеко за рамки обеспечения секретности данных. По мере все большей автоматизации процессов передачи и обработки информации и интенсификации информационных потоков криптографические методы приобретают уникальное значение. Новые информационные технологии в своей основе имеют двухключевую криптографию, которая позволяет реализовать протоколы, предполагающие, что секретный ключ известен только одному пользователю, т. е. протоколы, ориентированные на взаимное недоверие взаимодействующих сторон. Отметим основные приложения современной криптографии.

- Защита от несанкционированного чтения (или обеспечение конфиденциальности информации).
- Защита от навязывания ложных сообщений (умышленных и непреднамеренных).
- Аутентификация законных пользователей.
- Контроль целостности информации.
- Аутентификация информации.
- Электронная цифровая подпись.
- Системы тайного электронного голосования.
- Электронные деньги.
- Электронная жеребьевка.
- Защита от отказа факта приема сообщения.

- Одновременное подписание контракта.
- Защита документов и ценных бумаг от подделки.

Первое направление приложений было раскрыто выше. Поясним кратко остальные области применений криптографии.

Имитозащита — это понятие связано с защитой от навязывания ложных сообщений путем формирования в зависимости от секретного ключа специальной дополнительной информации, называемой *имитовставкой*, которая передается вместе с криптограммой. Дело тут в том, что само по себе шифрование данных недостаточно для защиты от навязывания ложного сообщения, хотя во многих случаях законный получатель, анализируя семантику сообщения, может легко определить, что криптограмма была модифицирована или подменена, например, при передаче по линии связи. Однако при искажении цифровых данных и в некоторых других случаях обнаружить факт изменения данных по семантике крайне сложно. Одним из способов защиты от навязывания ложного сообщения путем целенаправленного или случайного искажения шифртекста и является имитозащита. Для вычисления имитовставки используется алгоритм, задающий зависимость имитовставки от каждого бита сообщения. При этом могут быть использованы следующие два варианта:

- вычисление имитовставки по открытому тексту;
- вычисление имитовставки по шифртексту.

Чем больше длина имитовставки, тем меньше вероятность того, что искажение шифртекста не будет обнаружено санкционированным (законным) получателем. Нарушитель может модифицировать шифртекст, но поскольку он не знает секретного ключа, то он не может вычислить новое значение имитовставки, соответствующее модифицированному сообщению. Нарушитель либо не меняет имитовставки, либо подменяет ее на случайное значение. Если используется алгоритм вычисления имитовставки с хорошими криптографическими свойствами, то вероятность того, что факт изменений не будет обнаружен законным получателем, составляет $P = 2^{-n}$, где n — длина имитовставки в битах.

Аутентификация законных пользователей заключается в распознавании пользователей, после чего им предоставляются определенные права доступа к ресурсам вычислительных и автоматизированных информационных систем. Аутентификация основана на том, что законные пользователи обладают некоторой информацией, которая является неизвестной для посторонних. Частным вариантом использования процедуры аутентификации является парольная защита входа в компьютерную систему. Например, пользователь формирует некоторую случайную информацию и, сохраняя ее в секрете, использует как пароль. Пароль в явном виде не хранится в памяти ЭВМ или другого устройства, применяемого для выполнения аутентификации. Это требование направлено на то, чтобы потенциальный внутренний нарушитель не имел возможности считать чужой пароль и присвоить себе полномочия другого пользователя. Для того чтобы систе-

ма защиты могла идентифицировать легальных (санкционированных) пользователей, в памяти ЭВМ хранятся образы их паролей, вычисленные по специальному криптографическому алгоритму, реализующему так называемую одностороннюю функцию $y = F(x)$. Основное требование к односторонней функции состоит в том, чтобы сложность вычисления значения функции по аргументу была низкой, а сложность определения аргумента по значению функции — высокой (например, практически неосуществимой за 10 лет при использовании всех вычислительных ресурсов человечества).

Аутентификация пользователя на рабочей станции может быть осуществлена путем выполнения следующей последовательности шагов:

1. Запрос на ввод идентификатора со стороны системы защиты.
2. Ввод пользователем своего идентификатора (имени) NAME.
3. Запрос на ввод пароля со стороны системы защиты.
4. Ввод пользователем пароля P.
5. Вычисление системой защиты значения односторонней функции y, соответствующей значению аргумента $x = P$.
6. Сравнение системой защиты значения $F(P)$ со значением образа (S) пароля, соответствующего пользователю с идентификатором NAME.

Если $F(P) = S$, то система защиты предоставляет пользователю права доступа (полномочия), соответствующие идентификатору NAME. В противном случае в журнале учета работы пользователей регистрируется событие попытки несанкционированного доступа. Для того чтобы выдать себя за санкционированного пользователя, нарушитель должен ввести правильный пароль. Зная образ S, вычислительно невозможно определить пароль P. Если в системе защиты предусмотрены механизмы противодействия перехвату пароля с помощью программных или аппаратных закладок, а также через побочные электромагнитные излучения и наводки или акустический и оптический канал, то данный способ аутентификации пользователей обеспечивает высокую надежность защиты от захвата чужих полномочий.

Рассмотренный пример относится к аутентификации пользователей на рабочих станциях, т. е. к задаче защиты входа в ЭВМ. Для взаимной аутентификации удаленных рабочих станций принципиальным является предположение, что потенциальный злоумышленник прослушивает канал связи, а следовательно, описанный выше способ аутентификации неприемлем, поскольку недопустима передача пароля по незащищенному каналу. Аутентификация удаленных рабочих станций может быть выполнена по следующей схеме, основанной на использовании алгоритма шифрования **E** и общего секретного ключа ***K*** для удаленных станций **A** и **B**.

1. Станция **A** посылает запрос на соединение со станцией **B**.
2. Станция **B** посылает **A** случайное число R.

3. Станция **A** зашифровывает R по секретному ключу $\boldsymbol{K}$, получая шифртекст $C_a = \mathbf{E}_{\boldsymbol{K}}(R)$, и направляет станции **B** значение C_a.
4. Станция **B** вычисляет $C_b = \mathbf{E}_{\boldsymbol{K}}(R)$ и сравнивает значения C_b и C_a. Если $C_b = C_a$, то принимается решение, что запрос на установление сеанса связи отправлен станцией **A**, в противном случае связь прерывается.

Правильно зашифровать случайный текст может только тот, кто знает секретный ключ. Если нарушитель будет перехватывать правильные криптограммы от случайных чисел, то при длине числа R не менее 64 бит он за разумное время не встретит двух одинаковых чисел, а следовательно, не будет иметь возможности подставить ранее перехваченную правильную криптограмму. В этой схеме роль станции **B** может играть сервер локальной вычислительной сети. Отметим, что данная схема позволяет станции **B** удостовериться в том, что связь устанавливается со станцией **A**. Однако для станции **A** может существовать аналогичная проблема аутентификации станции **B**. В этом случае дополнительно осуществляется аналогичная процедура аутентификации, а именно — аутентификация станции **B** станцией **A**. Такая схема взаимного распознавания удаленных абонентов (станций) называется *протоколом рукопожатия*.

Контроль целостности информации — это обнаружение любых несанкционированных изменений информации, например, данных или программ, хранимых в компьютере. Имитозащита, в сущности, является важным частным случаем контроля целостности информации, передаваемой в виде шифртекста. В практических приложениях часто требуется удостовериться, что некоторые программы, исходные данные, базы данных не были изменены каким-либо несанкционированным способом, хотя сами данные не являются секретными и хранятся в открытом виде. Контроль целостности информации основан на выработке по некоторому криптографическому закону *кода обнаружения модификаций* (КОМ), имеющего значительно меньший объем, чем охраняемая от модифицирования информация. Основным требованием к алгоритму вычисления КОМ является задание зависимости значения КОМ от каждого бита двоичного представления всех символов исходного текста.

Соответствие информации своему эталонному состоянию (контроль целостности информации) выполняется следующим образом. При фиксации эталонного состояния, например, программного модуля File.exe вычисляется значение КОМ, соответствующее этому файлу. Полученное значение КОМ записывается в таблицу, которая будет использоваться при каждой процедуре проверки целостности информации. Пусть программа File.exe управляет сложным и ответственным технологическим процессом, а ее сбои могут привести к простоям, материальным и финансовым потерям. В таком случае перед каждым запуском программы целесообразно удостовериться в ее целостности. С этой целью каждый раз вычисляется КОМ и сравнивается с соответствующим значением, хранимым в таблице кодов. Этот способ эффективен для обнаружения непреднамеренных искажений данных, которые имеют случайный характер.

При преднамеренном модифицировании информации такая схема контроля целостности данных неприемлема, поскольку злоумышленник может обойти ее следующим образом. Он изменит по своему усмотрению данные, вычислит новое значение КОМ, соответствующее измененным данным, и внесет его в таблицу кодов взамен эталонного значения КОМ (соответствующего эталонному состоянию данных). Чтобы предотвратить такое нападение, можно использовать дополнительно один из следующих механизмов:

- секретный алгоритм вычисления КОМ;
- алгоритм вычисления КОМ с применением секретного ключа, от которого зависит значение КОМ;
- хранение таблицы кодов в защищенной области памяти или хранение таблицы кодов на переносных носителях, доступ к которым контролируется организационными мерами.

В первом случае сложно обеспечить секретность алгоритма, поскольку он является долговременным элементом криптосистемы. Третий случай требует существенных трудозатрат на выполнение дополнительных организационных мер. Наиболее приемлемым является использование второго механизма. Однако во всех трех случаях необходимо предусмотреть защиту от программных закладок.

Методы, используемые для контроля целостности, должны обеспечить чрезвычайно малую вероятность какого-либо умышленного или неумышленного изменения данных, при котором их кодовое представление осталось бы неизменным. В этой области задача криптоанализа заключается в том, чтобы на основе изучения слабых мест алгоритма генерации КОМ осуществить изменение исходной информации таким образом, чтобы значение контрольного кода не изменилось. Алгоритмы вычисления КОМ называются алгоритмами защитного *контрольного суммирования*, а вырабатываемое ими проверочное значение — *контрольной суммой*. Большую роль в современных криптографических протоколах и системах играют *хэш-функции*, которые представляют собой частный случай алгоритмов вычисления защитных контрольных сумм.

Аутентификация информации — это установление санкционированным получателем (приемником) того факта, что полученное сообщение послано санкционированным отправителем (передатчиком). Соблюдение заранее оговоренного протокола (набора правил и процедур) должно обеспечить максимальную вероятность этого факта. Очевидно, что при этом контролируется и целостность сообщения на возможность подмены или искажения. Принятый протокол должен обеспечить противодействие использованию потенциальным противником ранее переданных сообщений. В симметричных криптосистемах аутентификация осуществляется с применением одного или нескольких секретных ключей и защитных контрольных сумм. В асимметричных криптосистемах аутентификация выполняется с использованием открытых ключей. Чтобы это было возможным, при распределении открытых ключей осуществляется их аутентификация с использованием организационно-технических мероприятий.

Проблема аутентификации открытых ключей как одна из фундаментальных проблем криптографии предстала в явном виде с момента открытия *криптографии с открытым ключом* (двухключевой, или асимметричной криптографии) в середине 70-х годов прошлого века. Криптография с открытым ключом дала чрезвычайно удобное решение проблемы распределения секретных ключей, но ее использование требует осуществления процедуры аутентификации открытых ключей. Следует отметить, что проблема аутентификации ключей не является порождением двухключевой криптографии. Она в неявном виде всегда присутствовала в криптографии с секретным ключом. Действительно, при распределении секретных ключей по защищенному каналу одновременно осуществляется и их аутентификация. Например, при получении опечатанного пакета с секретным ключом получатель проверяет целостность пакета и печатей.

Если работа К. Э. Шеннона "Теория связи в секретных системах" [126] заложила фундамент формирования криптологии как науки, то открытие двухключевой криптографии ознаменовало собой ее переход в качественно новую фазу развития. Это послужило основой для наиболее полного решения проблем аутентификации информации и разработки систем электронной цифровой подписи, которые призваны придать юридическую силу документам и другим сообщениям, переданным в электронном виде.

Электронная цифровая подпись (ЭЦП) основывается на двухключевых криптографических алгоритмах, в которых предусмотрено использование *двух* ключей — *открытого* и *секретного*. Идея использования открытого (т. е. известного всем пользователям криптосистемы и потенциальному злоумышленнику) ключа является фундаментальной, поэтому двухключевые криптосистемы еще называют открытыми шифрами, а выполняемые преобразования — открытым шифрованием. Двухключевые криптоалгоритмы позволяют обеспечить строгую доказательность факта составления того или иного сообщения конкретными абонентами (пользователями) криптосистемы. Доказательство основано на том, что двухключевые криптосистемы функционируют в условиях, когда пользователю нет необходимости сообщать свой секретный ключ какому-либо второму субъекту. Факт использования секретного ключа при выработке подписи к тому или иному электронному документу проверяется с использованием *открытого* ключа. При этом знание открытого ключа не дает возможности выработать правильную цифровую подпись. Таким образом, ответственность за сохранность *секретного* ключа и соблюдение правил его использования лежит на самом владельце этого ключа. Секретный ключ позволяет составить сообщение со специфической внутренней структурой, связанной с подписываемым документом и открытым ключом. Тот факт, что сообщение имеет структуру, сформированную с помощью секретного ключа, проверяется посредством открытого ключа. Эта процедура называется проверкой электронной цифровой подписи. Вероятность того, что некоторое сообщение, составленное нарушителем, может быть принято за сообщение, подписанное каким-либо абонентом системы ЭЦП, является чрезвычайно низкой, порядка 10^{-30}.

Таким образом, процедура проверки ЭЦП с помощью открытого ключа позволяет с высокой степенью гарантии удостовериться в том, что полученное сообщение было составлено владельцем секретного ключа. Общедоступный открытый ключ формируется на основе секретного ключа или оба вырабатываются одновременно по специальным процедурам, причем определение секретного ключа по открытому ключу является вычислительно сложной математической задачей.

Под *вычислительно сложными* (трудно решаемыми) задачами понимаются задачи, заведомо имеющие решение, но требующие для его нахождения выполнения чрезвычайно большого числа операций вычислителя (ЭВМ или другого вычислительного устройства). Это число должно быть таким большим, что использование всех вычислительных устройств, которые могут быть вовлечены в единый вычислительный процесс, не позволит найти решение с существенной вероятностью (например, вероятностью 0,001) за обозримое время (десятилетия, столетия, тысячелетия и т. д.). Среднее число операций, которые необходимо выполнить для того, чтобы найти решение с помощью лучшего алгоритма, принимается за количественную меру сложности трудно решаемой задачи. Проблема оценки сложности заключается в ее зависимости от алгоритма поиска решения. Для разных алгоритмов в общем случае получаются различные значения сложности. Для конкретной трудно решаемой задачи нелегко доказать, что найден алгоритм с минимальной трудоемкостью (т. е. лучший алгоритм). Применение двухключевых шифров основано на предположении о существовании трудно решаемых задач, т. е. таких задач, для которых отсутствует решение со сравнительно низкой трудоемкостью.

На базе двухключевых криптографических алгоритмов предложены системы *тайного электронного голосования*, которые используют механизм слепой подписи, т. е. возможности подписать сообщение без ознакомления с его содержанием. Различные способы тайного электронного голосования имеют большие перспективы для совершенствования политической системы управления современным обществом с развитой информационной инфраструктурой.

Протокол слепой подписи позволяет построить различного рода системы *электронных денег*. Отличие электронных денег от перечислений денежных средств с использованием ЭЦП состоит в том, что они обеспечивают тайну покупки, осуществленной между покупателем и продавцом. Общественный интерес представляют также системы электронной жеребьевки, вариантом которой является игра в покер по телефону. В более широком плане могут быть созданы электронные игорные дома, которые будут гарантировать игрокам защиту от обмана на гораздо более высоком уровне, чем это делается в обычных игорных домах.

Рассмотрим простейший вариант проведения *электронной жеребьевки*. Пусть удаленные абоненты **A** и **B** хотят сыграть по телефону партию в шахматы, причем они желают справедливо разыграть цвет фигур, т. е. обеспечить равную вероятность выбора белых фигур для каждого из них. Криптография позволяет реализовать эту жеребьевку по указанной ниже схеме, в которой используется

односторонняя функция $y = F(x)$ и оговаривается, что абонент, угадывающий результат опыта с двумя равновероятными событиями, получает право первого хода.

1. Абонент **A** выбирает случайное число x_a, двоичное представление которого имеет, например, 80 разрядов, вычисляет значение $y_a = F(x_a)$ и сообщает величину y_a абоненту **B** (абонент **B** должен угадать четность числа x_a).
2. Поскольку используемая функция является однонаправленной (односторонней), то **B** не может по значению y_a определить x_a, поэтому он вынужден лишь угадать четность x_a. Пусть **B** останавливается на выборе варианта, что "x_a является четным числом", и сообщает свое предположение абоненту **A**.
3. Абонент **A** сообщает абоненту **B** число x_a.
4. Абонент **B** вычисляет значение $y = F(x_a)$, и если $y = y_a$, то **B** убеждается, что его партнер действительно предоставил для проверки первоначально выбранное число.

Если результаты жеребьевки касаются не розыгрыша цвета фигур в шахматной партии на досуге, а служат для раздачи карт в игре на деньги в покер по телефону (схема случайной раздачи карт по телефону является только технически более сложной), то можно предусмотреть дополнительно применение системы ЭЦП для подписывания всех сообщений, используемых в схеме раздачи карт по телефону и назначении ставок.

В качестве примера экономической выгоды электронной жеребьевки можно привести возможность ее использования на чемпионате мира (Европы и т. д.) по футболу (баскетболу, волейболу и т. д.). Для проведения обычной жеребьевки регулярно в одно и то же место съезжаются представители команд-участниц и международных спортивных организаций. При этом затрачивается время и несутся значительные финансовые затраты. В случае замены этой процедуры электронной жеребьевкой экономится время, а финансовые затраты сводятся к минимуму. Другими примерами ее использования являются проведение розыгрыша лотерей, организация справедливого распределения между сотрудниками различных учреждений путевок в престижные дома отдыха или других ограниченных ресурсов.

Криптографическая защита документов и ценных бумаг от подделки является наиболее надежным современным способом пресечения их фальсификации [114]. Она основана на уникальности любого конкретного материального носителя информации по своей микроструктуре. При наличии соответствующего оборудования (например, сканера с высокой разрешающей способностью в случае анализа бумаги) можно выявить уникальные особенности структуры каждого экземпляра носителя из одной и той же заводской партии. Криптографическая защита от подделки осуществляется следующим образом. Считывается информация об уникальных особенностях данного конкретного носителя, формируется цифровой паспорт, включающий содержание документа и информацию о микрострук-

туре носителя. Затем законный изготовитель документа, используя свой секретный ключ, вычисляет цифровую подпись паспорта и записывает на носителе паспорт и соответствующую ему цифровую подпись.

Проверка истинности документа выполняется путем сканирования микроструктуры материального объекта, на котором сформирован документ, считывания записанной на нем информации и проверки цифровой подписи изготовителя документа по открытому ключу, который является общедоступным и публикуется, например, в ряде официальных изданий или распространяется по официальным каналам. Изготовление фальшивого документа на другом материальном объекте или модифицирование содержания документа (и его цифрового паспорта) практически неосуществимо без знания секретного ключа, с помощью которого формируется подпись. Любая подделка будет обнаружена путем считывания цифрового паспорта и цифровой подписи, сопоставления паспорта с содержанием документа и проверки подписи по открытому ключу (предполагается, что в этом способе защиты документов от подделки используется криптографически стойкая система электронной цифровой подписи).

1.2.3. Защита информации на технологическом уровне

При разработке системы компьютерной безопасности для массового применения актуальным вопросом является защита информации на технологическом уровне. Особенность этой проблемы заключается в необходимости построения такой системы защиты, которая предоставляет возможность ее конфигурирования в соответствии с конкретными условиями применения.

Наиболее широкие возможности для проведения атаки на компьютерную систему (КС) имеют законные пользователи, которые технологически вовлечены в процесс работы КС и которые могут неправильно использовать предоставленные им права доступа. В общем плане сложность защиты информации на технологическом уровне состоит в необходимости предоставить операторам определенные права доступа и в то же время воспрепятствовать неправильному использованию предоставленных полномочий. Сильное ограничение в правах доступа приводит к снижению производительности работы пользователей, а следовательно, и всей КС. Для сохранения высокой производительности КС механизмы защиты должны функционировать в реальном масштабе времени, что позволит свести к минимуму непроизводительные задержки, связанные с защитными преобразованиями данных или контрольно-аналитическими функциями системы защиты.

Одними из наиболее эффективных и гибких механизмов защиты являются криптографические преобразования. В связи с этим очевидна задача разработки скоростных алгоритмов шифрования, ориентированных на программную реализацию. Решение этой задачи является базовым этапом разработки системы компьютерной безопасности массового назначения.

Таким образом, использование шифрования данных в системах компьютерной безопасности массового назначения носит принципиально технологический характер. Это требует получения максимально высокой скорости применяемых методов криптографического преобразования при условии обеспечения необходимого уровня их стойкости. Это всего лишь одна из проблем, стоящих перед разработчиками систем компьютерной безопасности. Другой специфической проблемой является создание программно-ориентированных алгоритмов преобразования различных типов.

С технологической точки зрения важно решить следующие задачи:

- разработки алгоритмов масштаба реального времени для файлового шифрования (алгоритмы прозрачного файлового шифрования);
- разработки алгоритмов для шифрования информации в загрузочном секторе жесткого диска (так называемые мини-алгоритмы);
- разработки скоростных алгоритмов для блочного шифрования данных перед их записью на встроенный магнитный носитель (алгоритмы прозрачного дискового шифрования).

Каждый из этих типов алгоритмов должен удовлетворять специфическим требованиям, связанным с особенностями их роли в системе защиты. Процедура дискового шифрования имеет большое значение ввиду интенсивности ее использования. Очевидно, что она должна выполняться автоматически и в масштабе реального времени, т. е. в прозрачном режиме.

Можно сформулировать различные требования к системе компьютерной безопасности, которые соответствуют точке зрения различных сторон, участвующих в процессе создания или использования систем защиты. С точки зрения оператора (пользователя) система компьютерной безопасности должна соответствовать следующим требованиям:

- не должна менять привычную технологию работ на ЭВМ;
- не должна требовать выполнения излишних дополнительных действий со стороны пользователей;
- должна позволять пользователю работать с необходимыми ему прикладными программами;
- не должна приводить к дополнительным временным задержкам.

С точки зрения экономической эффективности системы защиты можно выдвинуть следующие требования:

- невысокую цену;
- простоту сопровождения и эксплуатации, что позволит уменьшить штат обслуживающего персонала;
- полнофункциональность, что позволит уменьшить число используемых средств защиты для обеспечения комплексной безопасности;

Рис. 1.1. Общие типы угроз компьютерной безопасности и анализ потенциального ущерба

Рис. 1.2. Классификация внешних угроз и их целей

Рис. 1.3. Классификация внутренних угроз и их целей

- возможность установки системы защиты без приостановки технологического процесса обработки данных;
- функционирование в масштабе реального времени.

Для разработки системы защиты, удовлетворяющей перечисленным выше требованиям, представляется целесообразным ориентироваться на следующие конструктивные принципы:

- тотальное дисковое шифрование;
- многоуровневое шифрование (дисковое шифрование, прозрачное файловое шифрование, шифрование по запросу);
- сохранение базовой операционной системы в неизменном виде;
- контроль целостности информации в режиме реального времени.

Общие типы угроз компьютерной безопасности представлены на рис. 1.1.

Рис. 1.4. Задачи защиты и механизмы их реализации

Внешние, внутренние и комбинированные угрозы обусловливают три основных типа потенциального ущерба: нарушение конфиденциальности информации, нарушение целостности информации и нарушение работоспособности КС. На рис. 1.2 показаны различные виды внешних угроз и их цели. Внутренние угрозы (рис. 1.3) ориентированы на те же цели, что и внешние угрозы, хотя они реализуются с помощью других способов атаки.

В общем случае механизмы компьютерной безопасности должны обеспечить конфиденциальность, целостность данных и защиту от отказов. Тремя главными механизмами защиты (рис. 1.4) являются шифрование данных, контроль за процедурой загрузки и использование защитных криптографических контрольных сумм. Эти механизмы необходимо внедрить в систему защиты в различных формах в зависимости от особенностей проектирования системы защиты для конкретных областей применения. Основные механизмы должны сочетаться с дополнительными (рис. 1.5), обеспечивающими полноту системы безопасности и ее эффективность для защиты информации на технологическом уровне.

Рис. 1.5. Дополнительные механизмы защиты и их задачи

Примером системы защиты, ориентированной на компьютерную безопасность на технологическом уровне, является широко известная в России система SPECTR-Z, криптографические механизмы которой рассмотрены в *гл. 2*.

1.3. Основы одноключевой криптографии

1.3.1. Условная и безусловная секретность

Одна из замечательных теоретических работ в криптографии была опубликована К. Э. Шенноном в открытой печати в конце 40-х годов прошлого века. Эта фундаментальная работа [126], посвященная теоретическому анализу секретных систем (шифров), положила начало формированию современной криптологии и явилась базой для разработки новых криптосистем. К. Э. Шеннон рассматривает шифрование как отображение исходного сообщения в зашифрованное (крипто-

грамму): $C = F_i(M)$, где C — криптограмма, F_i — отображение, M — исходное сообщение, индекс i соответствует конкретному используемому ключу. Для того чтобы была возможность однозначного расшифрования сообщения, отображение F_i должно иметь единственное обратное отображение, такое, что $F_i F_i^{-1} = I$, где I — тождественное отображение: $M = F_i^{-1}(C)$.

Предполагается, что источник ключей является статистическим процессом или устройством, которое задает отображения $F_1, F_2, \ldots, F_{N_1}$ с вероятностями $p_1, p_2, \ldots, p_{N_1}$, а число возможных сообщений N_2 конечно, и сообщения $M_1, M_2, \ldots, M_{N_2}$ имеют априорные вероятности $q_1, q_2, \ldots, q_{N_2}$ (q_i — это вероятность, которая может быть приписана тому событию, что перехваченная криптограмма содержит сообщение M_i, без проведения криптоанализа).

Рассмотрим простейший шифр, в котором исходный алфавит сообщения совпадает с множеством знаков ключа и множеством знаков криптограммы, а шифрование выполняется путем последовательной замены знаков исходного сообщения на знаки криптограммы, в зависимости от очередного значения знака ключа. В этом случае сообщение, ключ и криптограмма представляются в виде последовательности букв одного и того же алфавита: $M = (m_1, m_2, \ldots, m_n)$, $K = (k_1, k_2, \ldots, k_n)$, $C = (c_1, c_2, \ldots, c_n)$.

Текущий шаг шифрования описывается соотношением $c_i = f(m_i, k_i)$. В практических криптосистемах длина ключа обычно значительно меньше длины шифруемого сообщения, поэтому часто последовательность (называемая ключевой гаммой) $k_1, k_2, \ldots, k_n$ вычисляется на основе некоторого первичного ключа меньшего размера или может быть даже периодической.

Задачей криптоаналитика является вычисление исходного сообщения по криптограмме при знании множества отображений $F_1, F_2, \ldots, F_{N_1}$. Существуют криптосистемы, для которых любой объем перехваченной информации недостаточен для того, чтобы найти шифрующие отображения, причем данная ситуация не зависит от вычислительной мощности, доступной криптоаналитику. Шифры такого типа называются *безусловно стойкими* (по К. Э. Шеннону — совершенно секретными). Строго говоря, безусловно стойкими являются такие шифры, для которых криптоаналитик (даже если он обладает бесконечными вычислительными ресурсами) не может улучшить оценку исходного сообщения M на основе знания криптограммы C по сравнению с оценкой при неизвестной криптограмме. Это возможно только в случае, когда M и C являются статистически независимыми, т. е. когда выполняется условие $P(M = M_i / C = C_i) = P(M = M_i)$ для всех возможных сообщений M. Последнее соотношение означает, что вероятность того, что в анализируемой криптограмме содержится сообщение M, не зависит от вида криптограммы, т. е. от конкретной последовательности знаков в шифртексте.

Безусловно стойкие криптосистемы существуют, что легко показать. Пусть в рассмотренном выше простейшем шифре используется алфавит из L букв, а текущие знаки криптограммы выбираются по формуле: $c_i = f(m_i, k_i) = (m_i + k_i) \bmod L$,

где каждому знаку c_i, m_i и k_i поставлен в соответствие их порядковый номер в алфавите. Выберем в качестве ключевой гаммы последовательность, состоящую из n случайных знаков $k_1k_2 \ldots k_n$, т. е. выберем случайный ключ, размер которого равен длине сообщения. Для генерации ключа воспользуемся некоторым физическим генератором случайных чисел, обеспечивающим на выходе равную вероятность появления любого элемента из множества чисел $\{1, 2, ..., L\}$. Вырабатываемое таким генератором число будем брать в качестве индекса выбираемого знака ключа. Выбранный нами источник обеспечивает равновероятность выбора любого ключа длины n. В этом случае вероятность выбора произвольного заданного ключа длины n составляет $P(K = K_i) = L^{-n}$.

Примененный нами способ зашифрования может преобразовать любое сообщение M_i в криптограмму C_i при использовании некоторого ключа K_i, значение которого зависит от M и C_i. Поскольку для всех i $P(K = K_i) = \text{const}$, то произвольное сообщение M_i может быть преобразовано с одинаковой вероятностью в любую криптограмму C_i и M_i, т. е. выполняется условие $P(M = M_i / C = C_i) = L^{-n}$.

Последнее означает, что данной криптограмме длины n с вероятностью L^{-n} может соответствовать любое исходное сообщение длины n. При шифровании нового сообщения будем брать новый случайный ключ. Описанные процедуры шифрования обеспечивают безусловную стойкость. Криптосистемы, использующие равновероятный случайный ключ, имеющий длину, равную длине сообщения, называются шифрами с *лентой однократного использования* или шифрами с *бесконечной ключевой гаммой.* На практике такие криптосистемы получили ограниченное применение, поскольку требуют передачи ключа большого объема.

Можно строго доказать, что для достижения безусловной стойкости необходимо использовать равновероятный случайный ключ, имеющий длину, равную длине сообщения, независимо от вида процедур шифрования. Это означает, что для шифров этого типа сами процедуры криптографического преобразования играют второстепенную роль, а принципиальным является использование бесконечного случайного ключа.

Криптосистемы второго типа характеризуются тем, что по мере того, как объем доступной криптоаналитику криптограммы возрастает при определенном значении $n = n_0$, существует единственное решение криптоаналитической задачи. Минимальный объем криптограммы, для которого имеется единственное решение, называется *расстоянием единственности*. В случае ленты однократного использования $n_0 \to \infty$. При конечной длине секретного ключа значение n_0 конечно. Заранее известно, что по криптограмме, имеющей размер больше расстояния единственности, можно найти единственное решение криптоаналитической задачи. Однако для криптоаналитика, обладающего ограниченными вычислительными ресурсами, вероятность найти это решение за время, в течение которого информация представляет ценность, чрезвычайно мала (10^{-30} и менее).

Шифры такого типа называются *условно стойкими*. Их стойкость основана на высокой вычислительной сложности решения криптоаналитической задачи.

Цель разработчика условно стойких криптосистем состоит в том, чтобы уменьшить затраты на процедуры шифрования и расшифрования и одновременно задать такой уровень сложности успешного решения криптоаналитической задачи, при котором для ее осуществления требуются настолько большие затраты, что делает нахождение решения экономически невыгодным. Задачи такого объема вычислений называются трудными или вычислительно сложными, а об их решении говорится, что оно является *вычислительно нереализуемым* (или вычислительно неосуществимым). Шифры, основанные на задачах, для которых нахождение решения является вычислительно нереализуемым, называются также *вычислительно стойкими*. Наибольшее практическое применение получили именно вычислительно стойкие криптосистемы.

Под стойкостью криптосистем такого типа понимается сложность решения задачи криптоанализа в определенных условиях. К. Э. Шеннон ввел понятие *рабочей характеристики* $W(n)$ шифра как среднего количества работы по нахождению ключа по известным n знакам криптограммы при использовании лучшего алгоритма криптоанализа. Количество работы может быть измерено, например, количеством операций, которые необходимо выполнить для вычисления ключа. Этот параметр непосредственно связан с алгоритмом вычисления ключа. Сложность определения $W(n)$ связана со сложностью нахождения лучшего способа раскрытия. Особый интерес представляет предельное значение $W(n)$ при $n \to \infty$. В настоящее время не известны вычислительно стойкие криптосистемы, для которых строго получена нижняя граница $W(\infty)$. Ввиду сложности таких оценок практические шифры характеризуют достигнутой оценкой рабочей характеристики $W'(\infty)$, которую получают для наилучшего из известных методов вычисления ключа.

К. Э. Шеннон предложил модель для оценки расстояния единственности, из которой получено соотношение $n_0 = H(K)/D$, где $H(K)$ — *энтропия ключа* (для случайного ключа это длина ключа в битах), D — *избыточность языка*, измеренная в битах на 1 знак. Данное соотношение можно переписать в виде $H(K) \le nD$, где $H(K)$ характеризует число неизвестных в двоичном представлении ключа, а nD — число уравнений, имеющихся для нахождения ключа. Если число уравнений меньше числа неизвестных, то решение системы уравнений неоднозначно, и криптосистема в этих условиях является безусловно стойкой. Если число уравнений больше числа неизвестных, то существует единственное решение, и криптосистема не является безусловно стойкой. Однако она может оставаться вычислительно стойкой при $n >> n_0$. Уровень стойкости вычислительно стойких криптосистем в определяющей степени зависит от конкретного типа шифрующих процедур (здесь мы исключаем случай выбора секретного ключа очень малого размера, при котором сложность перебора возможных ключей является низкой). Конкретные процедуры преобразования также определяют ход рабочей характеристики, т. е. конкретный вид зависимости $W(n)$.

В следующих разделах будут рассмотрены двухключевые шифры, которые определяют содержание современных направлений развития криптографии. Они по

своей природе являются вычислительно стойкими, но не безусловно стойкими криптосистемами. Предположение о существовании сложных вычислительных проблем является фундаментальным для современной криптографии.

1.3.2. Общие вопросы разработки шифров

В предыдущем разделе было показано, что шифр, обладающий безусловной стойкостью, может быть построен только при использовании случайного равновероятного ключа, длина которого равна длине исходного текста, причем для каждого нового сообщения применяется новый ключ. Поскольку ключ используется однократно и выбирается случайно, то можно говорить о бесконечном случайном ключе. Заметим, что при использовании бесконечного ключа нет необходимости применять какие-либо сложные процедуры преобразования знаков исходного текста в знаки шифртекста, поскольку достаточно применить простейшую операцию наложения знаков ключа на соответствующие знаки открытого текста (например, операцию побитового сложения по модулю два).

Под практической секретностью часто понимается трудоемкость решения задачи криптоанализа шифров с конечным ключом. Это относится к теоретической модели криптосистемы, которая абстрагируется от конкретных условий использования шифров, т. е. связано с теоретической оценкой вычислительной сложности решения задачи криптоанализа. Представляется предпочтительным характеризовать шифры с ключом конечного размера как вычислительно стойкие криптосистемы, поскольку под практической секретностью можно понимать также секретность, которая зависит не только от теоретического уровня стойкости, но и от организационно-технических условий применения шифров. Например, при использовании шифров с бесконечным ключом практическая секретность определяется различными каналами утечки, связанными с обработкой исходного сообщения в криптосистеме. Через эти отводные каналы может произойти перехват части сообщения или ключа. Реально даже при применении таких шифров имеется определенная вероятность того, что передаваемая информация станет известной нарушителю.

Под практической стойкостью в широком смысле можно понимать стойкость шифра с учетом большого числа факторов, имеющих место в условиях реального применения криптосистемы и связанных с обеспечением контроля целостности всех составных частей реальной криптосистемы: криптографического протокола, шифратора, защищенного канала, по которому передается секретный ключ, процедур управления ключами и элементов окружения (защищенных помещений, обслуживающего персонала, физических и технических средств защиты и т. п.) В теоретической модели криптосистемы рассматривается только стойкость, связанная с решением задачи криптоанализа, которая определяет максимально достижимый уровень секретности. (В конкретных условиях практической эксплуатации криптосистем их секретность может быть значительно ниже этого предельного значения.)

Поскольку реальная практическая секретность даже при применении шифров с бесконечной ключевой гаммой не может гарантировать абсолютную защиту информации, то вполне оправданно использовать удобные в практической эксплуатации криптосистемы с конечным ключом, основанные на высокой вычислительной сложности криптоаналитической задачи. Сам факт возможности раскрытия вычислительно стойких шифров ничего не говорит о трудоемкости такого раскрытия. Для осуществления раскрытия шифра на практике трудоемкость задачи криптоанализа является важнейшей характеристикой, поскольку ресурсы нападающего реально являются ограниченными.

При использовании шифров с конечным размером ключа конкретный выбор алгоритма шифрования является решающим для обеспечения практической стойкости. Именно процедуры преобразования определяют сложность криптоанализа. При этом необходимо иметь в виду, что размер ключа должен быть выбран достаточно большим, чтобы предотвратить тотальный перебор по пространству ключей (т. е. сделать вычислительно невозможным опробование всех возможных ключей при использовании современных вычислительных систем). При длине ключа 128 бит и более это условие выполняется.

Если вычислительно стойкий шифр не позволяет раскрыть ключ с вероятностью больше, чем вероятность утечки информации через каналы, связанные с практическими условиями функционирования криптосистемы, то их применение является предпочтительным. Кроме того, существует ряд задач, в которых шифры с бесконечным ключом неприменимы. (Например, в средствах защиты информации, циркулирующей в компьютерных системах, где используется шифрование всех данных, хранимых на встроенном магнитном носителе.) В этих случаях предпочтительными для использования являются шифры с конечным ключом (при условии, что такие криптосистемы обладают достаточно высокой вычислительной стойкостью).

При разработке вычислительно стойких шифров используют два общих принципа: рассеивания и перемешивания.

Перемешиванием называется распространение влияния одного знака открытого текста на много символов шифртекста, что обусловливает лавинный эффект (в случае блочных шифров требуется обеспечить распространение влияния каждого бита входного текста на все биты выходного текста). *Рассеиванием* называется шифрующее преобразование, нарушающее взаимосвязи статистических характеристик входного и выходного текста, т. е. маскировку статистических свойств исходного сообщения. Примером процедур, обеспечивающих рассеивание, является такая перестановка знаков открытого текста, которая приводит к выравниванию распределения избыточности по всему тексту. (Нужно отметить, что избыточность исходного текста играет большую роль при осуществлении криптоанализа на основе шифртекста, однако при криптоанализе на основе известного или специально подобранного текста ее рассмотрение теряет смысл.)

Для предотвращения возможности вычисления ключа по частям реализуют также принцип распространения влияния одного знака ключа на большое число зна-

ков криптограммы. Криптосистемы, в которых выполняются несколько последовательных простых процедур шифрования, последний принцип воплощают автоматически при реализации принципа рассеивания.

1.3.3. Композиционные и итеративные блочные шифры

В современных автоматизированных системах обработки информации во многих случаях предпочтительно применение блочных шифров. *Блочные шифры* представляют собой криптосистему, которая осуществляет шифрование информации блоками фиксированной длины, например, равной n бит. Этот вид криптографических преобразований называется блочным шифрованием. Для осуществления блочного шифрования данные представляются в виде последовательности n-битовых блоков. В реальных ситуациях файлы, определенные поля в электронных документах и другие виды электронных сообщений имеют произвольную длину и, как правило, не кратную длине блока, поэтому используется некоторый способ дополнения последнего блока данных.

Последний блок принято дополнять двоичным вектором (1, 0, 0, ..., 0), в котором количество нулей может быть от 0 до $(n - 2)$. Если длина последнего блока равна n, то к сообщению присоединяется дополнительный n-битовый блок, имеющий структуру (1, 0, 0, ..., 0). Этот способ позволяет однозначно распознать присоединенный двоичный вектор и отбросить его при необходимости. Используя такой способ дополнения сообщения до длины кратной n, любое сообщение M можно представить в виде последовательности (конкатенации) n-битовых подблоков M_i, а именно $M = M_1 \| M_2 \| \ldots \| M_i \| \ldots \| M_m$.

Каждый из блоков исходного сообщения может быть преобразован независимо от других блоков, поэтому при применении блочных шифров возможен произвольный доступ к зашифрованным массивам данных. Наиболее общим механизмом блочного шифрования является такой, в котором допустимо преобразование любого входного блока в любой выходной блок, причем размер выходного блока равен или больше размера входного блока. Блок шифртекста не может быть меньше блока открытого текста, т. к. в этом случае одному и тому же блоку шифртекста будут соответствовать несколько различных блоков открытого текста. Это означало бы неоднозначность расшифрования. Если длина выходного блока больше n, то одному и тому же блоку открытого текста могут соответствовать несколько различных блоков шифртекста. В этом случае расшифрование возможно и однозначно. (Примером таких криптосистем являются вероятностные шифры.) Поскольку увеличение длины зашифрованных данных вносит определенные ограничения на области применения, то в наиболее широко используемых шифрах размер выходных блоков равен размеру входных. Блочные шифры задают взаимно однозначное соответствие между возможными входными и выходными блоками. Поскольку входное и выходное множество блоков совпада-

ет, то шифрование задает некоторую подстановку на множестве чисел 0, 1, …, $2^n - 1$, которую можно представить в виде следующей таблицы:

$$\begin{pmatrix} 0 & 1 & 2 & \dots & 2^n - 1 \\ \mathbf{E}_K(0) & \mathbf{E}_K(1) & \mathbf{E}_K(2) & \dots & \mathbf{E}_K(2^n - 1) \end{pmatrix},$$

где $\mathbf{E}_K(M)$ — функция шифрования по ключу K, т. е. функция, задаваемая процедурами шифрования при использовании ключа шифрования K.

Функция шифрования ставит в соответствие блоку открытого текста M блок криптограммы C, что записывается в виде $C = \mathbf{E}_K(M)$. Для данного ключа реализуется одна подстановка. В общем случае различным ключам соответствуют различные подстановки. Если в шифре используется ключ длиной k бит, то этот шифр задает не более 2^k различных подстановок, что составляет обычно чрезвычайно малую долю от числа возможных подстановок, которое равно $2^n!$. Для того чтобы реализовать все возможные подстановки, необходимо использовать ключ длиной порядка $k = \log_2(2^n!) \approx n2^n$ бит.

Одним из статистических методов раскрытия шифров является *частотный криптоанализ*. Этот метод основан на исследовании частоты появления знаков в криптограмме и сопоставлении ее с частотой появления знаков в исходном тексте. Частотный метод позволяет раскрывать шифры, основанные на одноалфавитной подстановке, которые соответствуют блочному шифрованию при использовании входных блоков малого размера (например, при $n = 8$). С увеличением размера входного блока частотные свойства языка, на котором составлен открытый текст, становятся менее выраженными, но даже при $n = 16$ неравномерность частотных свойств исходного текста может быть эффективно использована для раскрытия шифра. При $n = 32$ частотный криптоанализ становится крайне сложным, и в некоторых случаях могут применяться блочные шифры с таким размером входа. Минимальной безопасной длиной блока принято считать значение $n = 64$. Чем больше размер входного блока, тем более высокая стойкость может быть достигнута. Однако для блоков большого размера усложняется производство устройств шифрования. При разработке американского стандарта DES определенным компромиссом между стойкостью и удобством реализации был выбор значения $n = 64$. На протяжении 25 лет этот размер был общепринятым. В настоящее время возможности микроэлектроники значительно выросли, и стандартным размером входного блока стало значение $n = 128$.

Сам по себе большой размер входного блока является только необходимым условием достижения высокой стойкости у разрабатываемого алгоритма. Проектирование стойких блочных шифров связано с использованием нелинейных преобразований с хорошими свойствами рассеивания и перемешивания или комбинированием линейных и нелинейных преобразований. Достоинством линейных преобразований является простота реализации, малое время их выполнения и удобство использования секретного ключа в качестве параметра преобразования.

Однако использование только линейных преобразований недостаточно для построения стойких шифров.

Один из способов достижения хорошего рассеивания и перемешивания состоит в построении составного *(композиционного)* шифра, включающего ряд последовательно используемых простых шифров, каждый из которых вносит небольшой вклад в рассеивание или в перемешивание. Данная идея построения композиционных шифров была предложена и обоснована К. Э. Шенноном. В композиционных шифрах шифрующие процедуры одного типа чередуются с процедурами другого типа. В качестве простых шифров могут быть использованы, например, подстановки (S), перестановки (T) и линейные преобразования (L). В этом случае результирующий шифр представляется в виде $F = T_n L_n S_n \ldots T_2 L_2 S_2 T_1 L_1 S_1$.

Секретный ключ может использоваться при выполнении процедур какого-то одного типа (T, L или S). Ключ может применяться также при осуществлении процедур всех (или нескольких) типов.

Рис. 1.6. Структура композиционного шифра на основе подстановок и перестановок

Простейший композиционный шифр показан на рис. 1.6, где блоки S обозначают операции подстановки над 4-битовыми подблоками входного сообщения, T — операцию перестановки над $4k$–битовым преобразуемым блоком данных, а L — операцию линейного преобразования, с помощью которой осуществляется смешивание шифруемых данных с секретным ключом, представленным в виде совокупности подключей $K_1, K_2, \ldots, K_r$. В данной r-каскадной криптосхеме процедура шифрования состоит в последовательном выполнении r раундов преобразования с использованием различных раундовых ключей. Выполнение операции подстановки состоит в замене входного 4-битового двоичного вектора на 4-битовый

выходной вектор в соответствии с некоторой таблицей, называемой таблицей подстановки. Представляя 4-битовые двоичные векторы их численными значениями (т. е. интерпретируя двоичный вектор двоичным числом), таблицу подстановки можно записать в виде:

$$\begin{pmatrix} 0 & 1 & 2 & \dots & 15 \\ \alpha_0 & \alpha_1 & \alpha_2 & \dots & \alpha_{15} \end{pmatrix},$$

где $\forall_i \ \alpha_i \in \{0, 1, \dots, 15\}$, а столбцы задают соответствие между входным 4-битовым значением (верхняя строка) и выходным 4-битовым значением (нижняя строка).

Для произвольно заданной таблицы подстановки легко записать таблицу, определяющую подстановку, которая является обратной по отношению к заданной. Аналогичным образом может быть задана и операция перестановки битов T и соответствующая ей обратная перестановка T^{-1}. Для линейного преобразования L также легко задать соответствующее ему обратное преобразование L^{-1}. Процедура расшифрования выполняется по схеме, приведенной на рис. 1.7.

Рис. 1.7. Схема процедуры расшифрования в композиционном шифре

Рассмотренный выше шифр относится к так называемым *итеративным шифрам*, в которых шифрование осуществляется как многократное выполнение типовой процедуры преобразования (именуемой *раундом шифрования* или *раундовой функцией* шифрования), представляющей собой композицию трех разнотипных простых преобразований. При этом в разных раундах используются разные ключи, называемые *раундовыми* ключами.

При реализации в виде быстродействующих устройств процедуры зашифрования или расшифрования в композиционном шифре будут выполняться с помощью

различных электронных схем. В дальнейшем мы рассмотрим специальные криптосхемы, которые обеспечивают возможность использования одной и той же электронной схемы для выполнения зашифрования и расшифрования, что делает аппаратную реализацию более экономичной. В таких шифрах смена режима шифрования осуществляется путем изменения очередности использования раундовых ключей.

Рис. 1.8. Криптосхема Фейстеля: *а* — зашифрование; *б* — расшифрование

Криптосхема Фейстеля (рис. 1.8) представляет собой общую схему построения n-битового блочного шифра на основе произвольной функции F с размером входа $n/2$. Важным достоинством данной структуры является то, что она задает один и тот же алгоритм для выполнения зашифрования и расшифрования. Задание конкретного режима шифрования определяется очередностью применения раундовых ключей. Смена режима шифрования обеспечивается изменением очередности использования раундовых ключей на обратную. Стойкость шифров, построенных по этой схеме, определяется свойствами раундовой функции F. Известно большое число различных шифров, основанных на этой схеме и отличающихся только числом раундов и устройством раундовой функции.

Замечательным является то обстоятельство, что для произвольной раундовой функции обеспечивается возможность расшифрования, т. е. криптосхема Фейстеля есть некоторый универсальный способ превращения произвольного отображения над двоичными векторами длины $n/2$ в подстановку (*биективное* отображение) над векторами длины n. Благодаря этому свойству разработчик криптоалгоритма свободен от необходимости ограничения выбора только таких преобразований, которые допускают обратные преобразования невысокой вы-

числительной сложности. При построении раундовой функции достаточно найти быстрые прямые преобразования, удовлетворяющие необходимым криптографическим свойствам. Эти же преобразования при смене расписания использования подключей будут выполнять корректную процедуру расшифровывания. Основной задачей при разработке шифров на основе криптосхемы Фейстеля является обеспечение раундовых преобразований с такими свойствами, которые обладают хорошими лавинными свойствами (влияние одного входного бита на многие выходные биты) и обеспечивают стойкость ко всем известным видам криптоатак.

Определенным недостатком криптосхемы Фейстеля является то, что при выполнении шифрования (т. е. вычислении раундовой функции) используется только половина преобразуемого блока данных. Этот недостаток устраняется в универсальных криптосхемах другого типа, основанных на использовании так называемых *управляемых* операций и рассматриваемых в *гл. 3—6*.

1.3.4. Управляемые операции — новый криптографический примитив

Общая характеристика управляемых операций

Современная прикладная криптография проявляет все более выраженную тенденцию к учету требований, связанных с особенностями электронных информационных технологий. В настоящее время использование шифрования для защиты информации принимает все более выраженный технологический характер. Технологический характер применения шифрования проявляется:

- в массовости применения;
- в разнообразии решаемых задач;
- в разнообразии условий применения;
- в специализации для решения конкретных задач.

Это обстоятельство привело к возрастанию требований по:

- *стойкости* — в разнообразных условиях применения становятся актуальными новые виды криптоаналитических атак, например, так называемое нападение на основе генерации аппаратных ошибок (Differential Fault Analysis);
- *скорости шифрования* — это продиктовано важностью сохранения высокой производительности компьютерных и телекоммуникационных систем при внедрении средств защиты;
- *экономичности аппаратной реализации* — это связано с тенденцией массового применения устройств шифрования.

Проведение конкурсов на разработку новых алгоритмов шифрования в США (конкурс AES), Европе (конкурс NESSIE) и Японии является проявлением признания технологической роли шифрования. Технологические применения на-

столько многообразны, что разработка новых специализированных алгоритмов шифрования, видимо, будет оставаться актуальной еще долгое время. Значительный интерес современной прикладной криптографии лежит в области поиска новых криптографических примитивов для построения блочных шифров, перспективных для технологических применений и обеспечивающих:

- высокую скорость;
- высокую стойкость;
- низкую сложность реализации.

Криптографическими примитивами, традиционно используемыми при создании одноключевых криптосистем, являются:

- подстановки;
- перестановки;
- арифметические и алгебраические операции;

  а также некоторые другие вспомогательные операции.

Особенно часто применяемой является операция подстановки, которая имеет наиболее общий характер. Данная операция является тем криптографическим примитивом, на котором основана стойкость большинства блочных шифров. Известны следующие варианты реализации подстановок.

- В программных и программно-аппаратных шифрах подстановки реализуются в виде таблиц подстановок, хранимых в памяти ЭВМ. В этом варианте легко реализуются подстановки над двоичными векторами длиной до 13 бит (требуемый объем памяти составляет около 10 Кбайт). Несмотря на большой объем оперативной памяти современных ЭВМ, реализация операций подстановок над двоичными векторами длиной более 16 бит представляется проблематичной.
- В аппаратных шифрах подстановки реализуются в виде комбинационных электронных схем. Подстановки общего вида над двоичными векторами длиной более 13 бит имеют высокую сложность реализации.

Достоинством операций подстановок общего вида является то, что принципиально могут быть найдены наилучшие по тем или иным криптографическим критериям подстановки. В случае подстановок малого размера (например 6×4) установлено множество эффективных подстановок. Однако для подстановок размера 8×8 и более выбор оптимальных подстановок является проблематичным. В связи с этим выбор подстановок большого размера в ряде шифров осуществляется через некоторые известные операции, обладающие определенными свойствами. Характерным примером является шифр SAFER, в котором подстановки определены через операцию возведения в дискретную степень и операцию дискретного логарифмирования в поле вычетов по модулю 257.

Ввиду имеющихся проблем при разработке скоростных блочных шифров на базе подстановок были предложены альтернативные решения. Одно из таких решений

представлено шифром RC5, в котором единственной нелинейной операцией является операция вращения (циклического сдвига), зависящая от преобразуемых данных, которая легко реализуется на современных массовых процессорах. Несмотря на крайнюю простоту построения, шифр RC5 оказался весьма стойким к линейному и дифференциальному криптоанализу. Теоретические исследования показали, что зависимость выбора модификаций операции вращения от преобразуемых данных является эффективным средством противодействия этим двум важнейшим типам атак. Благодаря своей эффективности циклический сдвиг, зависящий от преобразуемых данных, нашел применение в новых шифрах — RC6 и MARS.

Если фиксированная операция циклического сдвига как частный случай операции перестановки является линейной операцией, то *задание ее зависимости от преобразуемых данных* приводит к построению *новой нелинейной операции* с хорошими криптографическими свойствами. Видимо, кроме операции циклического сдвига, зависящей от преобразуемых данных, существуют и другие виды *управляемых операций*. Важными характеристиками последних являются их тип и количество различных вариантов, из которых осуществляется выбор текущей модификации, используемой для преобразования некоторого подблока данных. Второй параметр определяет, сколько дополнительных битов данных могут быть задействованы при выполнении управляемой операции над текущим преобразуемым n-битовым подблоком данных. В случае управляемой операции циклического сдвига имеются n модификаций. Несмотря на столь малое число модификаций, данная управляемая операция оказалась эффективным криптографическим примитивом. Можно ожидать, что весьма эффективными окажутся операции с существенно большим числом модификаций, например равным от 2^n до 2^{3n} и более. Такой управляемой операцией является операция перестановки битов, зависящая от преобразуемых данных, которая является обобщением операции управляемого циклического сдвига.

Самостоятельным важным направлением является разработка специальных *управляемых подстановочных операций* (controlled operational substitutions) для криптографических применений, в частности управляемых двухместных операций. Простейший вариант реализации операций такого типа представлен управляемым сумматором (рис. 1.9), позволяющим задать 2^n различных модификаций операции суммирования $Y = X *_V A$, включающих операцию поразрядного суммирования по модулю два ("$*_V$" = XOR) при $V = (0, 0, 0, \ldots, 0)$ и операцию суммирования по модулю 2^n ("$*_V$" = "+") при $V = (1, 1, 1, \ldots, 1)$ как частные модификации.

В отличие от табличных подстановок, может быть разработано много различных типов управляемых операций преобразования с достаточно большим размером входа (например, 32, 64 или 128 бит), которые легко реализуются в виде электронных микросхем. При этом операции фактически представляют собой частный подкласс подстановок. Однако этот подкласс подстановок относится к под-

становкам существенно большого размера. Это обстоятельство составляет объективные предпосылки для построения стойких скоростных шифров с низкой схемотехнической сложностью реализации.

$q_0 = 0$; и $\forall i$: $i \in 1, \ldots, n$; $u_i = q_{i-1} \vee v_i$;
$y_i = x_i \oplus a_i \oplus u_i$; $q_i = (x_i \oplus e)(a_i \oplus u_i) \vee a_i u_i$;

Рис. 1.9. Управляемый сумматор ($e = 0$ — "сложение", $e = 1$ — "вычитание")

Традиционные табличные подстановки, а также арифметические и прочие операции, которые изначально используются для решения других задач, не являются ориентированными на криптографическое применение. С криптографической точки зрения последние заключают в себе как положительные (например, операция поразрядного суммирования по модулю два проста для реализации и обладает высоким быстродействием), так и отрицательные (например, линейность) свойства. Для криптографического применения целесообразно разработать операции, оптимизированные в криптографическом отношении и обладающие специальными свойствами, которые необходимы для получения высокой стойкости алгоритмов шифрования. К прообразу таких операций можно отнести операцию циклического сдвига, зависящую от преобразуемых данных, которая была использована как базовый криптографический примитив в шифрах RC5 [44], RC6 и MARS [86]. Задание выбора текущей модификации такой операции в зависимости от преобразуемых данных определяет ее нелинейные свойства. Несмотря на то, что реализуется выбор всего из n (n — длина двоичного вектора, над которым выполняется операция циклического сдвига) различных модификаций, данный криптографический примитив оказался весьма эффективным. Его достоинства состоят в легкости программной реализации, нелинейности и увеличении эффективного размера входа до $\log_2 n$ бит (это количество битов данных, задающих выбор текущей модификации, т. е. управляющих битов).

В общем плане *управляемые перестановки* (УП) и *управляемые двухместные операции* (УДО) представляются существенно более эффективными для криптографических применений, поскольку они включают очень большое число реализуемых модификаций, что позволяет создать управляющий вход размером от n до $3n$ (а в некоторых случаях и более). Предварительные исследования алгебраических и вероятностно-статистических свойств УДО и УП показали их перспективность для построения стойких быстродействующих шифров [21, 115].

Структура и механизм действия операций управляемых перестановок и управляемых сумматоров являются достаточно наглядными, поэтому данные варианты управляемых операций воспринимаются как самостоятельный класс криптографических примитивов. Еще более широким классом управляемых операций представляются управляемые подстановочные операции. Однако данный тип операций не столь очевидным образом воспринимается как самостоятельный класс операций. В связи с этим уместно подчеркнуть, чем управляемые подстановочные операции отличаются от операций подстановок размера $m \times n$, где $m > n$. Последние будем называть *управляемыми подстановками*.

По своему содержанию и сути управляемые подстановочные операции являются специально проектируемыми операциями криптографической ориентации, выполняемыми над двумя и более двоичными векторами. Строятся такие операции по определенному правилу, позволяющему разрабатывать операции для преобразования двоичных векторов произвольного размера. При этом они обладают такой структурой, что сложность их аппаратной реализации растет примерно по линейному закону с увеличением размера преобразуемых двоичных векторов.

Управляемые подстановочные операции (УПО) характеризуются следующими признаками:

- использованием типовых булевых функций, задающих связь между входными и выходными битами;
- возможностью проектирования УПО для преобразования двоичных векторов размера от 32 до 256 бит;
- невысокой сложностью схемотехнической реализации;
- высоким быстродействием;
- возможностью теоретического обоснования выбора УПО определенного вида для произвольного размера входа.

На рис. 1.10 показаны возможные типы управляемых подстановочных операций.

Управляемые операции обладают следующими положительными особенностями:

- обеспечивают возможность участия всех битов преобразуемого блока данных при выполнении единой нелинейной операции;
- разрешают смену прямой операции на обратную путем инвертирования специального бита, задающего режим зашифрования или расшифрования;
- позволяют построить новые типы криптосхем, обеспечивающих смену режима преобразования путем изменения очередности использования подключей;
- обеспечивают возможность построения эффективных механизмов внутреннего развертывания ключа, что дает высокую скорость шифрования в приложениях с частой сменой секретных ключей.

Несмотря на изначальную ориентацию на аппаратную реализацию, проектирование эффективных управляемых операций потенциально может привести к суще-

ственному скачку в росте скорости программных шифров. Это связано с тем, что некоторые типы управляемых операций, например, управляемые перестановки, чрезвычайно эффективны как криптографический примитив и обладают очень низкой стоимостью схемотехнической реализации. Такое соотношение эффективности и стоимости реализации делают привлекательным для производителей массовых процессоров включение новой команды — управляемой перестановки — в состав стандартных команд процессора. Возможность обеспечения скорости программного шифрования до 800—2000 Мбит/с существенно повышает конкурентоспособность таких процессоров при минимальных схемотехнических затратах. Например, реализация операционного блока управляемых перестановок (БУП) с 64-битовым входом для преобразуемых данных и 192-битовым управляющим входом требует не более 1200 транзисторов, а реализация БУП с 32-битовым входом для данных и 80-битовым управляющим входом требует не более 1000 транзисторов.

Рис. 1.10. Типы управляемых подстановочных операций

1.4. Двухключевые криптосистемы

1.4.1. Система открытого распределения ключей

В 1976 г. У. Диффи и М. Е. Хеллман опубликовали статью, которая ознаменовала собой рождение двухключевой криптографии и привела к значительному росту числа открытых исследований в области криптографии. Она содержала ошеломляющий результат: *возможно построение практически стойких секретных сис-*

тем, которые не требуют передачи секретного ключа. У. Диффи и М. Е. Хеллман ввели понятие односторонней функции с потайным ходом. Под *односторонней* функцией f понимается функция $f(x)$, которая легко вычислима для любого значения аргумента x из области определения, однако для случайно выбранного y из области ее значений вычислительно сложно нахождение значения аргумента x, для которого выполняется $f(x) = y$. Использование таких функций для защиты входа в вычислительную систему путем одностороннего преобразования паролей было известно. Но как применить одностороннюю функцию в криптографических системах, когда даже законный получатель не сможет выполнить процедуру расшифрования? Для шифрования была предложена *односторонняя функция с потайным ходом (секретом)*.

Под односторонней функцией с потайным ходом понимается семейство обратимых функций f_z с параметром z, таких что для данного z можно найти алгоритмы $\mathbf{E}_z$ и $\mathbf{D}_z$, позволяющие легко вычислить значение $f_z(x)$ для всех x из области определения, а также $f_z^{-1}(y)$ для всех y из области значений, однако практически для всех значений параметра z и практически для всех значений y из области значений f_z нахождение $f_z^{-1}(y)$ вычислительно неосуществимо даже при известном $\mathbf{E}_z$. В качестве односторонней функции У. Диффи и М. Е. Хеллман предложили функцию дискретного возведения в степень

$$f(x) = \alpha^x (\mathrm{mod}\ p),$$

где x — целое число, $1 \le x \le p-1$, $(p-k)$-битовое простое число. Причем выбирается такое число $\alpha < p$, степени которого по модулю p представляют собой упорядоченное множество чисел $\{\alpha^1, \alpha^2, \ldots, \alpha^{p-1}\}$, являющееся некоторой перестановкой чисел $\{1, 2, \ldots, p-1\}$. (Такое число α называется первообразным корнем по модулю p.)

Даже для очень больших модулей p (например, при $k = 1024$ бит) для данного x легко вычислить значение этой функции. Процедура ее вычисления называется дискретным возведением в степень. Для выполнения этой процедуры достаточно выполнение около $2\log_2 p$ операций умножения k-битовых чисел (или $\log_2 p$ умножений и $\log_2 p$ делений $2k$-битовых чисел на k-битовые). Процедура дискретного возведения в степень основана на предварительном вычислении значений (по модулю p) $\alpha^1, \alpha^2, \alpha^4, \alpha^8, \ldots, \alpha^{2^{k-1}}$.

Обратной к функции дискретного возведения в степень является функция $f^{-1}(y)$, которая ставит в соответствие заданному значению y такое значение x, для которого выполняется условие $\alpha^x = y\ (\mathrm{mod}\ p)$. Задача нахождения такого x называется задачей дискретного логарифмирования (нахождения дискретных логарифмов). Дискретные логарифмы сложно вычисляются, когда число $p - 1$ содержит один большой простой множитель, например, когда оно представимо в виде $p - 1 = 2p'$, где p' — простое число. При этом условии трудоемкость задачи нахождения дискретного логарифма равна примерно $p^{1/2}$ умножений по модулю p.

Решение такой задачи является вычислительно неосуществимым при больших значениях k (например, при $k \geq 512$), а, следовательно, при указанных условиях, накладываемых на выбор чисел p и α, функция дискретного возведения в степень является односторонней.

Системой (методом) *открытого распространения ключей* Диффи-Хеллмана называется изложенный ниже способ использования дискретного возведения в степень для обмена секретными ключами между пользователями сети с применением только открытых сообщений.

Выбирается большое простое число p и соответствующий ему первообразный корень $\alpha < p$. (Для обеспечения стойкости рассматриваемой системы открытого шифрования на число p накладывается следующее условие — разложение числа $p - 1$ на множители должно содержать, по крайней мере, один большой простой множитель; размер числа p должен быть не менее 512 бит.)

Механизм распределения секретных ключей по открытому каналу состоит в следующем. Каждый абонент выбирает случайный секретный ключ x и вырабатывает открытый ключ y, соответствующий выбранному секретному ключу, в соответствии с формулой

$$y = \alpha^{x} \pmod p.$$

Для любого значения x легко вычислить y, однако при размере числа p равном 512 бит и более вычислительно неосуществимо выполнение дискретного логарифмирования, а следовательно, и определение числа x, для которого значение $\alpha^{x} \bmod p$ равно заданному значению y. Все абоненты размещают свои открытые ключи в общедоступном справочнике. Данный справочник должен быть заверен специально созданным доверительным центром, чтобы исключить возможные нападения путем подмены открытых ключей или навязывания ложных открытых ключей.

Если два абонента **A** и **B** хотят установить секретную связь, то они поступают следующим образом. Абонент **A** берет из справочника открытый ключ абонента **B** и, используя свой секретный ключ, вычисляет общий секретный ключ:

$$Z_{AB} = (y_B)^{x_A} = (\alpha^{x_B})^{x_A} = \alpha^{x_B x_A} \pmod p,$$

где y_A и y_B — открытые ключи абонентов **A** и **B**; x_A и x_B — соответствующие секретные ключи. Общий секретный ключ Z_{AB} нет необходимости передавать по сети связи, поскольку абонент **B** по известному из справочника открытому ключу абонента **A** аналогичным способом вычисляет значение

$$Z_{AB} = (y_A)^{x_B} = (\alpha^{x_A})^{x_B} = \alpha^{x_B x_A} \pmod p.$$

Оппоненту (потенциальному нарушителю) известны значения $y_B = \alpha^{x_B} \pmod p$ и $y_A = \alpha^{x_A} \pmod p$, но для того чтобы вычислить Z_{AB}, он должен решить трудную задачу дискретного логарифмирования. Общий секретный ключ может использо-

ваться абонентами для шифрования сеансовых секретных ключей, а последние — для шифрования сообщений с использованием симметричных методов шифрования. Решение задачи дискретного логарифмирования существует, но оно вычислительно неосуществимо. Таким образом, стойкость метода Диффи—Хеллмана основана на сложности дискретного логарифмирования.

В одноключевых криптосистемах существуют две принципиальные проблемы:

- распределение секретных ключей по защищенному каналу;
- аутентификация секретного ключа.

Под аутентификацией понимается проведение процедуры, которая позволяет получателю удостовериться, что секретный ключ принадлежит законному отправителю (например, центру распределения ключей).

Система открытого распределения ключей решает первую проблему, т. е. она позволяет обойтись без защищенного канала для распределения секретных ключей. Однако она не устраняет необходимости аутентификации. При этом надо отметить, что в двухключевой криптографии проблема аутентификации не возникает, а передвигается на первый план, так как проблема распределения ключей разрешается ее методами.

1.4.2. Понятие криптографического протокола

В криптографии часто используются термины "алгоритм" и "протокол". Интуитивно смысл этих терминов достаточно понятен. Они широко используются и в других научно-технических областях знаний. Алгоритм — одно из основных понятий в программировании и вычислительной математике, протокол — в связи. В дальнейшем изложении *под алгоритмом* мы будем понимать набор команд, действий, инструкций, вычислений, которые необходимо выполнить для того, чтобы из исходных данных получить тот или иной результат. При этом в процессе выполнения алгоритма могут возникать новые данные как результат преобразования исходных данных, либо как результат осуществления случайного выбора на каком-либо шаге алгоритма, либо как результат выполнения (вычислителем) измерений параметров окружения (внешних объектов). Алгоритм выполняется некоторым субъектом (вычислителем).

Под протоколом мы будем понимать совокупность действий (инструкций, команд, вычислений, алгоритмов), выполняемых в заданной последовательности двумя или более субъектами с целью достижения определенного результата. Корректность выполнения протокола зависит от действий каждого субъекта (пользователя, абонента) криптосистемы. В качестве субъектов могут выступать рабочая станция, программа для ЭВМ, радиопередатчик, космический спутник, оператор, сервер, орган власти и т. д. Обычно участвующие в протоколах той или иной системы субъекты действуют по предписанным алгоритмам, т. е. алгоритм

выступает как внутренний элемент протокола. Для того чтобы протокол приводил к желаемой цели, необходимо выполнение следующих условий:

- корректность протокола — совокупность действий, предусмотренных протоколом, должна обеспечить получение требуемого результата при всех возможных ситуациях;
- полнота и однозначность определения — протокол должен специфицировать действия каждого участника протокола для всех возможных ситуаций;
- непротиворечивость — результаты, полученные различными участниками протокола, не должны быть противоречивыми;
- осведомленность и согласие участников протокола — каждый субъект заранее должен знать протокол и все шаги, которые он должен выполнить; все субъекты должны быть согласны играть свою роль.

Криптографические протоколы — это такие протоколы, в которых используются криптографические преобразования данных. Хотя криптографические протоколы часто применяют те или иные алгоритмы шифрования, их целью не обязательно является секретность. Например, стороны криптографического протокола могут желать подписать одновременно какой-либо контракт, провести электронную жеребьевку, идентифицировать участников телеконференции и т. п.

Шифрование данных и вычисление односторонних функций представляют собой выполнение соответствующего алгоритма. Приведенные выше схемы аутентификации пользователя и удаленной рабочей станции, проведения электронной жеребьевки являются примерами протоколов. Если протокол использует некоторую криптографическую функцию, то она должна быть стойкой. Даже если используемые алгоритмы шифрования являются стойкими, то это не гарантирует стойкость протокола. Для того чтобы криптографический протокол был стойким, необходимо, чтобы используемые криптографические алгоритмы были стойкими именно в условиях конкретного применения.

В криптосистемах предполагается существование потенциального нарушителя (в практических приложениях это идеальное понятие вполне материализовано). Разработчики криптографических алгоритмов и протоколов предусматривают по возможности наиболее полный перечень предполагаемых действий нарушителя (или группы нарушителей) и стремятся обеспечить достижение цели протокола с учетом всех возможных атак. Атака на алгоритм, протокол или криптосистему состоит в выполнении нарушителем таких действий, с помощью которых он пытается прочитать криптограмму, взломать односторонние функции (т. е. вычислить значение аргумента по значению функции), выдать себя за другого субъекта, навязать ложные сообщения, расширить свои полномочия и, в общем случае, создать условия, при которых корректность использования алгоритмов и протоколов криптосистемы будет нарушена. Если такие действия нарушителя возможны, то говорят, что криптосистема уязвима по отношению к такой-то атаке.

По характеру своих действий можно выделить два типа нарушителей:

- пассивный нарушитель;
- активный нарушитель.

Пассивный нарушитель — это нарушитель, не предпринимающий действий по дезорганизации криптографического протокола. Его целью является перехват сообщений, циркулирующих в криптосистеме, с целью ознакомления с их содержанием, вычисления распределяемых ключей, раскрытия тайны голосования или результатов жеребьевки. Применение радиосвязи для передачи сообщений создает выгодные для пассивного нарушителя условия, в которых только косвенным путем можно установить тот факт, что криптосистема была подвергнута атаке. При проводной связи несанкционированные подключения демаскируют пассивного нарушителя. Однако необходимо учитывать, что для перехвата сообщений он может использовать побочные электромагнитные излучения и наводки.

Активный нарушитель — это нарушитель, который пытается навязать ложные сообщения, перехватить и модифицировать сообщения, получить доступ к базам данных, расширить свои полномочия, навязать ложный открытый ключ, подготовить фиктивные документы, отказаться от подписи и т. д. При использовании проводной телефонной связи создаются условия, благоприятные для активного нарушителя, тогда как при радиосвязи действия последнего могут быть легко обнаружены. Необходимо предусматривать случаи, когда активным нарушителем является легальный пользователь криптосистемы.

По характеру взаимоотношений с организацией, использующих криптосистемы (или другие средства защиты), нарушителей можно классифицировать на следующие два типа:

- внутренние нарушители;
- внешние нарушители.

Внутренним нарушителем является лицо, имеющее определенные легальные полномочия внутри организации, подвергающейся нападению с его стороны, или участник криптографического протокола, пытающийся нанести определенный ущерб другим участникам криптографического протокола. Внутренний и внешний нарушители могут быть активными или пассивными. Атака с привлечением внутреннего нарушителя называется атакой изнутри, или внутренней атакой.

Атака, в которой участвуют только *внешние* нарушители, называется внешней атакой. Возможен такой вид атак, когда внешние и внутренние нарушители объединяются, что создает наиболее серьезные угрозы безопасной эксплуатации криптосистем. Если нарушитель находится среди разработчиков, то возможны атаки на основе встроенных потайных ходов в алгоритмах формирования ключевых параметров и трудно обнаруживаемых вредоносных программных закладок.

1.4.3. Цифровая электронная подпись

Основываясь на идее использования односторонней функции с потайным ходом, описанной выше, У. Диффи и М. Е. Хеллман предложили структуру криптосистемы с открытыми ключами для сети с многими абонентами. Каждый абонент, например i-й, случайным образом выбирает некоторое значение параметра z_i и держит его в секрете. Далее он формирует алгоритм $\mathbf{E}_{z_i}$ и предоставляет его для опубликования в открытом справочнике. Он также формирует алгоритм $\mathbf{D}_{z_i}$ и держит его в секрете. Любой другой абонент, например j-й, может послать i-му абоненту секретное сообщение M. Для этого он использует открытый алгоритм шифрования $\mathbf{E}_{z_i}$ и вычисляет криптограмму $C = f_{z_i}(M)$, которую посылает i-му абоненту. Используя секретный алгоритм $\mathbf{D}_{z_i}$, абонент i вычисляет исходный открытый текст: $f_{z_i}^{-1}(C) = M$.

Авторы этой обобщенной схемы открытого шифрования показали возможность ее использования для получения *цифровых подписей*. В общем случае, цифровая подпись представляет собой некоторое число со специфической структурой, которое допускает проверку с помощью открытого ключа того факта, что оно было выработано для некоторого сообщения с использованием секретного ключа. Для реализации цифровой подписи необходимо выбрать такую одностороннюю функцию с потайным ходом (с секретом) f_z, для которой при всех значениях параметра z область определения функции f_z совпадает с областью ее значений. При этом условии для любого сообщения, которое может быть представлено в виде числа из области определения функции $f_z(x)$, абонент i может сформировать с помощью секретного алгоритма число $S = f_{z_i}^{-1}(M)$. Если сообщение слишком велико, то его можно разбить на части необходимого размера и каждую из них подписать независимо.

Каждый пользователь криптосистемы может по значению S восстановить сообщение M, используя открытый алгоритм шифрования $\mathbf{E}_{z_i}$. Если M представляет собой осмысленное сообщение или может быть сопоставлено с таковым по некоторому заранее оговоренному правилу, то значение S может рассматриваться как цифровая подпись абонента i под сообщением M. Действительно, только владелец секретного алгоритма $\mathbf{D}_{z_i}$ мог выработать "открытый" текст S, который с помощью алгоритма $\mathbf{E}_{z_i}$ зашифровывается в осмысленную криптограмму M, поскольку лишь абонент i знает способ вычисления $f_{z_i}^{-1}$.

Абонент i может послать абоненту j также секретное сообщение с подписью. Для этого он зашифровывает S с помощью открытого алгоритма $\mathbf{E}_{z_j}$, получая криптограмму $C = \mathbf{E}_{z_i}(S)$. Получив зашифрованное сообщение, j-й абонент расшифровывает его своим секретным алгоритмом $\mathbf{D}_{z_j}(C) = S$, затем число S зашифровывает открытым алгоритмом i-го абонента $\mathbf{E}_{z_i}(S) = M$. Таким образом, по полученной

криптограмме C абонент j восстанавливает подпись абонента i и исходное сообщение.

Использование протоколов на основе симметричных криптографических методов предполагает, что две стороны *доверяют друг другу*. Криптосистемы с открытым ключом (асимметричные криптосистемы) позволяют реализовать протоколы взаимодействия сторон, которые *не доверяют друг другу*. Системы электронной цифровой подписи являются одним из важнейших примеров такого типа. Для того чтобы цифровая подпись могла бы эффективно использоваться в реальных деловых взаимоотношениях абонентов, следует придать ей юридическую силу. Для этого необходимо принять соответствующие правовые законы на государственном (или международном) уровне, а обмен открытыми ключами подкреплять обычной юридической процедурой, чтобы обеспечить защиту от отказов от открытых ключей.

1.4.4. Криптосистема RSA

Система RSA является наиболее широко известной системой электронной цифровой подписи. При этом она и наиболее проста для понимания. Она изобретена Р. Ривестом (R. Rivest), А. Шамиром (A. Shamir) и Л. Адлеманом (L. Adleman) [116]. Рассмотрим эту криптосистему. Согласно известной из теории чисел теореме Эйлера, для любых взаимно простых целых чисел M и n, где $M < n$, выполняется соотношение $M^{\varphi(n)} = 1(\mathrm{mod}\ n)$.

В качестве числа M будем брать исходное сообщение, которое необходимо подписать или зашифровать. Условие взаимной простоты чисел M и n обеспечим тем, что будем выбирать число n, равное произведению двух больших простых множителей. В этом случае вероятность того, что случайное сообщение не будет взаимно простым с модулем, является пренебрежимо малым. В качестве одностороннего преобразования будем использовать возведение в степень по модулю простого числа. При некотором значении этой степени e будем иметь функцию шифрования $\mathbf{E}$, которая преобразует исходное сообщение M в криптограмму $C = \mathbf{E}(M) = M^e\ (\mathrm{mod}\ n)$.

Параметр e будем полагать общедоступным (открытым, публичным). По известному значению S при известных n и e вычислительно сложно найти M. В качестве потайного хода соответствующей односторонней функции зашифрования $M^e\ (\mathrm{mod}\ n)$ будем использовать также возведение в степень, но с другим значением степени. Новое значение степени d необходимо выбрать таким, чтобы функция расшифрования $\mathbf{D}(C) = C^d(\mathrm{mod}\ n)$ была обратной по отношению к $\mathbf{E}(M) = M^e(\mathrm{mod}\ n)$, т. е. должно выполняться условие $M = \mathbf{D}[\mathbf{E}(M)] = (M^e)^d = M^{ed}(\mathrm{mod}\ n)$.

Из этого соотношения следует: $ed = 1(\mathrm{mod}\ \varphi(n))$. Таким образом, две процедуры возведения в степень по модулю n будут взаимно обратными, если произведение

степеней равно единице по модулю функции Эйлера от числа n. Параметр d является ключом к потайному ходу, поэтому он считается секретным. Теперь задача состоит в выборе необходимых значений степеней e и d. Очевидно, что первоначально нужно установить значение функции Эйлера от модуля n. Легко заметить, что для любого простого числа p имеем $\varphi(p) = p - 1$. Поскольку мы выбрали $n = pq$, где оба множителя являются простыми числами, то, используя свойство мультипликативности функции Эйлера, получаем:

$$\varphi(n) = \varphi(pq) = \varphi(p)\varphi(q) = (p - 1)(q - 1).$$

Еще Евклиду было известно, что если целые числа e и m удовлетворяют условиям $0 < e < m$ и НОД$(m, e) = 1$, то существует единственное d, удовлетворяющее условиям $0 < d < m$ и $de = 1 \pmod m$, и, кроме того, d может быть вычислено с помощью "расширенного" алгоритма Евклида.

Приходим к следующей схеме функционирования криптосистемы RSA.

1. Каждый пользователь выбирает два больших не равных между собой простых числа p и q, находит их произведение $n = pq$ и вычисляет значение $\varphi(n) = (p - 1)(q - 1)$.

   Одним из требований к выбору p и q является то, что, по крайней мере, одно из чисел $(p - 1)$ или $(q - 1)$ должно иметь один большой простой множитель. Размер модуля n должен быть не менее 512 бит. Для ответственных применений системы RSA рекомендуемая длина модуля составляет 1024 бит.

2. Затем выбирается целое число e такое, что $e < \varphi(n)$ и НОД$(e, \varphi(n)) = 1$, и вычисляется d, удовлетворяющее условию $ed = 1 \pmod{\varphi(n)}$.

3. *Секретным ключом* является тройка чисел p, q, d, которая держится в секрете.

   На самом деле хранить в секрете достаточно только число d, поскольку простые числа p и q нужны лишь на этапе формирования модуля n и вычисления d. После этого числа p и q могут быть уничтожены.

4. Пара чисел n, e является *открытым ключом*, который предоставляется всем абонентам криптосистемы RSA.

5. *Процедура подписывания* сообщения M — это возведение числа M в степень d по mod n, а именно $S = M^d \pmod n$.

   Число S есть *цифровая подпись*, которую может выработать только владелец секретного ключа.

6. *Процедура проверки* подписи S, соответствующей сообщению M, — это возведение числа S в целую степень e по mod n, а именно $M' = S^e \pmod n$.

   Если $M' = M$, то сообщение M признается подписанным пользователем, который предоставил ранее открытый ключ e. Видно, что

$$S^e = (M^d)^e = M^{de} = M^{Q\varphi(n)+1} = M^{Q\varphi(n)}M = (M^{\varphi(n)})^Q M = 1^Q M \pmod n,$$

т. е. составить криптограмму, соответствующую данному открытому ключу и данному сообщению, можно только по известному секретному ключу d.

Стойкость криптосистемы RSA основана на сложности разложения модуля на два больших простых множителя. Если задачу такого разложения удалось бы решить, то тогда можно было бы легко вычислить функцию Эйлера от модуля и затем, используя расширенный алгоритм Евклида, определить секретный ключ по открытому.

До настоящего времени не найдены практически реализуемые общие способы решения этой задачи при длине модуля 512 бит и более. Однако для частных видов простых чисел p и q сложность этой задачи резко понижается, поэтому в криптосистеме RSA необходимо выполнить ряд специальных тестов при генерации секретного ключа. Другой особенностью системы RSA является ее мультипликативность: $\mathbf{E}(M_1, M_2) = \mathbf{E}(M_1)\mathbf{E}(M_2)(\mathrm{mod}\ n)$, что позволяет нарушителю на основе двух подписанных сообщений сформировать подпись третьего сообщения, которое равно $M_3 = M_1M_2\ (\mathrm{mod}\ n)$. Поскольку M_3 в подавляющем большинстве случаев не будет представлять собой какого-либо осмысленного текста, то наличие этой особенности не является недостатком. В системе RSA необходимо учитывать также следующую возможность. Выбрав произвольное значение S, можно вычислить значение $M' = S^e$, т. е. произвольное значение можно представить как подпись некоторого сообщения. Такие фиктивные сообщения, естественно, являются случайными. Однако в ряде применений имеют место случаи, когда требуется подписывать случайные сообщения. В таких случаях используют следующую схему.

1. К сообщению T, которое необходимо подписать и передать по открытому каналу, присоединяется заранее условленный двоичный вектор V длиной $v = 64$ бит:

   $M \to T \parallel V$.

2. Вырабатывается подпись для сообщения M:

   $S = M^d\ (\mathrm{mod}\ n)$.

3. Значение S направляется приемной стороне.
4. Приемная сторона по значению S вычисляет значения

   $M' = S^e\ (\mathrm{mod}\ n)$, $V' = M'(\mathrm{mod}\ 2^v)$ и $T' = M'\mathrm{div}\ 2^v$.

5. Если V' равно условленному заранее значению V, т. е. если выполняется условие $V' = V$, то принимается решение, что сообщение T' подписано владельцем секретного ключа, соответствующего открытому ключу, с помощью которого выполнялась проверка подписи. (Вероятность того, что случайное сообщение может быть принято за подписанное, равна 2^{-v}.)

Полезным свойством рассматриваемой системы открытого шифрования является то, что при зашифровании сообщения двумя и более пользователями процедуры

расшифрования можно выполнить в произвольном порядке. Например, пусть $C = \mathbf{E}_1[\mathbf{E}_2(M)]$, тогда $\mathbf{D}_1[\mathbf{D}_2(C)] = \mathbf{D}_2[\mathbf{D}_1(C)] = M$. Это свойство используется в протоколе слепой подписи и в системах тайного электронного голосования.

Таким образом, секретный ключ служит для подписывания сообщений, открытый ключ — для проверки подписи. Для того чтобы послать секретное сообщение некоторому абоненту **A**, любой пользователь может воспользоваться открытым ключом абонента **A** и сформировать криптограмму $C = \mathbf{E}_A(M)$. По значению C восстановить сообщение M может только абонент **A**, который знает секретный ключ, соответствующий открытому ключу, с помощью которого была сформирована криптограмма. В криптосистеме RSA генерация подписи совпадает с процедурой расшифрования, а проверка подписи — с процедурой зашифрования.

Скорость шифрования, обеспечиваемая двухключевыми (асимметричными) шифрами, на несколько порядков ниже скорости, которой обладают одноключевые (симметричные) криптосистемы. Поэтому наиболее эффективны гибридные криптосистемы, в которых информация шифруется с помощью одноключевых шифров, а распределение сеансовых ключей осуществляется по открытому каналу с помощью двухключевых шифров. Например, используя криптосистему RSA, можно легко обменяться сеансовым ключом с любым абонентом, зашифровав сеансовый ключ с помощью его открытого ключа. Зашифрованный сеансовый ключ можно безопасно передать по открытому каналу связи, поскольку необходимым для расшифрования секретным ключом обладает только абонент, открытый ключ которого был использован для зашифрования. Для непосредственного засекречивания информации двухключевые шифры находят ограниченное применение.

1.4.5. Цифровая подпись Эль-Гамаля

Рассмотрим систему цифровой подписи [65], носящую имя ее изобретателя Эль-Гамаля и основанную на схеме формирования открытых и секретных ключей, которая используется в методе Диффи—Хеллмана. Пусть мы имеем большое простое число p, такое, что разложение числа $p - 1$ содержит, по крайней мере, один большой простой множитель и первообразный корень α по модулю p.

Механизм подписывания состоит в следующем. Некоторый абонент **A** выбирает секретный ключ x_A, по которому формирует открытый ключ $y_A = \alpha^{x_A}$. Подписью абонента **A** под документом M (подписываемое сообщение должно иметь длину меньше простого модуля p, т. е. $M < p$) служит пара чисел (r, s) (где $0 \le r < p - 1$ и $0 \le s < p - 1$), которая удовлетворяет уравнению $(\alpha^M) = y_A^r r^s \pmod p$.

Это уравнение служит для проверки того факта, что документ подписан абонентом **A**. (Значение $y_A = \alpha^{x_A}$ является открытым ключом абонента **A** и доступно для всех пользователей, что предоставляет каждому возможность убедиться в том, что данное сообщение действительно подписано абонентом **A**.)

Данная система электронной цифровой подписи основана на том, что только действительный владелец секретного ключа x_A может выработать пару чисел (r, s), удовлетворяющую уравнению проверки подписи. Используя значение x_A, абонент **A** вырабатывает подпись по следующему алгоритму:

1. Сгенерировать *случайное* число k, удовлетворяющее условию:

   $0 < k < p - 1$ и НОД(k, $p - 1$) = 1.
2. Вычислить $r = \alpha^k \pmod p$.
3. Вычислить s из уравнения $M = x_A r + ks \pmod{(p - 1)}$.

Из теории чисел известно, что последнее уравнение имеет решение для s, если НОД(k, $p - 1$) = 1. Это уравнение легко получить путем подстановки в уравнение проверки подписи значения $r = \alpha^k \pmod p$, а именно:

$$\alpha^M = \alpha^{x_A r} \alpha^{ks} = y_A^r r^s \pmod p.$$

Из двух последних формул видно, что владелец секретного ключа в состоянии подписать документ, а его подпись может быть проверена по его открытому ключу. Нахождение пары чисел (r, s) без знания секретного ключа вычислительно сложно. Различных подписей, соответствующих данному документу, может быть чрезвычайно много (заметим, что k может иметь разные значения), но выработать правильную подпись способен только владелец секретного ключа. Множество возможных подписей отличаются значением r, но для данного r найти соответствующее значение s без знания секретного ключа практически невозможно. Для вычисления секретного ключа по открытому ключу необходимо решить задачу дискретного логарифмирования, которая является вычислительно сложной.

Особенностью системы цифровой подписи Эль-Гамаля является генерация случайного числа k. В этой криптосистеме одно и то же значение k не допускается использовать для формирования подписи к двум разным сообщениям. Это связано с тем, что на основе двух разных подписей, сформированных при одном и том же значении k, имеется возможность вычислить секретный ключ. Кроме того, использованные при выработке значения k подлежат гарантированному уничтожению. Если нарушитель сможет получить значение k, то он также сможет вычислить секретный ключ. В реально используемых системах реализуется случайная генерация числа k большого размера и некоторый механизм его уничтожения после выработки подписи. При программной реализации обеспечивается такая схема формирования подписи, что число k появляется только в регистрах микропроцессора и оперативной памяти, а механизм уничтожения состоит в записи нового случайного значения в ячейки памяти, которые до этого содержали значение k.

Выше мы рассмотрели механизмы двухключевой криптографии, которые дают возможность подписывать сообщения ограниченной длины (порядка 10^3 бит). Если сообщение имеет большой размер, то при прямолинейном использовании

таких механизмов потребуется разбить исходное сообщение на большое число блоков меньшего размера и выработать столько подписей, сколько блоков будет содержаться в сообщении. Это сильно усложняет проблему хранения подписей и самих подписанных сообщений, а также организации баз данных, содержащих большое число подписанных документов. Для упрощения этой проблемы подписывается не сам документ, а некоторый его цифровой образ небольшого размера, полученный по специальным криптографическим процедурам, называемым *хэшированием*.

Алгоритм хэширования должен быть таким, чтобы обеспечить вычислительную неосуществимость нахождения двух сообщений с одинаковым значением цифрового образа (значением хэш-функции от сообщения). В настоящее время разработаны алгоритмы, которые удовлетворяют этому условию и позволяют легко вычислить значение хэш-функции от данного документа. Вместо подписывания большого числа отдельных частей электронного документа в реальных системах цифровой подписи осуществляется вычисление хэш-функции от документа и подписывание хэш-функции. Если хэш-функция подписана, то документ считается подписанным.

1.4.6. Слепая подпись Чаума

Понятие *слепой подписи* впервые было введено Д. Чаумом (D. Chaum), который также предложил первые варианты ее реализации [54—56]. Под слепой подписью понимается двухключевая криптосистема, которая позволяет осуществить подписывание электронных сообщений таким образом, чтобы подписывающая сторона не имела доступа к информации, содержащейся в подписываемом сообщении. Такое надуманное и нелепое на первый взгляд требование в ряде криптографических протоколов имеет очень большое значение. Слепая подпись используется, например, в системах тайного электронного голосования и электронных деньгах, т. е. в таких криптографических протоколах, где требуется обеспечить решение задачи неотслеживаемости. Сама процедура слепой подписи требует от одного из участников согласия на то, что он может подвергнуться некоторым посягательствам путем приписывания ему определенных обязательств, которые он не хотел бы брать на себя.

В реальных протоколах, где используется слепая подпись, предусматриваются также процедуры, которые дают определенные гарантии подписывающей стороне, что ее не обманут. Эти гарантии основаны на некоторых дополнительных процедурах и соглашениях, вносящих ограничения и ответственность на сторону, представляющую документ для слепого подписывания. Естественно, что каждая из сторон должна согласиться на тот или иной риск и на определенные гарантии, в противном случае нельзя было бы решить ту задачу, которую хотят решить обе стороны. Так, при использовании систем ЭЦП, пользователи принимают риск, что кто-то может вычислить их секретный ключ. Гарантией защиты их интересов является высокая сложность задачи вычисления секретного ключа. Другим ти-

пом гарантий, используемых в криптографических протоколах, служит низкая вероятность повторения случайных чисел, если они являются достаточно длинными. Используя протокол слепой подписи, можно решить некоторые важные для практики проблемы (такие как, например, построение систем тайного электронного голосования, электронных денег).

Рассмотрим протокол слепой подписи Чаума, основанный на уже известной читателю криптосистеме RSA. Пусть субъект **A** желает подписать некоторое сообщение M у субъекта **B**. Для этого необходимо осуществить следующие шаги:

1. Пользователь **A** (субъекты являются пользователями данной криптосистемы) генерирует случайное простое число k, такое что НОД(k, N) = 1, где N — часть открытого ключа пользователя **B**, т. е. модуль, по которому осуществляются вычисления.
2. Далее он вычисляет значение $M' = k^e M \pmod N$, которое представляет для подписывания. Подписывающий не может получить доступ к сообщению M, поскольку оно зашифровано путем наложения на него "разового" ключа k^e и использованием операции модульного умножения.
3. Пользователь **B** подписывает сообщение M' в соответствии с процедурой подписывания в криптосистеме RSA: $S' = (k^e M)^d = kM^d \pmod N$.

   Сформировав подпись S', подписывающий также не имеет возможности получить доступ к значению M^d, поскольку оно зашифровано путем наложения на него "разового" ключа k. Если бы подписывающий смог узнать значение M^d, то он тогда смог бы легко вычислить M: $(M^d)^e = M \pmod N$. Это означает, что, получив значение $M^d \pmod N$ — это цель протокола слепой подписи, пользователь **A** должен держать его в секрете от подписавшего.
4. Теперь, используя расширенный алгоритм Евклида, пользователь **B** вычисляет для числа k мультипликативно обратный элемент (k^{-1}) в поле вычетов по модулю N и восстанавливает подпись для сообщения M, а именно: $S' = k^{-1}S' = k^{-1}kM^d = M^d \pmod N$.

Таким образом, цель достигнута — пользователь **A** сформировал правильную подпись пользователя **B**, соответствующую сообщению M, причем он уверен, что подписывающий не знает содержания сообщения M.

1.4.7. Виды нападений на цифровую подпись

В системе цифровой подписи используются три криптографических алгоритма:

- алгоритм формирования подписи по секретному ключу;
- алгоритм проверки подписи по открытому ключу;
- алгоритм вычисления хэш-функции от подписываемого сообщения.

К математическим основам цифровой подписи можно отнести также алгоритмы формирования секретного и открытого ключей. Функционирование реальных

систем требует также правового, организационного и программно-технического обеспечения. Правовое обеспечение включает принятие правовых законов, придающих юридическую силу цифровой подписи. Организационное обеспечение включает регистрацию пользователей в некотором доверительном центре, оформление документов между пользователем и доверительным центром (или между двумя пользователями) об ответственности за переданные друг другу открытые ключи. К программно-техническому обеспечению относится вся совокупность программных и аппаратных средств, позволяющих осуществить сложные вычисления и обеспечивающих сохранность базы данных, содержащей подписанные документы и образцы подписей к ним.

Возможные виды нападений на цифровую подпись можно классифицировать на несколько групп:

- нападения на криптографические алгоритмы;
- нападения, связанные с нарушением протокола;
- нападения, основанные на нарушении целостности механизмов системы цифровой подписи.

Нападающим может оказаться внешний субъект, а также подписывающая сторона (отказ от подписи) или сторона, проверяющая подпись (навязывание ложной подписи).

Нападения на криптографические алгоритмы связаны с решением сложных математических задач, например, дискретного логарифмирования по модулю большого простого числа. Шансы на успех у нападающего крайне низки. Такой вид нападений может быть предпринят против двухключевого криптографического алгоритма или против хэш-функции. В первом случае подделывается подпись, во втором — документ. К нападениям, связанным с нарушением протокола, относятся, например, повторения подписанных сообщений, задерживание сообщений. Чтобы предотвратить такие действия, в документе предусматриваются специальные поля, в которых указывается дата и номер документа. Необходимо использовать также механизмы, обеспечивающие защиту от фактов отказа о получении сообщений.

Атаки, основанные на нарушении целостности механизмов системы цифровой подписи, являются наиболее многообразными. К ним относятся удаление из базы данных подписанного сообщения, перехват секретного ключа программными или аппаратными средствами, навязывание ложного открытого ключа, подмена открытого ключа в базе данных. Эти примеры показывают, что широкий спектр нападений связан с несанкционированным доступом к данным, циркулирующим в системе цифровой подписи. Безопасное функционирование системы цифровой подписи требует использования защищенного окружения.

Нападения на криптосистему могут основываться также на навязывании пользователям системы цифровой подписи или средств защиты от несанкционированного доступа с замаскированными слабостями, а также навязывании другого сис-

темного и прикладного программного обеспечения с встроенными недокументированными закладками. Чтобы не допустить этого, криптографические средства и средства защиты от несанкционированного доступа подлежат сертификированию специальными организациями.

1.5. Вероятностные шифры

Одним из перспективных способов повышения стойкости известных шифров является задание неопределенности хода шифрования информации. Эта идея может быть реализована путем введения в преобразуемое сообщение случайных данных. Если в механизме шифрования используются операции или процедуры, зависящие от преобразуемых данных (как это имеет место в шифре RC5), тогда сами операции будут изменяться случайно. Идея введения элементов случайности в процедуру шифрования преследует цель затруднить использование общего принципа криптоанализа блочных шифров, основанного на попытках выявления статистических свойств алгоритма шифрования, например, путем подбора специальных исходных текстов или криптограмм.

1.5.1. Гомофонические шифры

Криптограммы, полученные методом одноалфавитной или многоалфавитной подстановки, легко раскрываются частотным криптоанализом. Чтобы скрыть частотные свойства источника сообщений и тем самым затруднить криптоанализ может быть применен метод *гомофонического* (монофонического) шифрования, который заключается в выравнивании частотности знаков криптограммы, т. е. использовании такого механизма криптографических преобразований, который бы вырабатывал криптограммы с одинаковой частотой появления каждого знака, используемого для записи шифртекста. Наиболее простым механизмом гомофонического шифрования является следующий способ. Пусть дан источник сообщений с известными статистическими свойствами. Выразим частоты появления каждой буквы исходного алфавита целым числом f_i, где i — номер буквы в алфавите: $f_1, f_2, ..., f_L$, где L — число букв в исходном алфавите. Каждой букве T_i, где $i = 1, 2, \ldots, L$, исходного алфавита поставим в соответствие подмножество Ψ_i знаков выходного алфавита (т. е. алфавита, с помощью которого записывается криптограмма). Причем зададим эти множества в соответствии с двумя условиями:

- никакая пара подмножеств не содержит одинаковых элементов;
- количество различных знаков в подмножестве Ψ_i равно f_i.

Шифрование будем осуществлять путем замены каждой буквы исходного текста T_i на случайно выбираемый знак из подмножества Ψ_i. В этом случае при многократной замене заданной буквы T_i исходного текста знаками подмножества Ψ_i каждый знак выходного алфавита будет использоваться в среднем одинаковое

число раз. Это число обратно пропорционально числу элементов в подмножестве Ψ_i, т. е. пропорционально $1/f_i$. Частота обращения к подмножеству Ψ_i равна частоте появления буквы T_i в исходном тексте, т. е. пропорциональна f_i. Из этих соотношений приходим к выводу, что в среднем частоты появления в криптограмме всех знаков выходного алфавита равны. Расшифрование не представляет трудностей — по знаку криптограммы определяем подмножество, к которому он относится, а по конкретному подмножеству — соответствующую ему букву исходного алфавита. Описанный способ шифрования требует использования $f_1 + f_2 + ... + f_L$ знаков в выходном алфавите. Наиболее существенным элементом в этом способе шифрования является то, что в преобразования вводится вероятностный процесс — случайный выбор элемента из заданного подмножества.

Рассмотренный способ в настоящее время не представляет большого интереса для практических приложений, однако основная идея введения случайности в процесс шифрования может быть использована для построения современных вероятностных блочных шифров.

1.5.2. Шифры с простым вероятностным механизмом

В предыдущих разделах был рассмотрен ряд шифров, в которых используются операции преобразования, зависящие от преобразуемых данных. Такие операции не являются заранее предопределенными и изменяются от одного входного блока к другому. Если такие шифрующие механизмы применяются для преобразования случайных данных, то операции будут изменяться случайным образом. "Подмешивание" случайных данных к шифруемому сообщению позволяет задать вероятностный характер операций преобразования и тем самым повысить вычислительную стойкость криптосистемы [24]. Пусть **E** есть b-битовая функция шифрования, P есть p-битовый блок открытого текста и R — r-битовый случайный блок, где $b = r + p$. Подадим на вход шифрующей функции блок $B = R \parallel P$, где знак "||" обозначает конкатенацию двоичных векторов R и P: $P \to B = R \parallel P \to C = \mathbf{E}(B, K)$, где K — ключ шифрования. Поскольку при шифровании размер входного блока увеличивается, то такое шифрование отображает данный блок открытого текста P на большое множество блоков шифртекста $\{C_1, C_2, \ldots, C_n\}$, где $n = 2^r$. Общая схема вероятностного шифра с простым механизмом присоединения случайных данных показана на рис. 1.11. Датчик случайных чисел (ДСЧ) является внутренней частью шифратора, также как и алгоритм шифрования, реализующий функцию шифрования **E**. Предполагается, что он расположен в защищенной части шифрующей аппаратуры и не может быть подменен злоумышленником (т. е. злоумышленник не имеет доступа к значению R). Это предположение является приемлемым, поскольку шифраторы проектируются таким образом, чтобы обеспечить защиту от подмены алгоритма шифрования так же, как и от считывания и копирования ключа. При дешифровании блока шифртекста законный пользователь, владеющий секретным ключом, восстанав-

ливает блок $B = R \| P$, после чего значение R отбрасывается и выделяется исходное сообщение P.

Рис. 1.11. Базовая схема вероятностного шифрования

Выбирая различные значения отношения b/p, можно регулировать степень усиления шифрования. Чем больше указанное отношение, тем больше усиление. Отличие вероятностного шифрования от криптосхемы с частой сменой сеансовых ключей состоит в том, что она не приводит к существенному снижению скорости шифрования в случае использования шифров с этапом предвычислений, на котором формируется ключ шифрования под управлением сеансового ключа. Схема вероятностного шифрования позволяет регулировать степень уменьшения скорости преобразования. Если функция **E** имеет исходное значение скорости преобразования s_0, то при использовании вероятностного шифрования она составляет $s = s_0(b - r)/b$.

Известен ряд успешных атак на криптосистемы DES, RC5, Blowfish при использовании пониженного числа раундов шифрования. Очевидно, что можно выбрать такую длину случайного блока R, чтобы сделать редуцированные версии этих шифров стойкими к известным атакам. Для этого можно выбрать значение $r = b - 1$. Шифры с простым вероятностным механизмом предоставляют следующие преимущества:

- стойкость известных блочных шифров может быть существенно повышена;
- в определенном смысле можно управлять стойкостью шифра путем выбора различных значений отношения r/b;
- вероятностная криптосхема позволяет использовать новые механизмы задания зависимости процедур шифрования от секретного ключа.

Плата за эти достоинства состоит в следующих недостатках:

- скорость уменьшается в r/b раз;
- блоки шифртекста имеют длину больше, чем блоки исходного текста.

Последний недостаток накладывает существенные ограничения на применение вероятностных шифров в компьютерных системах. Для компенсации эффекта

расширения можно использовать предварительное сжатие исходного сообщения. Этот способ в ряде случаев позволяет разработать вероятностные шифры, для которых длины шифртекста равны длине входного сообщения.

Замечание

Интересно, что сжатие данных перед их шифрованием существенно повышает стойкость шифрования к атакам на основе шифртекста. Однако по отношению к атакам на основе известных исходных текстов или подобранных текстов предварительное сжатие информации не повышает стойкости шифрования, поскольку в соответствии с принципом Керкхоффа мы должны полагать, что криптоаналитик знает используемый нами алгоритм сжатия.

1.5.3. Вероятностное объединение битов данных и случайных битов

Рассмотренный выше простой механизм вероятностного шифрования, основанный на формировании шифруемого блока данных путем объединения случайных и информационных битов, может быть применен для повышения стойкости шифрования при использовании многих известных блочных криптоалгоритмов. Причем по отношению ко многим видам нападений эта задача решается при сравнительно небольшом отношении числа случайных битов к информационным. Однако для некоторых известных шифров, имеющих слабости к дифференциальному (ДКА) или линейному (ЛКА) криптоанализу, усиление на основе данного способа вероятностного шифрования, включая вариант объединения случайных и информационных битов в зависимости от секретного ключа, требует существенного увеличения доли случайных битов до 80% и более. Это приводит к заметному снижению эффективной скорости шифрования и серьезному увеличению размера шифртекста.

В настоящем разделе рассматриваются варианты повышения эффективности указанного способа вероятностного шифрования при малой доле случайных битов и использовании шифрующих процедур с хорошими рассеивающими свойствами, но, возможно, обладающих непредвиденными слабостями к ДКА и ЛКА. Данные варианты могут быть также применены пользователями для защиты от предполагаемых нападений с использованием потайных лазеек в криптоалгоритмах.

Схема вероятностного шифрования по патенту № 2103829 показана на рис. 1.11, где датчик случайных чисел (ДСЧ) рассматривается как внутренняя часть шифратора, предполагаемая недоступной нападающему. Пусть $\mathbf{E}$ есть b-битовая функция шифрования, T есть t-битовый блок открытого текста и R — r-битовый случайный блок, где $t < b$ и $r = b - t$. На вход шифрующей функции $\mathbf{E}$ поступает блок данных $B = R \parallel T$. Последовательность исходного текста T можно представить следующей записью:

$$T \rightarrow B = R \parallel T \rightarrow C = \mathbf{E}(B, K),$$

где K — ключ шифрования. Поскольку при шифровании размер входного блока увеличивается, то такое шифрование отображает заданный текст T на большое множество блоков шифртекста $\{C_1, C_2, \ldots, C_n\}$, где $n = 2^r$. При расшифровании блока шифртекста законный пользователь, владеющий секретным ключом, восстанавливает блок $B = R \parallel T$, после чего значение R отбрасывается и выделяется T. Выбирая различные значения отношения b/t можно регулировать степень достигаемого усиления шифрования. Чем больше указанное отношение, тем больше усиление.

Очевидно, что скорость шифрования уменьшается в $b/t = 1+r/t$ раз, и во столько же раз увеличивается размер шифртекста С. Рассмотрим несколько вариантов повышения эффективности вероятностного шифрования.

В первом варианте используется недетерминированное перемешивание случайных и информационных битов, благодаря чему обеспечивается уменьшение отношения r/t при существенном повышении стойкости. Для реализации данной идеи в случайном двоичном векторе выделяются две части с заранее условленной длиной: $R = R_1 \parallel R_2$. Затем до выполнения шифрующего преобразования двоичного вектора $R_2 \parallel T$ над ним выполняют перестановку битов в зависимости от случайного значения R_1, что задает случайное перемешивание битов сообщения T и битов случайного значения R_2. Для перемешивания битов могут быть использованы управляемые операционные блоки перестановок **P**, примененные ранее в качестве базового криптографического примитива для построения стойких скоростных шифров [3, 16]. Выполняемая блоком **P** перестановка зависит от значения управляющего вектора V, который формируется в зависимости от R_1. Последовательность преобразований в варианте со случайным объединением информационных и случайных битов (рис. 1.12) имеет следующий вид:

$$T \to R_2 \parallel T \to \mathbf{P}_V(R_2 \parallel T) \to R_1 \parallel \mathbf{P}_V(R_2 \parallel T) \to \mathbf{E}_K(R_1 \parallel \mathbf{P}_V(R_2 \parallel T)).$$

Рис. 1.12. Схема с вероятностным перемешиванием случайных и информационных битов

В типичных **P**-блоках длина управляющего вектора V (v) в два и более раза превышает длину преобразуемого вектора $R_2 \parallel T(r_2 + t)$. В рассматриваемом случае

предполагается, что имеет место условие $r_1 < r_2 + t < v$, поэтому управляющий вектор может быть сформирован, например, в виде многократного повторения вектора R_1 ($V = R_1 \parallel ... \parallel R_1 \parallel R_1$) или поочередного повторения R_1 и фрагмента K_1 секретного ключа ($V = R_1 \parallel K_1 \parallel R_1 \parallel K_1$). В последнем случае перемешивание битов R_2 и T осуществляется в зависимости от секретного ключа вероятностным образом. Повышение стойкости к ДКА и ЛКА связано со случайным распределением информационных битов по разрядам шифруемого блока данных. Например, при ДКА на основе специально подобранных исходных текстов вероятность получить два блока данных с заданной разностью является существенно малой при $r_1, r_2 = 8$. При $b = 64$ и 128 это соответствует достаточно малой доле случайных битов (25% и 12% соответственно).

Второй вариант усиления упрощенной схемы вероятностного шифрования связан с идеей предварительного шифрования исходного текста T по случайно генерируемому значению R, используемому в качестве разового ключа предварительного шифрования (рис. 1.13). Последовательность преобразований имеет вид:

$$T \to \mathbf{E}'_R(T) \to \mathbf{E}''_K(R \parallel \mathbf{E}'_R(T)).$$

Эффект усиления достигается за счет дополнительных преобразований по ключу разового использования, вероятность повторения которого составляет порядка 2^{-r} при атаках на основе подобранных значений T и C (благодаря хорошим рассеивающим свойствам шифрующих процедур $\mathbf{E}''$). При выполнении предварительного шифрования может быть применена исходная схема вероятностного шифрования, что приводит к следующей последовательности преобразований (рис. 1.14):

$$T \to R_2 \parallel T \to \mathbf{E}'_{R1}(R_2 \parallel T) \to \mathbf{E}''_K(R_1 \parallel \mathbf{E}'_{R1}(R_2 \parallel T)).$$

Рис. 1.13. Схема с предварительным шифрованием данных по случайному вектору

Рис. 1.14. Двухступенчатое вероятностное шифрование

Этот случай относится к третьему варианту усиления и является обобщением первого варианта, в котором перемешивание случайных и информационных битов может быть рассмотрено как частный случай шифрующего преобразования.

Наиболее экономичным при аппаратной реализации является первый вариант, а при программной реализации — второй и третий. С точки зрения повышения стойкости предпочтительным является третий вариант. В общем плане эффект усиления в рассмотренных вариантах связан с тем, что в соотношения, связывающие пары значений T и C, входит случайная (псевдослучайная) величина R при атаке на основе подобранных исходных сообщений T (подобранных шифртекстов C).

Рассмотренные способы вероятностного шифрования представляются весьма эффективными для страхования от непредвиденных слабостей используемого алгоритма шифрования и от встроенных потайных "дверей". Эффект расширения блока шифртекста накладывает существенные ограничения на применение вероятностных шифров в компьютерных системах. Для компенсации эффекта расширения можно использовать предварительное сжатие исходного сообщения. Этот способ в ряде случаев позволяет разработать вероятностные шифры, для которых длина шифртекста равна длине входного сообщения. При этом сжатие данных перед их зашифрованием существенно повышает стойкость шифрования. Для многих применений в телекоммуникационных системах рассмотренный вариант вероятностного шифрования может быть использован без существенных ограничений.

1.5.4. Вероятностные механизмы в двухключевых шифрах

В двухключевых криптосистемах шифрование выполняется по общеизвестному алгоритму шифрования $\mathbf{E}_z$ (открытому ключу), поэтому принципиально возможна следующая атака. При наличии шифртекста $C = \mathbf{E}_z(T)$, для того чтобы найти исходный текст T, криптоаналитик может выбирать случайно различные варианты возможных открытых текстов T_i' и вычислять соответствующие им криптограммы $C_1' = \mathbf{E}_z(T_1')$, $C_2 = \mathbf{E}_z(T_2)$, …, $C_m' = \mathbf{E}_z(T_m')$. Если он угадает истинный исходный текст, то этот факт он установит по равенству $C' = C$. В такой схеме криптоанализа каждая неудачная попытка дает криптоаналитику определенную информацию, поскольку она исключает количество оставшихся вариантов возможных исходных текстов.

Естественно, что вероятность выбора правильного исходного текста крайне низка для обычно используемых размеров входного сообщения, поэтому такая атака на практике не может привести к успеху с существенной вероятностью. Сейчас нас интересует только сам факт утечки определенной информации об исходном сообщении. Можно, однако, представить следующий случай, когда подобная атака имеет высокую результативность. Допустим, что распоряжения руководящего центра (открытые тексты) рассылаются исполнительным органам в виде

документов определенного формата и стиля. В этом случае различные варианты открытого текста отличаются только в определенных полях электронного бланка. Кроме того, в определенных случаях стандартный стиль распоряжений может привести к тому, что криптоаналитику будет неизвестна только малая часть исходного сообщения. Например, дата, объем перечисляемой суммы денег, имя исполнителя и т. п.

Вероятностные механизмы позволяют предотвратить утечку информации в приведенном выше способе криптоанализа. Так же, как и в случае одноключевых криптосистем, при использовании вероятностных механизмов шифрования в двухключевых шифрах данному исходному тексту соответствует множество возможных криптограмм $\{C_1, C_2, \ldots, C_N\}$, каждая из которых дешифруется с помощью секретного алгоритма дешифрования $\mathbf{D}_z$ (секретного ключа) в один и тот же исходный текст T: $T = \mathbf{D}_z(C_1) = \mathbf{D}_z(C_2) = \ldots = \mathbf{D}_z(C_N)$. Это возможно только в том случае, если длина шифртекстов превышает размер исходного текста. Если длина шифртекста больше длины исходного текста на r бит, то можно построить такой вероятностный механизм, для которого число шифртекстов, соответствующих данному входному тексту, составляет $N = 2^r$. При вероятностном шифровании для данного T формируется одна из возможных криптограмм, например, C_i, выбираемая из множества $\{C_1, C_2, \ldots, C_N\}$ по случайному закону.

Криптоаналитик может правильно выбрать исходный текст, но он не сможет проверить этот факт, поскольку при шифровании он получит в общем случае другую криптограмму из множества возможных вариантов. Криптоаналитик, зашифровав T, получит $C_j = \mathbf{E}_z(T)$. Вероятность того, что $C_j = C_i$, составляет $1/N$. Криптоаналитику надо угадать не только исходный текст, но и значение некоторого случайно выбираемого параметра, который управляет процессом вероятностного шифрования.

1.5.5. Открытый шифр Эль-Гамаля

Рассмотрим способ шифрования Эль-Гамаля, который использует вероятностный механизм преобразования исходного текста. Этот способ основан на процедурах возведения в дискретную степень и состоит в следующем. Аналогично методу открытого распределения ключей Диффи—Хеллмана, выбирается большое простое число p и соответствующий ему первообразный корень α. Каждый пользователь секретной сети выбирает секретный ключ x, вычисляет свой открытый ключ $y = \alpha^x$ и помещает y в заверенный справочник. Чтобы послать пользователю i секретное сообщение T, отправителю надо выполнить следующие шаги:

1. Выбрать случайное число R, которое является взаимно простым с числом $p - 1$.
2. Вычислить значение $C' = \alpha^R \pmod p$.
3. По открытому ключу i-го пользователя вычислить $C'' = y^R T \pmod p$.
4. Отправить криптограмму (C', C'') пользователю i.

В этом методе длина шифртекста примерно в 2 раза больше длины исходного текста, и данному открытому тексту соответствуют не менее 2^k различных криптограмм (k — длина модуля p в битах). Пользователь i, получив криптограмму (C', C''), может легко вычислить исходный текст: $T = C''/(C')^x \pmod p$. Действительно:

$$(C')^x = (\alpha^R)^x = \alpha^{Rx} \pmod p \text{ и } C''/(C')^x = y^R T/\alpha^{Rx} = (\alpha^x)^R T/\alpha^{Rx} = \alpha^{xR} T/\alpha^{Rx} = T \pmod p.$$

Для двухключевых шифров общим методом введения случайности в процесс шифрования может также служить простой механизм присоединения случайных данных к шифруемому сообщению. В этом случае оговаривается размер t блока исходного текста, а также то, что случайный двоичный вектор присоединяется со стороны, например, старших разрядов. Структура шифруемого блока B может быть следующей: $B = R \parallel T$, где R — случайное число и T — блок исходного текста. Очевидно, что численное значение блока B не должно превышать максимально допустимое значение для используемой криптосистемы. Поэтому размер b блока B должен быть выбран таким, чтобы при максимальном значении он не выходил за пределы допустимых значений. Например, в шифре RSA значение B должно быть меньше модуля $n = pq$. Отсюда получаем условие $2^b < n$ или $b < \log_2 n$.

Если перечисленные выше условия соблюдаются, то последовательные шаги шифрования исходного текста описываются следующим образом:

$$T \to B = R \parallel T \to C = \mathbf{E}_z(R \parallel T),$$

а расшифрование криптограммы выполняется так:

$$C \to B = \mathbf{D}_z(C) \to R \parallel T \to T.$$

Любое сообщение можно разбить на тексты необходимого размера и каждый из них зашифровать с использованием вероятностного шифрования.

1.6. Применение шифрования на практике

1.6.1. Алгоритмы шифрования в средствах защиты ЭВМ

Современные микроэлектронные технологии обеспечивают непрерывный рост качества средств вычислительной техники (СВТ) и служат базой для сохранения основных тенденций ее развития — миниатюризации, снижения энергопотребления, увеличения объема оперативной памяти и емкости встроенных и съемных накопителей, роста производительности и надежности, расширения сфер и масштабов применения.

Данные тенденции привели к тому, что на современном этапе защита информации в компьютерных системах от несанкционированного доступа (НСД) харак-

теризуется возрастанием роли программных механизмов защиты по сравнению с аппаратными. Следует отметить, что роль физических и аппаратных средств защиты, например, направленных на защиту от побочного электромагнитного излучения, наводок по цепям питания, акустического прослушивания и т. п., является традиционной, и в этих областях применение компьютерной техники и специального программного обеспечения имеет вспомогательный характер. Возрастание роли программных средств защиты нужно понимать в том смысле, что возникающие новые проблемы в области защиты вычислительных систем (ВС) от НСД требуют использования механизмов и протоколов со сравнительно высокой вычислительной сложностью и могут быть эффективно решены путем использования ресурсов ЭВМ.

Применение чисто программных механизмов в средствах защиты информации (СЗИ) позволяет эффективно решить важные практические задачи. Среди таких задач можно упомянуть снижение стоимости СЗИ, уменьшение сроков разработки защищенных информационных технологий, ускорение внедрения СЗИ в массовом масштабе и обеспечение высокой переносимости на различные технические платформы и совместимости с системным и прикладным программным обеспечением.

При построении программных средств защиты информации целесообразно применение техники шифрования на различных уровнях обработки данных в ЭВМ, что позволяет экономично, с использованием чисто программных модулей, решить следующие задачи:

- ❒ защиты КС от программных закладок;
- ❒ защиты информации при хищении носителей или компьютера в целом;
- ❒ обнаружения несанкционированных изменений в информационно-программном обеспечении;
- ❒ защиты от проникновения вирусов;
- ❒ широкого разграничения полномочий пользователей;
- ❒ сохранения высокой производительности и привычной технологии работы пользователей;
- ❒ обеспечения защиты информации при проведении ремонтных работ

и др.

ВС подвержены широкому спектру потенциальных угроз для информации. Это обусловливает необходимость предусмотреть большой перечень возможных средств и функций защиты. Наиболее целесообразно в первую очередь обеспечить защиту самых информативных каналов утечки информации: возможности быстрого и легко реализуемого копирования данных на сменные магнитные носители большой емкости, открытых каналы связи, хищения ЭВМ или встроенных накопителей. Проблема перекрытия этих каналов утечки усложняется тем, что процедуры защиты данных не должны приводить к заметному снижению производительности КС.

Это определяет следующие высокие требования к шифрам, ориентированным на применение в системах защиты от НСД и функционирующим в масштабе реального времени.

- Высокая стойкость при криптоанализе на основе большого объема известного или подобранного текста, зашифрованного на одном и том же ключе.
- Высокая скорость шифрования при программной реализации.
- Сохранение возможности произвольного доступа к данным. Последний момент определяет необходимость использования блочных шифров в СЗИ от НСД.

Современная система компьютерной безопасности, ориентированная на массовое применение, должна быть стойкой, технологически эффективной и эргономичной. Перечислим ряд основных свойств, которые делают систему привлекательной для широкого круга пользователей.

- Комплексность — возможность установки самых разнообразных режимов защищенной обработки информации с учетом специфических требований различных пользователей.
- Совместимость — система должна быть совместима со всеми программами, написанными для данной ОС, и должна обеспечивать защищенный режим работы компьютера в сети.
- Переносимость — возможность установки системы на различные типы компьютерных систем, включая портативные.
- Удобство в работе — система должна быть проста в эксплуатации и не должна менять привычную технологию работы пользователей.
- Работа в масштабе реального времени — процессы преобразования информации, в т. ч. шифрования, должны выполняться с большой скоростью.
- Высокий уровень защиты информации.
- Минимальная стоимость системы.

В качестве идеологической основы для разработки СЗИ массового назначения предлагается широкое использование методов скоростного шифрования программными средствами с использованием новых типов блочных шифров. Для построения эффективных СЗИ масштаба реального времени можно предложить такие принципы:

- глобальное шифрование — вся информация на встроенном магнитном носителе, включая загрузочный сектор, системное и прикладное программное обеспечение и т. д., должна быть преобразована скоростным программным шифром;
- базовая ОС должна быть сохранена в неизменном виде, чтобы обеспечить высокую переносимость и совместимость;

- использование специального криптографического модуля для инициализации процесса первоначальной загрузки и обеспечения полного контроля за процедурой загрузки.

Указанные выше механизмы являются основными. Их необходимо дополнить целым рядом дополнительных модулей для решения стандартных задач защищенного функционирования ВС, например, таких как контроль целостности программ и данных, гарантированное уничтожение остаточной информации, блокировка клавиатуры и экрана, контроль за доступом к файловой системе, защита информации, передаваемой по линиям связи в вычислительных сетях и др.

В современных программных системах компьютерной безопасности используется технология так называемой прозрачной защиты, согласно которой для пользователя не меняется привычная среда его работы, и он не испытывает неудобств, вызванных функционированием защитных средств. Другими словами, система защиты при функционировании является для пользователя невидимой. Базой технологии прозрачной защиты является метод динамического шифрования конфиденциальной информации, с которой работает пользователь. Конфиденциальная информация, записываемая на внешние устройства, подвергается автоматическому зашифрованию по ключу, зависящему от пароля пользователя. При считывании санкционированным пользователем эта информация автоматически расшифровывается. Такое динамическое шифрование информации, по причине того, что его не замечает пользователь, называют *прозрачным* шифрованием, или прозрачным криптографическим преобразованием.

В комплексных системах компьютерной безопасности (например, СЗИ НСД "Кобра") можно выделить подсистемы, перечень и содержание которых приведены на рис. 1.15.

К не раскрытой на рис. 1.15 группе вспомогательных подсистем можно отнести следующие:

- подсистема затирания, предназначенная для гарантированного уничтожения остаточных данных в областях внешней памяти;
- подсистема блокировки клавиатуры и монитора, обеспечивающая блокировку устройств доступа к компьютеру по тайм-ауту при оставлении компьютера без присмотра;
- подсистема окончания работы на компьютере, предназначенная для корректного завершения сеанса пользователя, в процессе которого выполняется проверка соответствия состояния операционной среды эталонному состоянию и занесение соответствующей записи в журнал учета работ;
- подсистема создания дополнительных логических дисков, позволяющая создать такие диски на винчестере с сохранением на нем информации и без его повторного разбиения.

Обеспечиваемый уровень защиты информации зависит как от качества проектирования и полноты системы компьютерной безопасности, так и от настройки

режимов ее функционирования с учетом модели нарушителя. Администратор безопасности должен настроить необходимую схему защиты. Под схемой защиты понимается совокупность активизируемых компонентов системы защиты, а также параметры их настройки и устанавливаемые режимы функционирования.

Рис. 1.15. Структура системы "Кобра" и назначение основных компонентов

Для формирования конкретной схемы администратор службы безопасности должен выполнить следующие этапы:

1. Определение и уяснение требований к защите на основе выявления и анализа угроз информации в компьютерной системе.
2. Определение требуемых уровней защиты на основе анализа возможностей системы защиты информации.
3. Систематизация полученной информации и окончательное формирование схемы защиты.

При выполнении этапов необходимо учитывать, что эффективная защита должна удовлетворять двум противоречивым требованиям:

- обеспечение надежной защиты информации в компьютерной системе;
- обеспечение удобства работы пользователей на компьютере (отсутствие назойливости со стороны системы защиты, а также других недостатков, мешающих нормальной работе пользователей).

Учет этих требований приводит к повышению эффективности использования системы компьютерной безопасности. После определения схемы защиты администратор службы безопасности может приступить к установке и настройке всех необходимых компонентов системы защиты.

Типовыми возможностями по разграничению доступа пользователей к компьютерным ресурсам, предоставляемыми системами защиты, являются следующие:

- разграничение на уровне логических дисков и портов ввода/вывода, реализуемое подсистемой санкционирования доступа;
- разграничение на уровне файлов и каталогов, реализуемое подсистемой разграничения доступа к файлам.

Многие системы компьютерной безопасности поддерживают многоуровневое шифрование — дисковое шифрование, шифрование файлов, шифрование загрузочного сектора.

1.6.2. Особенности приложений

Длина ключа и стойкость

Длина ключа определяет верхнюю границу стойкости криптосистемы. Нападающий всегда может воспользоваться силовой атакой, которая состоит в тотальном переборе по всему пространству возможных ключей. Однако размер этого пространства при увеличении длины ключа растет по экспоненциальному закону. Если длина ключа в битах $l = 64$, то число возможных ключей составляет более 10^{19}. При $l = 128$ их число уже составляет более 10^{38}. В настоящее время вычислительные технологии находятся близко к решению задачи перебора 10^{20} вариан-

тов за разумный интервал времени. Перебор 10^{38} вариантов представляется недостижимым не только для современных технологий, но и в обозримом будущем. В настоящее время происходит массовый переход от 56-битового ключа в криптосистеме DES к 80-битовому или 128-битовому ключу в новых симметричных шифрах.

К доказуемо стойким блочным криптосистемам можно отнести шифры, для которых теоретически было бы доказано, что лучший способ криптоанализа имеет трудоемкость не меньше некоторого значения, которое гарантирует, что в ближайшее время этот способ будет вычислительно неосуществимым. В настоящее время для многочисленных шифров предложены различные методы криптоанализа и дана оценка трудоемкости решения криптоаналитической задачи для каждого из них. Однако существует принципиальная теоретическая трудность нахождения лучшего алгоритма криптоанализа, что и определяет сложность общего доказательства стойкости.

Для того чтобы новый шифр мог быть принят на вооружение, необходимо выполнить следующие условия:

- он должен быть составлен в соответствии с требованиями конкретного приложения и включать механизмы, реализующие современные принципы задания криптографической стойкости;
- в течение длительного времени он должен тестироваться опытными экспертами и должна быть доказана его стойкость ко всем известным методам криптоанализа.

Тестирование шифров относится к наиболее сложному и дорогостоящему этапу разработки криптосистем. Для повышения уверенности в стойкости новых шифров их подвергают испытаниям в условиях, благоприятствующих успеху решения криптоаналитической задачи. Например, исследуются модификации с уменьшенным числом раундов шифрования, предполагается возможность генерации аппаратных ошибок шифратора, считается, что часть ключа шифрования известна нападающему. В случае тестирования недетерминированных шифров рассматриваются нападения на основе известной модификации алгоритма шифрования и на основе выбора наиболее слабой модификации. Могут быть использованы варианты атак на модификации шифра с уменьшенным размером ключа шифрования и с уменьшенным размером входного блока данных. Чем строже процедура тестирования, тем больше гарантий, что не будут найдены новые специфические способы нападений. Теоретические гарантии стойкости могут дать только шифры с бесконечным случайным ключом. Однако для защиты компьютерной информации они представляются крайне неудобными. Кроме того, в таких приложениях возникает проблема защиты ключа, объем которого равен объему информации.

Близким к понятию секретного ключа является понятие *пароля*. Пароль также представляет собой секретный элемент. Обычно под ключом понимают элемент,

управляющий процессом шифрования, а под паролем — элемент, служащий для аутентификации субъекта (например, пользователя или рабочей станции). Во многих случаях пароль служит в качестве информации, управляющей процессом шифрования, а секретный ключ — для идентификации субъекта. Случайный ключ или пароль труден для запоминания, поэтому его часто хранят на съемных носителях, которые, в свою очередь, хранятся в условиях, обеспечивающих защиту от несанкционированного доступа. Выбор пароля, удобного для запоминания, существенно сужает число возможных его вариантов, поэтому требуется выбирать более длинные пароли или даже целые фразы.

При использовании двухключевых шифров применяются секретные ключи, имеющие длину существенно бóльшую, чем длина ключей в одноключевых криптосистемах. Это связано с особенностями криптоаналитической задачи в случае асимметричных шифров. В некоторых двухключевых криптосистемах секретные ключи выбираются случайно, а затем генерируются соответствующие им открытые ключи (например, в методе Диффи—Хеллмана и цифровой подписи Эль-Гамаля). В других криптосистемах формируются секретные ключи, удовлетворяющие специальным требованиям (например, множители p и q в системе цифровой подписи RSA). Во втором случае секретные ключи также выбираются случайно, однако после случайного выбора секретного параметра осуществляется его проверка на удовлетворение определенным условиям (например, требуется использование простых чисел или чисел взаимно простых с некоторым уже выбранным параметром).

Случайные секретные ключи удобно хранить на отчуждаемых электронных устройствах (электронных ключах, интеллектуальных электронных картах). Возможно комбинированное использование запоминаемых паролей и секретных ключей, хранимых на отчуждаемых носителях.

Шифрование и архивирование

При создании секретных архивов предполагается хранение данных в зашифрованном виде. Как правило, архивы содержат большой объем информации, поэтому целесообразно при архивировании выполнять сжатие данных, устраняющее избыточность информации. Благодаря этой процедуре значительно уменьшается размер архива и упрощается создание рабочих и резервных копий. При передаче по низкоскоростным каналам связи больших массивов данных также целесообразно использовать предварительное сжатие данных. Шифрование преобразует исходный текст в псевдослучайную последовательность, которая не сжимается, поэтому процедуру сжатия данных необходимо выполнить до их шифрования.

Замечание

Одним из тестов, с помощью которых предварительно дается качественная оценка стойкости шифра, является тест на сжатие криптограммы. Если криптограмма плохо сжимается, то считается, что шифр удовлетворяет этому тесту.

Таким образом, при архивировании секретных данных (или передаче по телекоммуникационному каналу) выполняется следующая последовательность преобразований:

1. Сжатие данных.
2. Шифрование.
3. Запись на носитель информации (или передача по каналу связи).

Для извлечения данных из архива или приема по каналу связи выполняется обратная последовательность процедур:

1. Считывание шифртекста с носителя (или прием зашифрованных данных по каналу связи).
2. Расшифрование.
3. Восстановление (разархивирование) данных.

Устранение избыточности исходного сообщения значительно повышает стойкость шифрования к атакам на основе шифртекста. Однако по отношению к атакам на основе известного текста стойкость шифрования не изменяется, если используется известный алгоритм сжатия. Поскольку мы допускаем, что оппонент знает алгоритм шифрования, то тем более мы должны предполагать, что он знает и алгоритм сжатия информации.

Представляет интерес совместное использование процедуры сжатия данных и вероятностного шифрования с простым механизмом включения случайных данных в криптограмму. Каждое из этих двух дополнительных преобразований повышает стойкость шифрования. При этом одно из них уменьшает исходный размер сообщения, а другое — увеличивает. Вероятностные шифры на основе конкатенации блоков шифруемых данных со случайными двоичными векторами позволяют легко регулировать степень увеличения размера криптограммы, поэтому параметры вероятностного шифрования можно подобрать такими, что порождаемый шифртекст будет обладать размером, примерно равным размеру сообщения до устранения избыточности.

Шифрование и кодирование

При передаче конфиденциальной информации по телекоммуникационным каналам с шумом, кроме шифрования, необходимо осуществить также помехоустойчивое кодирование передаваемых данных. Для блочных криптосистем процедуры шифрования и дешифрования обладают выраженным рассеиванием влияния входных знаков на многие выходные, причем такое влияние имеет псевдослучайный характер. В связи с этим при использовании блочных криптосистем на передающем пункте процедуру помехоустойчивого кодирования необходимо осуществить после шифрования, а на приемном пункте сначала выполнить декодирование, а затем — расшифрование.

При использовании поточного шифрования, которое предусматривает выработку ключевой гаммы по секретному ключу и синхронное побитовое наложение гам-

мы на передаваемое сообщение, размножение ошибок при расшифровании не имеет места. В этом случае при любой очередности кодирования и шифрования на приемном конце канала расшифрование и декодирование могут быть выполнены с одинаковым успехом. Например, при передаче и приеме можно использовать следующую очередность преобразования данных: кодирование — зашифрование — расшифрование — декодирование.

1.6.3. Стандартизация алгоритмов шифрования

Значительная роль алгоритмов шифрования в вопросе решения задач защиты электронной информации в массовом масштабе обусловила необходимость принятия стандартов шифрования, из которых наиболее известным и широко используемым является американский федеральный стандарт DES, принятый в середине 1970-х годов и послуживший аналогом для российского стандарта ГОСТ 28147-89. В настоящее время данные стандарты представляются архаичными и не соответствующими современным требованиям по скорости преобразования данных. Кроме того, в DES используется 56-битовый секретный ключ, который в настоящее время не может обеспечить достаточной стойкости к атаке, основанной на переборе возможных значений ключа, а российский стандарт является сложным для аппаратной реализации. Существенным недостатком шифра ГОСТ 28147-89 является то, что в нем используются секретные долговременные элементы, к которым относятся таблицы замены, поставляемые "в установленном порядке". По общепринятым криптографическим требованиям стойкость алгоритма должна основываться только на секретности легко сменяемого элемента — ключа. Оба шифра обладают низкой стойкостью к атакам на основе случайных аппаратных ошибок. Эта атака является актуальной для шифров, применяемых в интеллектуальных электронных карточках. В этом типе нападений предполагается, что нарушитель имеет возможность оказать на шифратор внешнее физическое воздействие и вызвать одиночные случайные ошибки в регистрах, содержащих данные, при выполнении процедуры шифрования.

Тот факт, что первый в мире открытый официальный стандарт шифрования был принят в США, сыграл важную роль в закреплении лидерства американских производителей средств защиты информации. Влияние американского стандарта DES на тенденции развития открытой криптографии во всем мире оказалось настолько большим, что все прошедшие годы разработчики блочных алгоритмов шифрования были "загипнотизированы" подходами, использованными при разработке DES. Общая схема построения криптосистемы DES стала образцом для подражания. Последнее привело к определенному сдерживанию новых подходов к построению блочных криптосистем, которое отчетливо проявилось в начале 1990-х годов, когда возникла широкая потребность в скоростных методах шифрования, ориентированных на программную реализацию.

Разработке новых алгоритмов шифрования, удовлетворяющих современным технологическим требованиям, ведущие страны уделяют большое внимание.

В 1998—2000 гг. США, сохраняя свое первенство в этом направлении, провели всемирный конкурс на принятие своего нового стандарта — Advanced Encryption Standard (AES) (**http://www.nist.gov/aes**). Этому конкурсу было посвящено несколько конференций, на которых рассматривались и обсуждались представленные шифры. В конкурс были вовлечены ведущие криптографы мира, которые представили 15 кандидатов AES. Конкурс завершился. Победителем стал бельгийский шифр Rijndael. Сильную конкуренцию победителю составили шифры RC6 и Towfish. Вслед за этим был объявлен аналогичный европейский конкурс. Через распространение своих алгоритмов шифрования и средств ЗИ ведущие страны закрепляют позиции на рынке продуктов и услуг в области информационной безопасности, а также усиливают влияние и приобретают в определенной степени возможность контроля защищенности информационного ресурса других стран.

В России также официально декларирована проблема разработки скоростных шифров, соответствующих современным мировым требованиям и тенденциям. Одной из последних является возрастание технологических требований к алгоритмам шифрования. При сохранении требования обеспечения гарантированной стойкости стоит задача обеспечения высокой скорости шифрования как при программной (более 100 Мбит/с), так и при аппаратной реализации (более 1000 Мбит/с). При этом устройства шифрования по своей стоимости должны быть доступны массовому пользователю. Возникает необходимость обеспечения стойкости алгоритмов к ряду нетрадиционных атак, например к атаке на основе случайных аппаратных ошибок.

Потребности практической информатики привели к формированию зарубежной открытой криптографической школы в середине 1970-х годов. В России открытая криптография начала активно формироваться с середины 1990-х годов. В настоящее время прикладная криптографическая тематика прочно внедрилась в такие журналы, как "Автоматика и телемеханика", "Проблемы передачи информации", "Вопросы защиты информации", "Безопасность информационных технологий" и др. Российские авторы получили более 30 патентов на изобретения в области шифрования. На конференциях по безопасности информации, как правило, рассматриваются и вопросы шифрования. В конце 1999 года в России была образована общественная организация — Ассоциация "РусКрипто", которая призвана популяризировать криптографические знания и проводить работу по повышению качества разрабатываемых в России средств защиты информации и расширению российских открытых исследований по криптографии.

Российские подходы к построению скоростных шифров и ряд конкретных шифров обладают элементами мировой новизны. Например, предложенные способы шифрования (патенты РФ №№2140710, 2140714, 2140716, 2141729) на основе перестановок, зависящих от преобразуемых данных, имеют большие преимущества при аппаратной реализации. Данный подход получил достаточное теоретическое и экспериментальное обоснование, а также привел к формированию ново-

го направления в прикладной криптографии, связанного с использованием специально проектируемых криптографических операций — криптографических примитивов.

Первостепенным требованием к алгоритмам шифрования является высокая стойкость. Если это требование не выполняется, то остальные характеристики просто-напросто теряют свое значение. Специалист, знакомый с основами криптографии и имеющий практический опыт в данной области, может достаточно быстро разработать стойкий алгоритм, однако к алгоритмам широкого применения предъявляется также и ряд других требований. Перечислим некоторые из них.

- Необходимо убедить специалистов и потребителей в стойкости предлагаемого алгоритма. Для этого формулируются критерии построения, проводятся теоретические исследования используемых операций и механизмов преобразования. Выполняются объемные исследования статистических свойств шифра.
- Необходимо убедить пользователей в том, что в предлагаемом алгоритме отсутствуют "потайные двери", дающие возможность читать шифртексты без знания секретного ключа или вычислять его.
- Алгоритм должен обладать необходимыми характеристиками, обеспечивающими невысокую стоимость аппаратной реализации (и/или удобство программной реализации).
- Алгоритм должен обеспечивать высокую скорость шифрования.
- Алгоритм должен обеспечивать высокую стойкость не только к традиционным криптоаналитическим атакам (на основе известного исходного текста и на основе специально подобранных исходных текстов или шифртекстов), но и к атакам, связанным со специальными применениями, например, допускающими атаку на основе:
  - генерации случайных аппаратных ошибок;
  - измерения потребляемой мощности;
  - измерения времени вычислений

  и другим типам атак.

К кандидатам AES предъявлялось требование высокой скорости как при программной, так и при аппаратной реализации (при условии экономичности схемотехнического воплощения). Это представляется целесообразным, поскольку алгоритм широкого применения, ориентированный, в первую очередь, на обеспечение конфиденциальности информации, предполагает разные виды реализации. При этом крайне трудно обеспечить максимальные значения скорости шифрования. Очевидно, что алгоритмы, ориентированные только на программную или только на аппаратную реализацию, могут обеспечить более высокую производительность в соответствующих областях. Это обстоятельство представляется весьма важным при разработке шифров с повышенными технологическими требованиями. В эпоху развитости информационных технологий даже незначитель-

ные потери производительности после многократного тиражирования выливаются в существенные экономические потери. В критических областях применения даже небольшая задержка может привести к катастрофическим последствиям. Эти соображения свидетельствуют о том, что принятие общего стандарта шифрования не снимает всех проблем прикладной криптографии. В непрерывном развитии находится не только криптоанализ уже разработанных алгоритмов, но и разработка новых шифров.

1.6.4. Вопросы встраивания потайных лазеек

Под *потайной лазейкой* (дверью, ходом) понимается наличие некоторого секрета, знание которого позволяет достаточно легко раскрывать шифр, т. е. осуществлять чтение шифртекста без знания ключа или вычислять ключ шифрования. Одним из важных вопросов широкого применения методов шифрования является обеспечение доверия со стороны массового пользователя. В алгоритме DES за период времени, превышающий 20 лет, не обнаружено потайных лазеек, но большие сомнения в этом вопросе у многих пользователей остаются и по сей день. Использование секретных подстановок в ГОСТ 28147-89 затрудняет подробное исследование по проблеме потайных ходов, что усиливает недоверие независимых пользователей. Насколько существенна для пользователей угроза того, что им могут быть навязаны шифры с потайными ходами? Длительное время рассмотрение вопроса о потайных ходах в открытой криптографической литературе носило односторонний характер, а именно, они были направлены на поиск потайных ходов в уже разработанных шифрах. Бельгийские исследователи V. Rijmen и B. Preneel в статье [113] по-другому поставили вопрос о потайных ходах: можно ли в принципе разработать шифры, для которых установить наличие потайных ходов было бы вычислительно невозможным? Даже при условии, что алгоритм шифрования опубликован в открытой печати и исследован во всех подробностях широким кругом пользователей и специалистов в области криптографии. Исследование бельгийцев дало положительный ответ на этот вопрос. Они показали пути практического построения блочных DES-подобных шифров, содержащих потайные лазейки, для которых обнаружение наличия последних было бы вычислительно неосуществимой задачей в настоящее время. Этот результат является достаточно пессимистическим для пользователей. Если с риском наличия потайных ходов в шифрах следует считаться, то можно ли избежать атак на основе скрытых лазеек? Полной гарантии защиты от такой атаки дать трудно. Однако могут быть предложены следующие подходы существенного снижения риска.

- Пользователи или его доверенные представители должны участвовать в разработке блоков подстановки или всего шифра целиком (этот подход требует наличия профессиональной подготовки у пользователя; такой подход имеет и другой недостаток, который связан с доверием других пользователей к выработанной таким способом криптосистеме).

- Можно отказаться от использования криптосистем с фиксированными подстановками и отдать предпочтение шифрам, в которых подстановки (например, таблицы подстановок) формируются по секретному ключу пользователя (примером являются шифры Blowfish, Twofish).
- Отказ от использования подстановок в качестве базового криптографического примитива (примерами являются шифры RC6, SPECTR-H64); этот вариант также связан с вопросом о потайных ходах — не будут ли впоследствии найдены возможности внедрения потайных ходов также и в шифры без подстановок?
- Можно использовать криптосистемы, в которых применяются управляемые операции, т. е. такие криптографические примитивы, которые задают преобразования, зависящие от переменных параметров процесса шифрования — от секретного ключа пользователя и/или от шифруемого сообщения. Применение операций, зависящих от преобразуемых данных, в качестве криптографических примитивов дает высокие гарантии избежать потайных ходов в используемых шифрах. Примерами таких шифров являются алгоритмы RC5, RC6, MARS и SPECTR-H64 с операциями, зависящими от преобразуемых данных.
- Для нейтрализации потайных лазеек могут быть использованы схемы упрощенного вероятностного шифрования, подробно описанные в *гл. 6*. Для сохранения исходного объема данных можно осуществить предварительное сжатие информации (заметим, что этот способ одновременно приводит к существенному повышению стойкости шифрования).
- Можно использовать двукратное шифрование с помощью двух разных алгоритмов, между разработчиками которых сговор является маловероятным, например один раз данные зашифровать с использованием DES, а затем по ГОСТ 28147-89 (один шифр нейтрализует потенциальную потайную лазейку другого и наоборот).

1.6.5. Криптография и стеганография

Стеганографией называется техника скрытой передачи или скрытого хранения информации. Целью стеганографии является сокрытие самого факта передачи сообщений. Для этого могут быть использованы невидимые чернила, акростих, микрофотография, тайники. Использование электронных средств обработки информации позволяет применить новые стеганографические методы. Стеганографические методы используют распределение по псевдослучайному закону информации в пространстве или времени (применяется также комбинирование этих способов), зашумление и маскирование в некотором сообщении-контейнере или в служебной информации. В мультимедиа они могут быть основаны, например, на передаче сообщения внутри видеокадров.

Принципиальное отличие криптографии от стеганографии состоит в том, что в ней не скрывается факт передачи сообщений, а скрывается только его содержа-

ние (смысл). Применение стеганографии для передачи важных сообщений является весьма рискованным. Криптография представляется гораздо более надежным инструментом для защиты информации. Стеганографические методы могут обеспечить высокий уровень защиты информации только в том случае, когда они будут дополнены предварительным криптографическим преобразованием сообщения. Причем последние будут определять реальную стойкость такой комбинированной секретной системы.

В настоящее время разработчики стеганографических методов используют секретный ключ, что фактически означает разработку стеганографических методов, заключающих в себе криптографическую подсистему. То обстоятельство, что стеганография пришла к необходимости применения секретного ключа, означает признание того факта, что разработка стойких методов сокрытия факта передачи сообщений связана с использованием элементов криптографии. Идеи одноключевой и двухключевой криптографии могут существенно обогатить стеганографию.

Стеганография не заключает в себе тех многогранных внутренних возможностей, которые являются свойственными криптографии и которые определили важнейшую роль криптографии в обществе с развитой информационной инфраструктурой.

ГЛАВА 2

Гибкие программные шифры

2.1. Секретность алгоритма и стойкость

В самом общем смысле любая блочная криптосистема является способом задания той или иной подстановки. Суть шифрования заключается в замене блока исходного текста на блок шифртекста. Очевидно, что для обеспечения стойкого шифрования требуется использовать достаточно сложные процедуры преобразования, которые, согласно правилу Керкхоффа, предполагаются известными атакующему. На практике наиболее часто используются шифры с размером входного блока 64 и 128 бит. Используются также 32-битовые блочные шифры (достаточно редко) и 512-байтовые. Любой блочный шифр формально можно представить в виде множества таблиц подстановок (таблиц замены), каждая из которых соответствует одному из возможных секретных ключей и задает (для данного ключа) соответствие всех допустимых входных сообщений и соответствующих им криптограмм. Однако размер каждой из этих таблиц неимоверно велик ($n2^n$ бит), что делает невозможным использование на практике табличного способа задания шифров.

Табличное описание шифров является наиболее общим. Шифры, задаваемые в виде алгоритма, составляют малую долю возможных подстановок, однако это именно те подстановки, которые могут быть использованы на практике, поскольку они описываются компактным способом. Таким образом, блочный шифр, представленный в виде некоторого алгоритма преобразования, реализует способ выбора конкретной таблицы замены входного блока на выходной в зависимости от секретного ключа. Смена секретного ключа означает смену таблицы замены, которую нет необходимости строить полностью. Работа с ней осуществляется как вычисление выходного блока по заданному входному блоку при известном ключе шифрования.

Алгоритм шифрования, обладающий высокой стойкостью, задает "псевдослучайное" преобразование, т. е. такое преобразование, которое практически неот-

личимо от случайного преобразования. Для хорошего блочного алгоритма без знания секретного ключа вычислительно сложно подобрать такие входные тексты, при которых наблюдение за соответствующими закрытыми текстами позволяет предсказывать связь очередного результата. Наблюдая все соответствующие им выходные тексты, можно было бы предсказать какую-либо связь результата преобразования еще одного специально подобранного текста с известными выходными текстами. На самом деле, такая зависимость уже задана самим алгоритмом шифрования, но для криптографически стойких преобразований она является псевдослучайной. Первый этап криптоанализа состоит в определении некоторых условий, в которых шифрующее преобразование можно отличить от случайного преобразования. Если такие условия определены, то это означает, что найдены предпосылки для вскрытия шифра. Однако надо иметь в виду, что задача вычисления секретного ключа может оказаться намного более трудоемкой, чем задача выявления отличия шифрующего преобразования от случайного.

Знание алгоритма преобразования является весьма существенным подспорьем для атакующего и позволяет исследовать статистические свойства шифра. При обнаружении определенных статистических связей между входными и выходными текстами появляются предпосылки для раскрытия (вычисления) секретного ключа. Возникает естественный теоретический вопрос о том, возможно ли построение криптосистем с секретным алгоритмом, обладающих безусловной стойкостью при использовании ключа конечного размера. Ниже мы покажем, что обеспечение безусловной стойкости шифров с секретным алгоритмом возможно только при бесконечной сложности шифрующих процедур. В этом случае время шифрования какого-либо текста бесконечно, что делает бессмысленным практическое использование таких шифров.

Рассмотрим вопрос о безусловной стойкости лучшей криптосистемы с секретным алгоритмом и с конечным временем шифрования. Мы предполагаем, что потенциальный злоумышленник является представителем земного мира, обладает доступом к любой современной вычислительной технике и знает язык, на котором написано исходное сообщение. Допустим, что он желает прочитать криптограмму, соответствующую исходному тексту с размером, многократно превышающим размер ключа. Поскольку мы рассматриваем безусловную стойкость, то предполагаем, что нападающий обладает бесконечными вычислительными ресурсами.

Размер текста, описывающего любой алгоритм с конечным временем шифрования, является конечным, поэтому нападающий может опробовать все возможные алгоритмы. Действительно, для достигнутого уровня развития технологии вычислений каждый конечный текст можно интерпретировать как алгоритм, написанный на языке машинных команд. При этом число вариантов интерпретации произвольной конечной последовательности битов является конечным, следовательно, при наличии бесконечных вычислительных ресурсов путем перебора можно определить как конечный секретный ключ, так и алгоритм шифрования.

Таким образом, с точки зрения безусловной стойкости неизвестность динамически формируемого алгоритма является несущественным фактором. Только разовый случайный ключ, длина которого равна длине сообщения, позволяет достигнуть безусловной стойкости. Последнее понятие имеет больше теоретический смысл, чем практический. Действительно, практика не является замкнутой. Все возможные случаи действий нападающего в реальных условиях функционирования криптосистемы не могут быть предусмотрены замкнутой теоретической моделью.

Использование длинных ключей порождает проблему усложнения процедур безопасного управления ключами, что может сделать безусловно стойкие шифры менее надежными, чем условно стойкие криптосистемы с ключом размером 512, 256 или даже 128 бит. Для условно стойких (т. е. вычислительно стойких) криптосистем секретность алгоритма является принципиальным моментом. Действительно, трудоемкость взлома криптосистемы оценивается по отношению к конкретным алгоритмам криптоанализа. За значение стойкости шифра принимается трудоемкость его раскрытия с использованием лучшего известного алгоритма криптоанализа.

Лучший алгоритм криптоанализа зависит от алгоритма шифрования. Кроме того, наиболее эффективные методы криптоанализа опираются на знание алгоритма шифрования и предварительное исследование его математических и вероятностных свойств. Это дает основание полагать, что сохранение алгоритма шифрования в секрете будет существенно повышать сложность криптоанализа. Такой способ повышения стойкости шифрования не применяется широко на практике в основном потому, что крайне трудно обеспечить секретность элементов криптосистемы, которые используются длительное время и являются известными достаточно широкому (хотя и первоначально ограниченному) кругу пользователей. При этом секретность алгоритма должна быть обеспечена также на стадии проектирования и тестирования.

Таким образом, в случае вычислительно стойких криптосистем секретность алгоритма шифрования может быть принципиально использована в качестве одного из механизмов значительного повышения стойкости. Однако для практического применения этого обстоятельства необходимо построить такую криптосистему, в которой алгоритм шифрования был бы легко сменяемым элементом. Это можно сделать, используя предвычисления, с помощью которых реально было бы осуществить генерацию алгоритма шифрования в зависимости от секретного ключа пользователя. Такие криптосистемы можно назвать недетерминированными или гибкими. Первый термин подчеркивает то, что криптоаналитику алгоритм шифрования неизвестен, второй — то, что алгоритм шифрования модифицируется в зависимости от секретного ключа.

В недетерминированных криптосистемах алгоритм предвычислений является формой задания большого числа возможных алгоритмов шифрования. Поскольку алгоритм предвычислений предполагается известным (это долговременный элемент шифра), то криптоаналитику известно описание возможных модификаций

шифрующих функций. Он не знает только выбор конкретной модификации. В *гл. 3* рассмотрены практические схемы построения недетерминированных шифров, позволяющие задать очень большое число различных модификаций алгоритма шифрования.

Алгоритм шифрования может рассматриваться секретным (в смысле неопределенности выбора модификации шифрующих процедур), несмотря на детальное обсуждение и исследование подобных криптосистем в открытой печати. Секретность алгоритма является специфической формой повышения стойкости криптосистемы и может быть эффективно использована для построения новых стойких шифров. Наличие этапа предвычислений является фактором, вносящим некоторые ограничения на применение в определенных специфических областях, например, в случаях необходимости частой смены ключа (каждую секунду или долю секунды). В большинстве приложений использование этапа предвычислений является допустимым, поэтому недетерминированные программные шифры потенциально являются весьма перспективными для широкого применения.

2.2. Принципы построения программных шифров

Под *программными* шифрами понимаются такие системы шифрования, которые используют операции над компьютерными словами и учитывают специфику обработки данных в компьютерных системах, что дает возможность получить высокие скорости шифрования при использовании микропроцессоров общего применения. С учетом массового характера проблемы защиты электронной информации программные шифры имеют перспективы широкого практического внедрения. Рассмотрим особенности проектирования программных шифров. Организация наиболее надежного защищенного режима обработки данных в компьютерных системах связана с шифрованием всех данных, хранимых в энергонезависимой памяти. В общем случае, обработка информации связана с произвольными запросами на чтение и запись данных, поэтому необходимо обеспечить независимое шифрование отдельных блоков данных.

Современные ЭВМ характеризуются большим объемом постоянной памяти и высокой скоростью записи и считывания. Эти два фактора определяют высокие требования к шифрам, ориентированным на применение в системах защиты информации от несанкционированного доступа, функционирующих в масштабе реального времени:

- высокую стойкость при криптоанализе на основе большого объема известного или подобранного текстов, зашифрованных на одном и том же ключе;
- высокую скорость шифрования при программной реализации.

Аддитивные поточные шифры имеют ограниченное применение в компьютерных системах, поскольку при использовании глобального шифрования (т. е. при

шифровании всей информации, хранимой в энергонезависимой памяти) необходимо обеспечить произвольный доступ к хранимой информации. Поточные криптосистемы могут быть использованы для этой цели только при их дополнении специальной подсистемой генерации уникальных ключей шифрования для каждого отдельного элементарного блока данных. (Это требование связано с тем, что в поточных шифрах нельзя допустить повторного использования одинаковых участков ключевого потока.)

Наиболее подходящими для применения в компьютерных системах являются блочные шифры. Как правило, в известных блочных шифрах ключ шифрования используется по детерминированному расписанию, т. е. на заданных шагах шифрования используются одни и те же подключи для всех преобразуемых входных блоков данных. Этот механизм позволяет легко задать дешифрующие процедуры и обеспечивает влияние всех битов ключа на результат преобразования всех блоков. Однако тот факт, что криптоаналитику известна последовательность применения подключей, требует использования достаточно большого числа раундов шифрующего преобразования. Последнее ведет к проблематичности получения высоких скоростей шифрования.

Для построения скоростных программных шифров была предложена идея использования псевдослучайных выборок из ключа шифрования [26, 90]. Криптосхемы с подобной недетерминированной выборкой подключей характеризуются тем, что элементы криптографического ключа, используемые при преобразовании входного блока данных, выбираются в зависимости от структуры преобразуемого блока и от структуры ключа шифрования. Для шифров, основанных на подобной криптосхеме, расписание выборки подключей не является предопределенным, что является существенным фактором повышения стойкости шифрования. За счет этого количество раундов шифрования может быть уменьшено, и тем самым повышена скорость шифрования. Шифры с выборкой подключей в зависимости от преобразуемых данных можно назвать псевдослучайными по характеру выборки подключей.

Выборка подключей в зависимости от преобразуемых данных в самом общем случае есть просто некоторая формальная операция отображения (подстановки), выполняемая по таблице, которая задается ключом шифрования. Ключ шифрования представляет собой совокупность пронумерованных подключей. Входной блок задает номер выбираемого подключа, а последний — значение, получаемое в результате операции отображения. Массив элементов, по которому выполняется операция отображения, иногда называется S-блоком (или блоком подстановки, блоком замены). Мы будем пользоваться термином "выборка подключей в зависимости от преобразуемых данных", поскольку подблок преобразуемых данных, который определяет текущий выбираемый подключ, не заменяется выбранным подключом, а *вместе с этим подключом* участвует в дальнейших преобразованиях. В таких случаях процесс шифрования представляется более наглядным и понятным, если он описывается в терминах, близких к реально выполняемым процедурам. Термин "операция подстановки" представляется предпочтительным

в случаях, когда процедура преобразования состоит в замене входного блока на значение из таблицы подстановки.

В случае программных шифров базовый принцип рассеивания и перемешивания дополняется следующими механизмами:

1. Зависимостью S-блоков от секретного ключа.
2. Увеличением размера S-блоков.
3. Увеличением размера блока преобразуемых данных до 512 байт.
4. Зависимостью алгоритма шифрования от секретного ключа. (Например, настройкой операций преобразования по секретному ключу.)
5. Зависимостью операций преобразования от преобразуемых данных (т. е. управлением выбором модификации текущей операции от входного сообщения).
6. Зависимостью выборки подключей от преобразуемых данных.
7. Различными сочетаниями механизмов, указанных в п. 1—5. (Например, заданием зависимости подмножеств возможных модификаций управляемой операции от секретного ключа, т. е. сочетанием п. 3 и 4.)

Несмотря на большое число различных механизмов шифрования и их конкретных форм реализации, в самом общем случае блочный шифр представляет собой некоторое множество подстановок, заданных на множестве возможных входных блоков. Число подстановок не превышает числа возможных различных секретных ключей, предусмотренных в шифре. Выбор секретного ключа соответствует выбору конкретной подстановки. Таким образом, шифр представляет собой способ задания простой подстановки на очень большом алфавите. Из-за того, что размер исходного алфавита очень большой (для n-битовых входных блоков число различных входных блоков составляет 2^n, а число возможных перестановок равно $2^n!$), подстановки не могут быть заданы табличным способом, и единственным практически удобным способом является их задание алгоритмическое, т. е. посредством описания правила вычисления выходных блоков по входным блокам.

Способ задания подстановки в виде процедур, управляемых секретным ключом, определяет тот факт, что подстановки выбираются случайно не из множества ($2^n!$) всех возможных подстановок, а из подмножества, состоящего из 2^k подстановок, где k — максимальная длина секретного ключа в битах. Поскольку алгоритм шифрования предполагается известным криптоаналитику, то это означает, что он знает подмножество подстановок, реализуемых этим алгоритмом. Сложность криптоанализа обусловлена тем, что число возможных подстановок крайне велико, и каждая из подстановок может быть рассмотрена только как описывающий ее алгоритм при некотором значении секретного ключа.

Конкретный вид алгоритма определяет правило построения подмножества возможных подстановок, а следовательно, блочный шифр обладает определенными

алгебраическими и статистическими свойствами, которые могут быть использованы для проведения криптоанализа. Вопрос только в том, насколько сложен тот или иной способ криптоанализа для конкретного алгоритма шифрования. Если в шифре с бесконечным ключом алгоритм преобразования играет второстепенную роль в задании криптостойкости, то в случае блочных шифров с фиксированным секретным ключом характеристики механизма шифрования являются определяющими. Принципы построения шифров и конкретные типы используемых механизмов ориентированы на то, чтобы задать псевдослучайную подстановку, которую трудно отличить от случайной при наличии у криптоаналитика большого (но практически разумного) объема исходных данных для осуществления атаки.

Сложность криптоанализа связана с двумя следующими обстоятельствами различного плана. При использовании алгоритмов шифрования с хорошими свойствами рассеивания и перемешивания трудоемкость криптоанализа состоит в *сложности анализа самих шифрующих процедур*. При использовании зависимости алгоритма шифрования от секретного ключа трудоемкость задачи криптоанализа существенно увеличивается за счет *неоднозначности интерпретации статистических связей, наблюдаемых между исходными текстами и соответствующими им криптограммами*. Шифры, в которых алгоритм шифрования формируется в зависимости от секретного ключа, называются гибкими или недетерминированными. В гибких шифрах данному ключу соответствует только одна реализуемая модификация алгоритма шифрования. Поскольку множество ключей ограничено, то это означает, что гибкий шифр представляет собой множество алгоритмов шифрования, описываемых с помощью некоторого алгоритма, который задает правило формирования алгоритма шифрования в зависимости от секретного ключа. Формирование секретных таблиц подстановок или алгоритма шифрования предполагает, что в программных шифрах используется некоторый этап предвычислений, выполняемый однократно при введении секретного ключа. На этом этапе осуществляется инициализация криптосистемы, которая в дальнейшем многократно выполняет процедуры зашифрования и расшифрования данных.

2.3. Инициализация программных шифров

Итеративные шифры, как правило, используют ключ размером от 56 до 256 бит. При длине ключа 80 бит и более атака на основе перебора возможных ключей в настоящее время практически не может быть реализована какой-либо организацией даже с большими вычислительными возможностями. Для обеспечения стойкости ко всем другим известным методам криптоанализа используется большое число раундов преобразования, на каждом из которых выполняются процедуры преобразования, обладающие хорошими свойствами рассеивания и перемешивания.

Очевидно, что при использовании более длинных ключей шифрования имеется больше возможностей для разработки скоростных шифрующих процедур. Одна-

ко при использовании длинных ключей усложняется управление ключами (генерация, передача и хранение ключей). В криптосистемах для программной реализации возможность применения и хранения длинных ключей шифрования может быть обеспечена применением специальных процедур одностороннего преобразования исходного секретного ключа сравнительно небольшой длины в расширенный криптографический, размер которого может достигать от 1 до 64 Кбайт. В этом случае достаточно обеспечить управление секретными ключами, поскольку при необходимости ключ шифрования может быть сформирован по известным процедурам в зависимости от секретного ключа. Этап формирования ключа шифрования является этапом предвычислений, выполняемых однократно при включении шифратора или ЭВМ. При использовании такой схемы построения шифра нападающий стоит перед выбором двух основных вариантов атаки.

1. Он может предполагать, что секретный ключ выбирается случайно, и рассматривать сравнительно простые процедуры шифрования с целью обнаружения в них тех или иных слабостей.
2. Он может рассматривать значительно более сложные процедуры формирования ключа шифрования и попытаться решать уравнения шифрования относительно секретного ключа.

Очевидно, что процедуры формирования ключа шифрования должны быть достаточно сложными, чтобы по отдельным фрагментам ключа шифрования нельзя было бы вычислить какие-либо части секретного ключа. Основными требованиями к алгоритму формирования ключа шифрования являются:

- примерно равновероятное влияние каждого бита секретного ключа на все биты формируемого расширенного ключа;
- высокая вычислительная сложность определения секретного ключа по известному расширенному ключу.

Высокая криптостойкость шифров на основе механизма расширения ключа принципиально достижима, поскольку на этапе предвычислений можно использовать алгоритмы, принадлежащие к более широкому классу преобразований по сравнению с алгоритмами шифрования. Известно, что принципиальным ограничением последних является необходимость выполнения обратного преобразования, контролируемого ключевыми параметрами и восстанавливающего по шифру исходное сообщение. На алгоритм формирования ключа шифрования подобное ограничение не накладывается, а в более широком классе алгоритмов можно найти и более стойкие с криптографической точки зрения.

Ввиду того, что процедуру настройки предполагается выполнять только при инициализации (включении) криптосистемы, то никаких существенных ограничений на длительность выполнения алгоритма настройки и на используемые вычислительные ресурсы не накладывается (подпрограмма предвычислений не является резидентной). Это означает, что приемлемо использование многозвенных односторонних процедур, в частности, преобразований с недетерминированным

алгоритмом — на первом этапе предвычислений генерируется алгоритм одностороннего преобразования, осуществляемого на второй стадии предвычислений, когда происходит окончательное преобразование секретного ключа в расширенный ключ шифрования.

При выполнении процедур настройки можно также предусмотреть генерацию уникальной для каждого пользователя модификации алгоритма шифрования [69, 94]. В действительности, алгоритм генерации процедур шифрования есть просто форма задания выбора по секретному ключу пользователя конкретной модификации алгоритма шифрования из очень большого числа возможных модификаций. Практическая реализация криптосистем с недетерминированным алгоритмом шифрования является специфической возможностью, предоставляемой программными средствам и вытекающей из идеи использования предвычислений.

Таким образом, программный шифр можно реализовать в виде двух подпрограмм — модуля инициализации, который предназначен для однократного запуска, и резидентного модуля шифрования, обслуживающего запросы других программ на зашифрование и расшифрование данных.

Одним из требований к алгоритму формирования ключа шифрования является следующее — количество возможных выходных последовательностей не должно быть существенно меньше числа возможных секретных ключей. Длина расширенного выходного ключа превышает длину секретного ключа, но для определенных процедур предвычислений может оказаться, что различным секретным ключам будут соответствовать одинаковые ключи шифрования. В случае применения односторонних преобразований на этапе формирования ключей шифрования значительное сужение пространства ключей шифрования является маловероятным, однако желательно получение гарантии того, что мощность множества различных ключей шифрования равна мощности множества секретных ключей специфицированной длины. Это легко достигается путем применения шифрующих процедур в качестве алгоритма построения расширенного ключа. Действительно, повторяя секретный ключ необходимое число раз, можно легко получить уникальный расширенный текст для каждого секретного ключа. Зашифровывая этот расширенный текст, получим шифртекст, который может быть взят в качестве расширенного ключа. Если необходимо построить недетерминированный шифр, то строим входной расширенный текст, размер которого превышает необходимую длину расширенного ключа. Часть преобразованного текста используем как расширенный ключ шифрования, а другую часть — для формирования алгоритма шифрования.

Обычно при проведении реальной атаки криптоаналитик не знает каких-либо элементов ключа шифрования, поэтому сформулированные выше к алгоритму предвычислений требования являются "избыточными". На самом деле даже применение одних только шифрующих преобразований с хорошими свойствами рассеивания и перемешивания является допустимым, поскольку в реальных ситуациях криптоаналитик не знает расширенного ключа. Это означает, что восполь-

зоваться тем, что шифрующие процедуры являются обратимыми, он не может (заметим, что шифрование на этапе предвычислений предполагается осуществлять по некоторому специфицированному известному ключу).

2.4. Недетерминированные программные шифры

2.4.1. Принципы построения гибких шифров

При классическом подходе криптосистема строится на основе некоторой фиксированной совокупности операций преобразований. Ключ применяется для управления процессом шифрования путем задания параметров, используемых в шифрующих преобразованиях. Этими параметрами могут служить выбираемые по определенному закону элементы ключа либо знаки псевдослучайной последовательности, генерируемой на основе ключа. Наиболее благоприятными условиями для проведения криптоаналитических атак на шифры является наличие:

- полной информации о применяемых процедурах закрытия информации;
- достаточно большого объема известного исходного текста, в том числе и специальным образом подобранного, и соответствующего ему шифра.

Общим правилом является тестирование криптосистем в таких условиях.

Использование криптоалгоритмов в программных системах защиты ЭВМ предопределяет то, что все процедуры шифрования должны быть записаны на магнитном носителе компьютера, если мы не хотим обременять пользователя постоянной необходимостью многократно выполнять дополнительные действия. Таким образом, в удобной для пользователя системе компьютерной безопасности алгоритм криптомодуля принципиально доступен злоумышленнику. Очевидно, что отсутствие информации или ее части об используемых криптографических процедурах значительно затрудняет потенциальному противнику задачу взлома системы защиты.

Использование программных средств для выполнения криптографических преобразований и этапа предвычислений, на котором происходит инициализация механизма шифрования, позволяет разработать практические криптосистемы, в которых алгоритм шифрования для конкретного пользователя не является заранее известным нападающему, потому что в таких гибких (недетерминированных) шифрах алгоритм шифрования сменяется автоматически при смене секретного ключа. Это достигается тем, что на этапе предвычислений выполняются процедуры формирования алгоритма шифрования в зависимости от секретного ключа. Таким образом, при использовании ЭВМ для усложнения криптоаналитической атаки алгоритм шифрования можно строить по некоторому порождающему алгоритму, который необходимо предполагать известным нападающему. Множество возможных алгоритмов шифрования конечно, а закон его описания (порождающий алгоритм) известен криптоаналитику. Однако последний не знает, какая

конкретная модификация алгоритма шифрования будет реализована, поскольку ее выбор задается секретным ключом.

Если число реализуемых модификаций будет составлять 10^9 и более, то задание неопределенности процедур преобразований может оказаться эффективным средством противодействия криптоаналитическим атакам. Более того, сведение задачи криптоанализа к перебору возможных вариантов вряд ли реально, так как требует предварительного рассмотрения большинства алгоритмов из данного класса, и для каждого из них нужно будет выработать индивидуальный подход. Последнее не реально вовсе. Возможен вариант криптоанализа, направленный на то, чтобы найти некоторые общие закономерности по всему множеству возможных модификаций криптоалгоритма или по достаточно большим подмножествам. Построение подобного обобщающего алгоритма криптоанализа, видимо, является существенно более трудной задачей, чем разработка метода криптоанализа шифра с фиксированным алгоритмом шифрования.

Универсальным методом криптоанализа является бесключевое чтение, основанное на использовании известных входных текстов и соответствующих им блоков шифртекста. Однако легко построить гибкий алгоритм, каждая модификация которого будет стойкой к данному обобщенному виду криптоанализа. В следующих разделах описаны гибкие алгоритмы, удовлетворяющие этому требованию. Кроме того, эти гибкие шифры являются стойкими к атаке на основе известной модификации криптоалгоритма и к атаке на основе специально выбранной модификации (в последнем случае атакующему предоставляется возможность выбрать наиболее слабый вариант криптоалгоритма из множества потенциально реализуемых модификаций).

Непосредственно с постановкой проблемы разработки криптосистем с неопределенным алгоритмом преобразований связан следующий теоретический вопрос: а не выработает ли система на этапе настройки слабый криптоалгоритм второго этапа? Трудно дать формализованное описание такого объекта как криптосхема с неопределенным алгоритмом преобразований, и тем более сложным является строгое теоретическое доказательство стойкости всех ее модификаций. Однако в пользу применения этих новых систем можно привести следующие доводы:

- вероятность генерации слабого алгоритма очень мала и резко падает при использовании разнотипных приемов задания недетерминированности процедур;
- на порождающий алгоритм можно возложить задачу контроля качества генерируемого алгоритма шифрования (например, предусмотреть на этом этапе контрольное шифрование, а затем анализ шифра на удовлетворение определенному спектральному тесту);
- в системах с многопроходным режимом шифрования в качестве одного или нескольких из составных алгоритмов можно использовать криптосхемы с фиксированными процедурами шифрования, которые допускают оценку их стойкости по отработанным методикам;

- возможна разработка криптосхем, допускающих произвольное модифицирование некоторого набора операций или процедур преобразований, т. е. криптосхем, стойкость которых не является чувствительной к модифицированию некоторой совокупности используемых операций и процедур.

Последний подход представляется наиболее перспективным для разработки практических недетерминированных шифров.

Недетерминированные шифры ставят перед криптоаналитиком принципиально новую логическую трудность — алгоритмическую неопределенность, в качестве количественной характеристики которой можно предложить число различных потенциально реализуемых модификаций алгоритма шифрования. В шифрах с фиксированным алгоритмом неопределенность процесса шифрования для криптоаналитика связана с тем, что он не знает секретного ключа, элементы которого (подключи) используются как параметры в процедурах преобразования.

С точки зрения неопределенности хода шифрующих преобразований гибкие шифры можно представить как форму задания ключевых элементов в виде настраиваемых операций и процедур шифрующих преобразований. Очевидно, что могут быть построены недетерминированные шифры, использующие только такие "функциональные" ключевые элементы, однако более целесообразна разработка гибких шифров, все модификации алгоритма шифрования которых используют также и обычный ключ шифрования, содержащий "параметрические" ключевые элементы. Особенностью использования "функциональных" ключевых элементов является задание неопределенности процедур преобразования, что сильно затрудняет общее аналитическое описание даже элементарных шагов преобразования.

При высоком уровне алгоритмической неопределенности преобразование заданного блока (или нескольких блоков) может быть описано подмножеством возможных модификаций алгоритма шифрования. С точки зрения количественной характеристики неопределенности параметрические и функциональные ключевые элементы эквивалентны, поскольку количество вариантов, по которым может идти процесс шифрования, зависит только от числа ключевых битов. Однако алгоритмическая неопределенность вводит существенную качественную особенность — сложность использования аналитических выражений при проведении криптоанализа. Неопределенность, задаваемая параметрическими ключевыми элементами, во многих случаях снимается при использовании математических формул, тогда как неопределенность, задаваемая функциональными ключевыми элементами, очень трудно поддается обобщенному рассмотрению и далеко не очевидно, как записать уравнения преобразований. Это существенно увеличивает сложность построения алгоритма криптоанализа, который позволял бы исключить переборный способ нахождения ключевых элементов.

Особенно эффективно комбинированное использование функциональных и параметрических ключевых элементов. Первые задают логическую трудность, а вторые — количественную. Основной смысл такого комбинирования заключа-

ется в том, чтобы обойти трудности задания высокого уровня алгоритмической неопределенности.

Следует отметить, что строгое рассмотрение вопроса криптостойкости недетерминированных криптосистем связано с рядом трудно поддающихся формализованному рассмотрению вопросов, из которых можно указать, например, учет различия криптостойкости каждой отдельной модификации. Допустимыми можно считать минимальные оценки, однако чрезмерно излишние допущения в пользу криптоаналитика могут привести к отклонению перспективных криптосхем. Другими важными проблемами, которые необходимо решить при разработке недетерминированных программных шифров, являются следующие:

- обеспечение высокой стойкости всех возможных модификаций криптоалгоритма к методу бесключевого чтения;
- задание большого числа неэквивалентных модификаций;
- обеспечение примерно одинаковой вероятности выбора модификации из каждого подмножества эквивалентных модификаций;
- задание стойкости к атакам на основе известной и выбранной модификации криптоалгоритма.

Рассмотрим несколько идей, на основе которых могут быть созданы недетерминированные шифры.

2.4.2. Криптосхема с перестановкой фиксированных процедур

Пусть шифрование каждого входного сообщения состоит в последовательном выполнении над ним нескольких элементарных шифрующих преобразований $\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_n$. В этом случае формирование конкретного алгоритма шифрования можно задать путем указания зависимости очередности выполнения элементарных преобразований от секретного ключа. Практически это может быть легко реализовано как формирование шифрующей программы на основе подпрограмм. Количество потенциально возможных различных алгоритмов составляет $S = n!$ и при $n = 16$ имеем $S > 10^{13}$.

2.4.3. Многопроходные криптосхемы с гибким алгоритмом

При использовании скоростных шифров есть возможность осуществления многократного шифрования исходного текста. Пусть имеется библиотека программ зашифрования $\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_n$ и соответствующих им программ расшифрования $\mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_n$. Для задания недетерминированности алгоритма шифрования можно предусмотреть реализацию вариантов процедуры шифрования, состоящих из выполнения m последовательных процедур, выбираемых из множества

$\{\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_n, \mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_n\}$, т. е. результирующее шифрование будет иметь вид суперпозиции m процедур: $\mathbf{E} = \mathbf{F}_{i_m} \circ ... \circ \mathbf{F}_{i_2} \circ \mathbf{F}_{i_1}$, где $\mathbf{F} \in \{\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_n, \mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_n\}$, а $i \in \{1, 2, ..., n\}$.

Поскольку выполнение следующих друг за другом процедур **E** и **D** с одинаковыми индексами не изменяют преобразуемого сообщения, то наложим запрет на модификации результирующей функции шифрования, соответствующие таким случаям. В качестве $\mathbf{F}_{i_1}$ может быть выбран любой из $2n$ элементов множества $\{\mathbf{E}_1, \mathbf{E}_2, ..., \mathbf{E}_n, \mathbf{D}_1, \mathbf{D}_2, ..., \mathbf{D}_n\}$, а в качестве $\mathbf{F}_{i_2}, \mathbf{F}_{i_3}, ..., \mathbf{F}_{i_{2m}}$ — один из $2n - 1$ элементов. Учитывая это замечание, легко подсчитать число возможных модификаций S результирующей функции шифрования: $S = 2n(2n - 1)^{m-1}$.

Если функция зашифрования имеет вид $\mathbf{E} = \mathbf{F}_{i_m} \circ ... \circ \mathbf{F}_{i_2} \circ \mathbf{F}_{i_1}$, то соответствующей ей результирующей функцией расшифрования будет

$$\mathbf{D} = \mathbf{F}_{i_1}^{-1} \circ \mathbf{F}_{i_2}^{-1} \circ ... \circ \mathbf{F}_{i_m}^{-1},$$

где $\mathbf{F}_i^{-1} = \mathbf{E}_j$, если $\mathbf{F}_i = \mathbf{D}_j$ и $\mathbf{F}_i^{-1} = \mathbf{D}_j$, если $\mathbf{F}_i = \mathbf{E}_j$, ($j = 1, 2, ..., n$).

Самостоятельный интерес представляют шифры, построенные на базе процедур и операций преобразования, которые зависят от преобразуемых данных. В этих шифрах на некотором данном шаге шифрования конкретный вид выполняемой операции (или процедуры) зависит от значения преобразуемого текста. При описании алгоритма шифрования на основе стандартных операций набор этих операций изменяется от одного входного текста к другому. В отличие от рассмотренных выше недетерминированных шифров, алгоритм преобразования в шифрах с операциями, зависящими от преобразуемых данных, является фиксированным, поскольку известно правило выбора конкретного вида операции от текущего значения преобразуемых данных. Использование шифрующих процедур с операциями, зависящими от входного сообщения, весьма перспективно для построения скоростных шифров, обладающих высокой криптостойкостью.

2.4.4. Криптосистема с настройкой операций преобразования

Идея использования операций преобразования в качестве ключевых элементов состоит в следующем. Задается шаблон для процедуры преобразования некоторого текущего подблока данных. В шаблоне зарезервирован набор операций преобразования, каждой из которых приписан порядковый номер. Конкретные значения операций устанавливаются на этапе предвычислений в зависимости от секретного ключа. Это может быть осуществлено по следующей схеме. Под управлением секретного ключа формируется псевдослучайная последовательность чисел. В зависимости от значения числа, соответствующего номеру операции, устанавливается конкретный вариант этой операции. Например, процедура

преобразования текущего 32-битового подблока данных T может состоять в осуществлении вычислений в соответствии со следующими формулами:

$$G := \{[(T *_{n_1} K_1)^{>>>x_1} *_{n_2} K_2)]^{>>>x_2} *_{n_3} K_3\}^{>>>x_3} *_{n_4} G$$

и

$$C := [(G^{>>>x_4} *_{n_5} K_4)^{>>>x_5} *_{n_6} K_5]^{>>>x_6} *_{n_7} K_6,$$

где C — преобразованное значение подблока данных; $K_1, K_2, \ldots, K_6$ — 32-битовые подключи, которые являются непосредственными константами в шифрующей программе или выбираются в зависимости от преобразуемых данных; G — двоичный вектор, задающий влияние предыдущих подблоков данных на преобразование текущего подблока; оператор $^{>>>x}$ обозначает зарезервированную одноместную операцию (т. е. операцию, выполняемую над одним числом, например, $W^{>>>x}$ обозначает операцию циклического сдвига вправо на x битов); знак $*_n$ обозначает двухместную операцию — сложение по модулю 2^{32} (+), вычитание по модулю 2^{32} (–) или поразрядное суммирование по модулю 2 ($\oplus$); $n_1, n_2, ..., n_7$ — номера зарезервированных двухместных операций. Конкретный набор зарезервированных операций предполагается устанавливать на этапе предвычислений в зависимости от секретного ключа.

2.4.5. Псевдовероятностный недетерминированный шифр

Представляет интерес создание шифров, в которых недетерминированный механизм преобразований непосредственно сочетается с выборкой подключей в зависимости от преобразуемых данных [93, 96]. При таком сочетании формирование слабых модификаций алгоритма шифрования предотвращается тем, что стойкость процедур шифрования с псевдослучайной выборкой подключей не является критичной к выбору конкретного набора используемых двухместных и одноместных операций преобразования. (Более того, если для любой модификации имеет место зависимость выборки подключей от каждого бита входного сообщения, то даже специальным образом подбирая наборы операций, не удастся сформировать слабую модификацию.)

Примером реализации такого подхода является следующий алгоритм, в котором используется ключ шифрования, включающий 2051 8-битовое слово: $\{q_j\}$, где $j = 0, 1, 2, ..., 2050$. 32-битовые подключи заданы соотношением $Q(x) = q_{x+3} \| q_{x+2} \| q_{x+1} \| q_x$, где $x = 0, 1, 2, ..., 2^{11} - 1$. Ключ шифрования предполагается формировать на этапе предвычислений. В качестве зарезервированных одноместных операций $W^{>>>x}$ используется операция циклического сдвига вправо слова W на $x(f)$, где $f = 1, 2, 3, ...$ — порядковый номер зарезервированной операции. Входным сообщением является 512-байтовый блок, представленный как последовательность 32-битовых слов $\{T_w\}$, $w = 0, 1, ..., 127$.

ВХОД: 512-байтовый блок данных $\{T_w\}$, $w = 0, 1, ..., 127$.

1. Установить $r = 1$, $R = 2k$ (k — натуральное число) и определить: $L_w = T_w$, $w = 0, 1, ..., 127$.
2. Установить счетчик $i := 1$ и начальные значения переменных:
   U, V, Y, G и n: $U := Q(1)$, $V := Q(2)$, $Y := Q(3)$, $G := Q(4)$,
   $n := Q(5) \bmod 2^{11}$.
3. Выполнить вычисления:
   - $n := \{[n \oplus (G \bmod 2^{11})] - U\} \bmod 2^{11}$.
   - $V := \{[V *_{n_{7r-6}} Q(n)]^{>>>x(5r-4)} + G\}^{>>>11}$.
   - $n := (n *_{n_{7r-5}} V) \bmod 2^{11}$.
   - $U := [U *_{n_{7r-4}} Q(n)]^{>>>x(5r-3)} - (G^{>>>22})$.
   - $n := (n + U) \bmod 2^{11}$.
   - $Y := [Y *_{n_{7r-3}} Q(n)]^{>>>x(5r-2)}$.
4. Вычислить индекс текущего преобразуемого подблока:
   $w = 120 - i$, если $r = 2, 4, ..., 2k$ или $w = i - 1$, если $r = 1, 3, \ldots, 2k - 1$.
5. Выполнить текущий шаг шифрования:
   - $G := (L_w *_{n_{7r-2}} V)^{>>>x(5r-1)} *_{n_{7r-1}} Y$.
   - $C_w = G^{>>>x(5r)} *_{n_{7r}} U$.
6. Сохранить значение C_w.
7. Если $i < 128$, то прирастить i и перейти к шагу 3.
8. Если $r < R$, то прирастить r, определить $L_w = C_w$, $w = 0, 1, ..., 127$, и перейти к шагу 2, в противном случае **СТОП**.

ВЫХОД: 512-байтовый блок шифртекста $\{C_w\}$, $w = 0, 1, ..., 127$.

Данный алгоритм описывает шифрующие процедуры. Соответствующие им процедуры расшифрования легко составить, поскольку они отличаются только в шагах 4 и 5. Число возможных модификаций (различных вариантов установки зарезервированных операций) этого недетерминированного алгоритма составляет $S \approx 10^{10R}$. Размер резидентной части программы шифрования для этих алгоритмов не превышает 5 Кбайт. Количество циклов шифрования данного входного блока определяется параметром R, допустимыми значениями которого являются $R \geq 3$. Скорость шифрования составляет около $300/R$ Мбит/с (для процессоров типа Pentium), а число потенциально реализуемых неэквивалентных модификаций криптоалгоритма — более 10^{10R}. Данный алгоритм является стойким при до-

пущении, что атакующему известны все операции преобразования и 90% подключей. Вариант атаки на основе известной части расширенного ключа рассмотрен ниже.

В общем случае стойкость конкретных модификаций псевдовероятностных шифров не является критически чувствительной к используемым наборам операций и их конкретным типам, поэтому применение механизма выборки подключей в зависимости от преобразуемых данных предоставляет широкие возможности варьирования и комбинирования различных типов бинарных и унарных операций преобразования. Главное достоинство этой особенности псевдовероятностных шифров состоит не столько в легкости составления широкого семейства скоростных криптоалгоритмов для программной реализации, сколько в возможности разработки многочисленных скоростных недетерминированных криптосистем, в которых для криптоаналитика алгоритм преобразования является неизвестным наравне с ключом шифрования. Для построения шифров с модифицируемым криптоалгоритмом необходимо на этапе предвычислений предусмотреть дополнительно функцию инициализации алгоритма шифрования, выполняемую под управлением секретного ключа (или пароля) пользователя. Если количество потенциально реализуемых модификаций криптоалгоритма задать очень большим (например, 10^{20}), то этот способ мотивировки стойкости преобразования будет являться весьма эффективным.

Хотя в недетерминированных шифрах предполагается неизвестной реализуемая модификация, это не противоречит принципу Керкхоффа, согласно которому алгоритм шифрования информации необходимо предполагать известным атакующему. Этот принцип является одним из важнейших при создании новых одноключевых криптосистем. Его можно представить в более общей формулировке: все долговременные элементы механизмов защиты информации необходимо считать известными потенциальному нарушителю. Например, в Российском стандарте шифрования ГОСТ 28147-89 "заполнение таблиц блока подстановки", которое "является долговременным ключевым элементом, общим для сети ЭВМ", должно рассматриваться известным атакующей стороне, хотя согласно документации оно "является секретным элементом и поставляется в установленном порядке". Использование подобных долговременных (хотя даже и сменяемых) секретных частей шифрующих систем вызывает и другие замечания. В недетерминированных шифрах долговременным элементом служат процедуры настройки алгоритма шифрования, а конкретная его модификация и ключ шифрования являются сменными элементами криптосистемы, которые автоматически заменяются одновременно со сменой паролей и выбираются уникальными для каждого пользователя (или каждой пары абонентов защищенной сети связи).

Шифры с псевдослучайной выборкой подключей являются стойкими к известным методам криптоанализа, но в будущем могут быть разработаны новые способы атаки на основе анализа статистических свойств операций и процедур преобразования или некоторых других особенностей алгоритма шифрования. Разра-

ботка недетерминированных шифров имеет своей целью минимизацию риска компрометации криптосистемы в течение длительного времени.

В последнее время криптографы уделяют внимание нападениям на криптосистемы, связанным с ожиданием или с целенаправленной генерацией аппаратных ошибок устройства шифрования. При этом принимается модель, согласно которой атакующий может оказывать на шифратор с введенным ключом некоторое внешнее воздействие с целью вызвать сбои в работе аппаратуры, причем он имеет возможность выполнить криптоанализ на основе адаптированных текстов.

Наиболее широко распространенные алгоритмы — американский стандарт DES, российский ГОСТ 28147-89, RC5 и некоторые шифры (включая криптосистемы с недетерминированным алгоритмом шифрования) с выборкой подключей в зависимости от преобразуемых данных не являются стойкими к этому методу криптоанализа. Видимо, это объясняется тем, что в то время, когда разрабатывались эти стандарты шифрования, возможность умышленного введения случайных аппаратных ошибок не принималась во внимание.

Рассмотренные далее псевдовероятностные блочные шифры являются стойкими к атакам на основе генерации случайных аппаратных ошибок в регистрах данных. Их стойкость к таким нападениям обусловлена следующими особенностями:

- номера выбираемых подключей никогда в явном виде не появляются как значение некоторой группы двоичных разрядов преобразуемых блоков. (Это не позволяет с существенной вероятностью истинности отбирать результаты опытов с ожидаемым типом ошибки);
- подключи не используются непосредственно в процедурах преобразования подблоков данных, а служат только для формирования ключевых переменных;
- в случае недетерминированных псевдовероятностных шифров сложно осуществить раздельное рассмотрение некоторых подмножеств возможных вариантов комбинаций операций преобразования, используемых в одном раунде шифрования.

2.4.6. Гибкие шифры с доказуемой неэквивалентностью модификаций криптоалгоритма

Механизм выборки подключей в зависимости от преобразуемых данных представляется той основой, которая может задать высокую стойкость преобразования по всему множеству потенциально реализуемых модификаций криптоалгоритма. Это обеспечивается тем, что каждая возможная модификация определяет уникальность выборки подключей для каждого входного сообщения. В случае недетерминированных шифров одним из важных вопросов является следующая проблема. Могут ли быть выбраны такие модификации многораундовой функции

шифрования, для которых несколько последующих раундов являются эквивалентными некоторой функции расшифрования, задающей обратные преобразования по отношению к нескольким предыдущим раундам шифрования? Этот вопрос является актуальным для шифров с формированием алгоритма преобразования по случайно выбираемому секретному ключу. Можно предположить, что для класса шифрующих функций, основанных на выборке подключей в зависимости от преобразуемых данных, такие ситуации не могут иметь места.

Доказательство этого в общем случае, видимо, является достаточно сложным, однако для частных типов гибких шифров это можно сделать. Примером является гибкий шифр, рассматриваемый в *разд. 6.7.2*. Доказательство того, что никакие модификации прямых шифрующих преобразований не сводятся к обратным в случае этого гибкого блочного 64-битового шифра, приведено в работе [32]. В данной работе показано, что для некоторого числа раундов из всех возможных модификаций алгоритма нельзя найти такую модификацию, в которой можно выделить некоторые группы раундов, задающих взаимно обратные функции преобразования. Иными словами, было доказано, что ни одна из модификаций не включает в себя какие-либо два этапа преобразований, являющиеся взаимно обратными.

Это доказательство имеет смысл как одно из свойств, свидетельствующих в пользу вывода о повышении криптостойкости при увеличении числа раундов шифрования. Для упомянутого выше 64-битового шифра ранее было доказано, что в некотором смысле все возможные модификации криптоалгоритма являются уникальными. Это положение весьма полезно для оценки того, что зависимость алгоритма шифрования от секретного ключа является эффективной в том смысле, что множество модификаций не распадается на подмножества эквивалентных криптоалгоритмов, что в принципе могло бы существенно упростить криптоанализ. Другим примером гибких криптосистем, для которых можно доказать свойства уникальности и несводимости, является 128-битовый шифр, рассмотренный в работе [95].

Следует отметить, что указанные выше специфические для гибких шифров вопросы только косвенно свидетельствуют о стойкости такого рода криптосистем. Если бы последние не удовлетворяли условиям несводимости прямых и обратных преобразований и уникальности всех шифрующих преобразований, то можно было бы говорить о недостаточной эффективности механизма задания зависимости алгоритма шифрования от секретного ключа. Но если данные условия выполняются, то вопрос об оценке стойкости гибкого шифра не снимается. Это означает, что оценку их стойкости надо выполнить в предположении известной модификации криптоалгоритма.

Возможность построения на основе механизма выборки подключей в зависимости от преобразуемых данных таких гибких криптосистем, которые обладают свойствами уникальности и несводимости, есть важная общая характеристика. Для практического использования более интересны скоростные шифры, приве-

денные ниже, хотя для них труднее доказать эти свойства ввиду более сложного механизма управления выборкой подключей. Также представляется весьма перспективным построение гибких шифров, в которых сочетается настройка по ключу как операций обычного типа, так и операций, зависящих от преобразуемых данных. Для таких гибких шифров также трудно дать формальное доказательство наличия свойств уникальности и несводимости, однако наличие этих свойств у более простых конструкций на основе выборки подключей в зависимости от преобразуемых данных есть один из элементов обоснования практического применения более сложных конструкций.

2.4.7. Комбинаторно-вероятностная модель

Стойкость описанного выше шифра основана главным образом на следующих факторах:

- для каждого входного блока выборка подключей уникальна;
- число потенциально реализуемых модификаций криптоалгоритма чрезвычайно велико;
- подключи не используются непосредственно при преобразовании слов входного сообщения.

Характер преобразований таков, что номера выбираемых при шифровании подключей образуют псевдослучайную последовательность, т. е. выборка подключей в зависимости от данных представляется псевдослучайной. Для обобщенной оценки стойкости такого типа шифров можно воспользоваться комбинаторно-вероятностной моделью (КВМ), которая выражается в следующих предположениях:

- криптосистема считается взломанной, если атакующий находит пару слов, при преобразовании которых использовался один и тот же набор значений переменных U, V и Y;
- трудоемкость нахождения такой пары слов во множестве известных или специально подобранных текстов определяет значение стойкости криптосистемы;
- в случае атаки на основе известных (КВМ-1) или специально подобранных (КВМ-2) входных текстов в раундах с номерами $r \geq 2$ (КВМ-1) или $r \geq 3$ (КВМ-2) переменные U, V, и Y принимают случайные значения.

Вероятность того, что для двух преобразованных слов совпадают использованные значения переменных, составляет $P = M^{-3(R-1)}$ (КВМ-1) и $P = M^{-3(R-2)}$ (КВМ-2), где $M \approx 2^{32}$ — число различных возможных значений, принимаемых переменными U, V и Y. С учетом "парадокса дней рождения" величина $N \approx P^{-1/2}$ задает число входных слов, для которых с вероятностью примерно 0,5 имеются два входных слова, преобразованных с использованием одинаковых значений переменных U,V и Y в раундах с номерами $r \geq 2$ (КВМ-1) или $r \geq 3$ (КВМ-2). Значение N определяет необходимый объем текстов для проведения криптоаналитической атаки.

Минимальная трудоемкость атаки $S_{\min}$ определяется перебором половины возможных сочетаний по 2 из N элементов. Легко получить $S_{\min} \approx s_r M^{3(R-1)}/4$ (для КВМ-1), $S_{\min} \approx s_r M^{3(R-2)}/4$ (для КВМ-2), где s_r равна сложности некоторого критерия распознавания повтора ($s_r \geq 1$ операций). Здесь мы полагаем, что модификация криптоалгоритма является известной атакующему. Секретность алгоритма рассматривается как дополнительная гарантия стойкости. Для тестирования фактора, связанного с секретностью набора операций преобразования, можно предложить атаку на основе специально подобранной модификации криптоалгоритма, которая состоит в том, что криптоаналитику предоставляется возможность самому выбрать удобную для него модификацию из множества потенциально реализуемых.

Псевдовероятностные шифры, подобные описанному выше, являются стойкими к атакам на основе известной части расширенного ключа, причем даже в случае известной модификации.

Рассмотрим вопрос оценки стойкости в случае, если нападающему известна доля расширенного ключа, равная Δ.

При достаточно большом значении Δ в варианте атаки на основе специально подобранных текстов (рассматриваем модель КВМ-2) атакующий может подобрать такой входной текст, для которого на первом раунде будут выбираться подключи только из известной части расширенного ключа, поэтому входной текст для второго раунда можно вычислить, но во втором и последующих раундах подключи будут выбираться по псевдослучайному закону, поэтому на каждом шаге выборки с вероятностью $1 - \Delta$ будет иметь место адресация к неизвестной части ключа. Если это произойдет до завершения преобразования нескольких слов на предпоследнем раунде, то вычислить значения подключей окажется сложной задачей, поскольку после такой ситуации становятся неизвестными номера выбираемых подключей. Вероятность того, что при преобразовании текущего слова во втором и последующих раундах будут выбираться подключи только из известной части расширенного ключа, составляет $P_1 = \Delta^3$. Вероятность выбора известных подключей на всех шагах в раундах $r = 2, 4, ..., R - 1$ составляет $P_2 = \Delta^{3e(R-2)}$, где e — число преобразуемых слов. Если P_2 составит малую величину $P_{\text{а}}$, например $P_{\text{а}} = 10^{-30}$, *то можно считать, что шифр является стойким к такому варианту атаки*. По величине $P_{\text{а}}$ можно определить безопасную известную долю расширенного ключа $\Delta_{\text{а}}$. При $e = 120$ из предыдущей формулы легко получить:

$$\Delta_{\text{а}} = \exp\frac{\ln P_{\text{а}}}{360(R-2)}.$$

В случае КВМ-1 аналогично можно получить следующую оценку:

$$\Delta_{\text{а}} = \exp\frac{\ln P_{\text{а}}}{360(R-1)}.$$

Для КВМ-2 Δ_a составляет около 0,82 (при $R = 3$) и 0,91 (при $R = 4$). Очевидно, что для одного и того же значения R безопасная доля известного ключа в случае КВМ-1 выше, чем Δ_a для КВМ-2.

2.5. Скоростные программные шифры. Обозначения и термины

При рассмотрении скоростных программных алгоритмов криптографического преобразования будем использовать следующие обозначения и термины:

- под термином "слово" будем понимать 32-битовое число, обозначаемое большими латинскими буквами; байты будем обозначать малыми латинскими буквами;
- "||" обозначает операцию конкатенации; конкатенацию двух байтов a_1 и a_2 обозначим $a_2 \| a_1$, где a_2 соответствует старшим разрядам; конкатенация четырех последовательных байтов $\{a_1, a_2, a_3, a_4\}$ представляется записью $A = a_4 \| a_3 \| a_2 \| a_1$;
- последовательность байтов $\boldsymbol{L} = \{l_0, l_1, \ldots, l_n\}$ будем интерпретировать так же, как последовательности 32-битовых слов $\boldsymbol{L} = \{L_0, L_1, ..., L_s\}$, где $L_j = \{l_{4j+3}, l_{4j+2}, l_{4j+1}, l_{4j+2}\}$ и $j = 0, 1, ..., s$; при интерпретации нескольких последовательных байтов двоичным числом правый байт относится к старшим разрядам числа; например последовательность $\boldsymbol{L}$ интерпретируется числом $l_n \| \ldots \| l_1 \| l_0$; в ряде случаев вместо записи L_i будем использовать эквивалентную запись $L[i]$, т. е. имеем $\boldsymbol{L} = \{l_0, l_1, ..., l_n\} = \{l[0], l[1], ..., l[n]\}$;
- ":=" обозначает операцию присваивания;
- "$+_n$" обозначает сложение по модулю 2^n (т. е. выражение $j := Z \bmod 2^{11}$ является эквивалентным выражению $j := Z +_{11} 0$);
- "$-_n$" обозначает вычитание по модулю 2^n;
- "$\oplus$" обозначает операцию поразрядного суммирования по модулю 2;
- "$^{>>>}$" ("$^{<<<}$") обозначает операцию циклического сдвига вправо (влево); например, циклический сдвиг слова X вправо на Y битов записывается в виде "$X^{>>>Y}$" (заметим, что только $\log_2 32 = 5$ младших битов Y используются для задания величины сдвига). Операция "$^{>>>}$" имеет более высокий приоритет ("$^{>>>}$" > "$*$", где "$*$" $\in$ {"$\otimes$", "$\oplus$", "$+_n$", "$-_n$"});
- "$W \leftrightarrow V$" обозначает операцию обмена значениями слов W и V;
- используемые шестнадцатеричные константы: F = FFFF07FF, a = 0D, P = B25D28A7 1A62D775, R = 98915E7E C8265EDF CDA31E88 F24809DD B064BDC7 285DD50D 7289F0AC 6F49DD2D.

2.6. Шифры на основе выборки подключей в зависимости от данных

Одним из направлений прикладной криптографии является разработка скоростных программных шифров, для построения которых были предложены различные подходы. Таким перспективным подходом является использование в качестве базового криптографического примитива выборки подключей в зависимости от преобразуемых данных [27, 97]. Ряд 512-байтовых блочных алгоритмов шифрования на основе такого механизма описан в работах [91, 92, 98]. Интересно, что до настоящего времени никем не предложено практически реализуемых подходов к их раскрытию. Для получения обобщенной оценки нижнего уровня стойкости шифров такого типа была предложена комбинаторно-вероятностная модель криптоанализа [91, 96]. Несмотря на эвристический характер модели, даваемые с ее помощью оценки до сих пор не были понижены. Эти факты свидетельствуют в пользу эффективности использования механизма выборки подключей в зависимости от преобразуемых данных при разработке скоростных программных шифров.

В первом приближении механизм выборки подключей в зависимости от преобразуемых данных есть не что иное, как классическая операция подстановки большого размера (от 8×16 в первых алгоритмах до 11×32 в последующих), выполняемая по секретной таблице, роль которой играет ключ шифрования. Так, например, зарубежный 64-битовый программный шифр Blowfish [121], построенный по криптосхеме Фейстеля, реализует раундовую функцию как подстановку размера 32×32 с помощью выполнения четырех табличных подстановок размера 8×32 и трех двухместных операций. Тот факт, что данный шифр признан стойким после его многолетнего публичного обсуждения, также дает общее подтверждение эффективности механизма выборки подключей в зависимости от преобразуемых данных.

При более детальном рассмотрении механизма выборки подключей в зависимости от преобразуемых данных можно установить, что в своей основной реализации он предусматривает зависимость выборки подключа не только от текущего подблока данных, но и от значений всех подблоков данных, преобразованных на предыдущих шагах. Это на первый взгляд несущественное отличие от операции табличной подстановки является принципиальным. Именно благодаря такому "продленному" влиянию преобразуемых данных рассматриваемый механизм выборки подключей не может быть описан не только простой табличной подстановкой разумного размера (например, 32×32), но он не может быть даже описан как табличная подстановка, зависящая от данных (например, подстановка размера 32×32, осуществляемая от текущего значения одного из 32-битовых подблоков данных). Именно то обстоятельство, что выборка подключей в зависимости от преобразуемых данных не может быть описана как табличная подстановка разумного размера, показывает внутренний потенциал данного криптографического примитива.

Одной из интересных реализаций шифров на основе выборки подключей в зависимости от преобразуемых данных является программный блочный алгоритм SPECTR-Z [2]. В отличие от предыдущих 512-байтовых программных алгоритмов он обладает более высокой скоростью, что было достигнуто благодаря использованию специального преобразования начальных и конечных 32-битовых подблоков входного 512-байтового блока данных. Особенностью блочных шифров с большим размером входного текста является то, что они позволяют построить такой механизм шифрования, в котором сильный лавинный эффект развивается в пределах одного раунда шифрования при последовательном шифровании подблоков данных, число которых может быть достаточно большим (например, в случае 512-байтовых шифров входной блок данных разбивается на 256 16-битовых подблоков или 128 32-битовых подблоков). Принципиально неустранимым узким местом рассматриваемого типа программных шифров является то, что зашифрование первого подблока данных в каждом раунде осуществляется с использованием фиксированных значений переменных, участвующих в процедуре шифрования. Это обстоятельство ведет к необходимости выполнить дополнительное преобразование начальных и конечных подблоков данных. В более ранних вариантах для этого осуществлялись один или два дополнительных полных раунда преобразования. В шифре SPECTR-Z дополнительное преобразование выполняется только над восемью 32-битовыми подблоками данных (4 начальных и 4 конечных) в виде сокращенного раунда из 40 шагов (5 циклов по 8 шагов) преобразования отдельных подблоков данных. Сокращенный раунд предназначен для осуществления эффективного и быстрого усиления лавинного эффекта, инициируемого изменениями битов в крайних подблоках. Действительно, сокращенный раунд выполняется в три раза быстрее, чем полный. Благодаря применению сокращенного раунда с такой структурой, процедура шифрования может быть ограничена только двумя полными раундами. Это позволило получить в случае алгоритма SPECTR-Z скорость шифрования более 140 Мбит/с для микропроцессора Pentium-II 266.

2.7. Алгоритмы шифрования в современных системах компьютерной безопасности

Современные средства защиты информации (СЗИ) от несанкционированного доступа (НСД) основаны на технологии глобального шифрования [31]. Требование обеспечения защищенности информации, циркулирующей в компьютерных системах, по отношению к многочисленным потенциальным угрозам делает необходимым применение в комплексных СЗИ НСД широкого набора системных и криптографических механизмов. Выделяются три базовых криптографических механизма:

- для поддержания режима глобального шифрования в масштабе реального времени;

- для шифрования файлов в масштабе реального времени;
- для шифрования загрузчика операционной системы.

Примером современных систем компьютерной безопасности является SPECTR-Z — программный комплекс защиты информации в ПЭВМ, нашедший широкое применение и признание в России. Система SPECTR-Z (сертификат Гостехкомиссии № 251) предназначена для защиты ЭВМ, функционирующих под управлением ОС Windows 95/98, и является новой версией полнофункционального средства защиты информации от несанкционированного доступа (СЗИ НСД) "Кобра" (сертификат № 20), получившей широкое распространение для защиты ПЭВМ, функционирующих под управлением ОС MS-DOS.

2.7.1. Скоростное шифрование дисковых данных

Критерии построения

В системах компьютерной безопасности, функционирующих в реальном масштабе времени, требуется применение скоростных алгоритмов для шифрования информации, обрабатываемой в компьютерных системах, а именно, для поддержания автоматического шифрования в режиме реального времени всех данных, хранимых в устройствах постоянной памяти. В реальных применениях алгоритм должен обеспечивать возможность осуществления динамического шифрования (зашифрование данных при записи и расшифрование при считывании с устройств постоянной памяти). При этом скорость шифрования должна быть существенно выше скорости считывания данных. Современные встроенные магнитные носители обладают скоростями записи/считывания порядка нескольких десятков Мбит/с, что делает необходимым обеспечить скорость шифрования более 100 Мбит/с при использовании серийных массовых процессоров типа Pentium.

Примером скоростного программного алгоритма шифрования является алгоритм SPECTR-Z, применяемый в одноименной системе СЗИ НСД для преобразования дисковых данных. При разработке алгоритма SPECTR-Z предполагалось, что основной областью его использования будет обеспечение внутреннего шифрования в компьютерных системах и применение в составе СЗИ НСД, сохраняющих высокую производительность компьютерных систем.

Замечание

Использование для защиты информации в телекоммуникационных каналах часто требует реализации шифров в виде крипточипов. Разработка универсальных алгоритмов, которые ориентированы как на программную, так и на аппаратную реализацию, не позволяет получить максимальную скорость шифрования для более узких применений.

Технологический характер использования данного шифра делает крайне актуальным получение максимально возможного быстродействия и высокой стойко-

сти к атакам на основе известных, а также специально подобранных текстов. С вопросом сохранения исходной высокой производительности защищаемых компьютерных систем связана также необходимость сохранения возможности произвольного доступа к данным в устройствах постоянной памяти.

В качестве размера входного блока данных для алгоритма SPECTR-Z было выбрано значение 512 байт, что соответствует минимальной дискретной порции данных в информационном обмене между оперативной и постоянной памятью. Кроме того, этот размер достаточно велик, чтобы можно было обеспечить высокую криптостойкость при низкой сложности процедур блочного шифрования. Для достижения этой стратегической цели разработчики использовали следующие критерии:

1. Процедура шифрования входного 512-байтового блока данных должна осуществляться как последовательное преобразование 128 четырехбайтовых (т. е. 32-битовых) слов с использованием операций, выполняемых за минимальное число тактов ("$+_{32}$", "$-_{32}$", "$\oplus$", "$>>>$", "$<<<$", операции обмена значениями между регистрами данных, операции выборки из оперативной памяти).
2. Каждый бит всех слов, преобразованных на предыдущих шагах шифрования, должен существенно влиять на ход преобразования всех последующих слов, т. е. последовательное преобразование слов должно осуществляться в режиме сцепления. Это будет обеспечивать сильный лавинный эффект при переходе от первых к последним словам входного блока. Благодаря тому, что в одном раунде предполагается выполнить 128 элементарных шагов преобразования 32-битовых слов, данный механизм будет играть существенную роль в обеспечении высокой стойкости при малом числе раундов шифрования.
3. Для задания эффективного режима сцепления в алгоритме шифрования необходимо предусмотреть по крайней мере две переменные, текущие значения которых будут формироваться в зависимости от их предыдущих значений и значения текущего преобразуемого слова.
4. В качестве базового механизма преобразования целесообразно использовать выборку подключей в зависимости от преобразуемых данных, прошедшую достаточную апробацию. Данный механизм представляет собой вариант операции табличной замены с использованием секретных таблиц.
5. Для усиления влияния битов преобразуемых слов на ход шифрования целесообразно некоторые операции циклического сдвига выполнять в зависимости от преобразуемых данных, которые быстро выполняются современными серийными процессорами и которые хорошо себя зарекомендовали в качестве базового криптографического примитива в шифрах RC5, RC6 и MARS.
6. Для получения возможности построения эффективного механизма выборки подключей в зависимости от преобразуемых данных размер ключа шифрования был выбран равным 2051 байт, что позволяет за одно обращение к ключу задать влияние 11 битов текущего преобразуемого слова.

7. Для формирования расширенного ключа шифрования необходимо предусмотреть процедуры преобразования секретного ключа пользователя, реализуемые как предвычисления. При этом последние должны обеспечивать сильное влияние каждого бита секретного ключа на значение расширенного. Изменение любого бита секретного ключа должно с вероятностью примерно равной 50% приводить к инвертированию каждого бита расширенного ключа.

Общая схема преобразований

В криптосистеме SPECTR-Z, также как и в других шифрах с расширенным ключом предполагается, что последний вырабатывается в зависимости от секретного ключа сравнительно небольшой длины с помощью специальной процедуры, выполняемой как предвычисления. В шифрах, ориентированных на применение в компьютерных системах в виде программных модулей, процедура предвычислений может быть достаточно сложной, поскольку ее предполагается выполнять только при инициализации (включении) криптосистемы, которую целесообразно реализовать на этапе начальной загрузки ЭВМ. Таким образом, в рассматриваемом варианте применения криптоалгоритмов никаких существенных ограничений на длительность выполнения процедур предвычислений и на используемые вычислительные ресурсы не накладывается (подпрограмма предвычислений не является резидентной). Это означает, что приемлемо использование многозвенных процедур, обладающих сильным лавинным эффектом, в частности, возможно многократное использование алгоритмов непосредственного шифрования криптосистемы SPECTR-Z, описанных в следующем разделе.

Использование предвычислений ставит атакующего перед выбором двух основных вариантов нападений:

- ❒ он может предполагать, что секретный ключ выбирается случайно, и рассматривать сравнительно простые процедуры шифрования с целью обнаружения в них тех или иных слабостей;
- ❒ он может рассматривать значительно более сложные процедуры формирования ключа шифрования с целью решения уравнения шифрования относительно секретного ключа.

Очевидно, что процедуры формирования ключа шифрования должны быть достаточно сложными, чтобы по отдельным фрагментам ключа шифрования нельзя было бы вычислить какие-либо части секретного ключа. Основными требованиями к алгоритму формирования ключа шифрования являются:

- ❒ примерно равновероятное влияние каждого бита секретного ключа на все биты формируемого расширенного ключа;
- ❒ высокая вычислительная сложность определения секретного ключа по частично (например, 50%) известному расширенному ключу.

Эти условия выполняются для приводимого ниже алгоритма предвычислений в системе SPECTR-Z.

Таким образом, программный шифр SPECTR-Z можно реализовать в виде двух подпрограмм: модуля инициализации, который предназначен для однократного запуска, и резидентного модуля шифрования, обслуживающего запросы других программ на зашифрование и расшифрование данных. Структура программного шифра с предвычислениями показана на рис. 2.1.

Рис. 2.1. Схема программного шифра с предвычислениями

В компьютерных приложениях наличие предвычислений играет дополнительную роль, которая заключается в том, что использование сложных процедур вычислений существенно затрудняют осуществление атак, связанных с подбором пароля. Действительно, проверка каждого варианта пароля требует выполнения предвычислений.

2.7.2. Предвычисления

В алгоритме SPECTR-Z используется 2051-байтовый ключ шифрования **Q**, который формируется на этапе предвычислений по секретному ключу пользователя. Расширенный ключ представляет собой упорядоченную последовательность байтов $q[i]$, а именно $\mathbf{Q} = \{q[i]\}$, где $i = 0, 1, ..., 2050$. При формировании ключа шифрования используются представленные ниже процедура ***Encrypt_Z*** и специфицированная таблица ***Z***, интерпретируемая как последовательность $\boldsymbol{Z} = \{z[i]\}$, где $i = 0, 1, ..., 2050$. Алгоритм генерации расширенного ключа состоит в выполнении процедуры ***Form_Q***, входным параметром которой является секретный ключ пользователя длины от 64 до 512 бит.

Таблица ***Z*** представляет собой последовательность из 2051 байта:

$$\{\boldsymbol{Z}', z[1024], z[1025], z[1026], z[1027], \boldsymbol{Z}''\},$$

где $z[1024] = 9A$, $z[1025] = 05$, $z[1026] = 3C$, $z[1027] = 29$, а $\boldsymbol{Z}'$ и $\boldsymbol{Z}''$ ($\boldsymbol{Z}''$ отличается от $\boldsymbol{Z}'$ только тем, что в $\boldsymbol{Z}''$ отсутствуют последние пять байтов) представлены последовательностью $\boldsymbol{G} = \{G_0, G_1, ..., G_{512}\}$, состоящей из 32-битовых слов, записанных в виде шестнадцатеричных чисел $G_k = g_{4k+3} \| g_{4k+2} \| g_{4k+1} \| g_{4k}$, где $k = 0, 1, ..., 256$:

```
59CD4F6D,   90873546,   F408639D,   B3B33D0D,
A226970F,   DD6C55E9,   4EE9E996,   E0BBD6B2,
ED569ED7,   422895FD,   F5A08568,   F260AA2E,
97CEFB13,   06D4837F,   B51D92BF,   1F1C2D64,
97F7A7CB,   7ED14440,   93DC3AE2,   3CC5A6C1,
E8CF40AB,   A75CEBD4,   F741D801,   852E14E2,
FC386A79,   136DD74A,   22E25EA8,   0E4F18E4,
2D96801C,   3C80C89C,   2587EBA7,   D5DA001D,
7C897B55,   76E1023E,   7329EC2A,   D8DF68AB,
04C87FF6,   B6249125,   B63EF3D8,   3DAE29FB,
BE73CD22,   DE96E603,   FA4A8E95,   8D5F76CA,
A69F48A7,   3AFBB8CE,   2CE563DA,   D9994EC9,
6B7DFA9F,   5A6B2CEB,   0A0B7E78,   0A4E2064,
EF964255,   D0C8F360,   55266FBB,   3A166B3E,
1FEFDEE8,   76CEB1B8,   35480947,   25704556,
70026E20,   67B492E3,   B91D054D,   1D850806,
125163DC,   D4177C68,   2BA29D5E,   E857ABCD,
69BF02D5,   7EDC842D,   6114C2C7,   2DB5A9F1,
983E70A3,   71B68CDD,   2DB7B8D9,   CF4E94B2,
0E4B408B,   E79E14A4,   FCD8328C,   0CB4816E,
A1277953,   C05BD4FE,   E28C3697,   0552C004,
D692E406,   D1ADBB1A,   5E66E661,   DF4C1388,
11905A1C,   7625AB81,   E46165AD,   F0D8FAD0,
804A9589,   A7A71158,   523A80A7,   1DD95546,
A7588D94,   960BDFD4,   152F1BE7,   1C3123BB,
EA88761F,   62ABD13A,   04A8986B,   73EC0AE6,
F78A987B,   8B34DB2F,   0C5152A3,   AC2B2612,
360320D2,   562DC9BC,   349C5922,   1D021BF7,
6155500B,   AB28C8B7,   1889F88C,   2A35806C,
D7DA49E7,   F218B83B,   39FCEE19,   FFD5F4AF,
3C81278E,   6FB41F54,   67EBC971,   6019F1F0,
D315E4C6,   91E47276,   F73D93E4,   C240B75A,
33F14370,   D2457429,   A32A5CCF,   40DDF94E,
80E47D16,   B2C7BD8E,   90BA4275,   6C7922AB,
B2A51B55,   E6250E32,   D9971E0F,   D70EF0DB,
685779AA,   05674EC3,   393D6C1B,   5A3CCDEA,
2374D804,   F3659EDD,   5BF178CD,   04AE819B,
F984DD85,   81FCA05D,   8548F8D7,   FCF546F9,
F7CC93BE,   EFC9298C,   8F8C98D0,   756C4F45,
F14223E1,   86FEF671,   2B112DA4,   331C975E,
1BBCA16D,   8D695847,   CC6097F7,   E914DF4F,
1B97C93D,   DB12475D,   57BBFFFC,   4D7A5833,
AECFB8B9,   8C8CB213,   F7A9E9D9,   C4AADB49,
```

```
950D7683,   C68375EA,   9E4A232C,   2EF4700E,
27B2A314,   A01D80B1,   5161633B,   2954D4C4,
B494D1AF,   1420879E,   EEAE1361,   0F183EB0,
4973279C,   E7A80D21,   E5671416,   86D8CE7A,
D247FA63,   D7AF44C2,   87443304,   BBBE9F57,
202C07FD,   875FF05A,   43CC216C,   06350407,
9FA206BB,   95319458,   AC8F2486,   E822D6DB,
0DC11C62,   B5479520,   45C446D7,   85938591,
948F0F43,   839F1C97,   D407CB1A,   A3537935,
E7DB9374,   3925653F,   D15A2D07,   97530098,
7FD905D0,   7CCCDFBC,   77AAF70E,   833676F2,
BEDF2436,   F6C26720,   D866E30C,   DFB940E9,
785E5336,   1C42407B,   0FCE38E6,   28E21B1A,
5D7DEB36,   FB3C8B16,   24B5C173,   28BC5727,
6ED16186,   BDDC35F7,   3B715F42,   C1A60EEA,
1B6D0DC3,   A43393D5,   59D046BA,   2799B4B6,
206DAD0A,   A37D7588,   17F4653D,   E98EDD7F,
DC941394,   6F81AD8E,   7611F662,   CA191B6C,
BDD491CF,   C6910F8F,   C94137C2,   40D5E6D5,
D0C4D719,   5760AA69,   6F819234,   F8E83558,
A5A482C9,   520DCAB6,   92E24E59,   8D1B7005,   9A053C29
```

Элементы последовательности $\boldsymbol{Z}$ выражаются через элементы последовательности $\boldsymbol{Z}'$ так: $z[j] = z[1028+j] = z'[j]$ при $0 \le j \le 1023$. В свою очередь элементы последовательности $\boldsymbol{Z}'$ выражаются через элементы последовательности $\boldsymbol{G}$ следующим образом: $z'[j] = g_j$.

Процедура *Form_Q*

Процедура ***Form_Q*** состоит из следующих шагов:

1. Повторить секретный ключ пользователя необходимое число раз до получения 2051-байтовой последовательности $\{p[i]\}$, где $i = 0, 1, ..., 2050$, разделяя каждую повторную запись байтом со значением 0.
2. Формировать последовательность $\boldsymbol{H} = \{h_0, h_1, ..., h_{2050}\}$, где $h_i = z_i \oplus p_i$ для $i = 0, 1, ..., 2050$, а z_i являются элементами последовательности $\boldsymbol{Z}$, описанной выше.
3. Формировать последовательность $\boldsymbol{R} = \{r_0, r_1, ..., r_{511}\}$, где $r_i = p_i$ для $i = 0, 1, ..., 511$.
4. Используя последовательность байтов $\boldsymbol{H}$ в качестве ключа, с помощью процедуры ***Encrypt_Z*** осуществить преобразования:

   $\boldsymbol{R}^{(0)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{R})$; $\quad \boldsymbol{R}^{(1)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{R}^{(0)})$;

   $\boldsymbol{R}^{(2)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{R}^{(1)})$; $\quad \boldsymbol{R}^{(3)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{R}^{(2)})$.

5. Сформировать 2051-байтовую последовательность:
   $\boldsymbol{L} = \{\boldsymbol{R}^{(0)}, \boldsymbol{R}^{(1)}, \boldsymbol{R}^{(2)}, \boldsymbol{R}^{(3)}, r[0], r[1], r[2]\}$.
6. Используя последовательность байтов $\boldsymbol{L}$ в качестве ключа, с помощью процедуры ***Encrypt_Z*** осуществить преобразования:
   $\boldsymbol{C}^{(0)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{R})$; $\boldsymbol{C}^{(1)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{C}^{(0)})$;
   $\boldsymbol{C}^{(2)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{C}^{(1)})$; $\boldsymbol{C}^{(3)} := \boldsymbol{Encrypt_Z}\,(\boldsymbol{C}^{(2)})$.
7. Сформировать 2051-байтовый расширенный ключ $\boldsymbol{Q}$:
   $\boldsymbol{Q} = \{\boldsymbol{C}^{(0)}, \boldsymbol{C}^{(1)}, \boldsymbol{C}^{(2)}, \boldsymbol{C}^{(3)}, c[0], c[1], c[2]\}$,
   где $c[0]$, $c[1]$, $c[2]$ — первые три байта последовательности $\boldsymbol{C}^{(0)}$.
8. **СТОП**.

В результате осуществления процедуры преобразований получаем ключ $\boldsymbol{Q}$, который в дальнейшем используется для шифрования 512-байтовых блоков данных (например, секторов жесткого диска в ЭВМ). Ключ шифрования $\boldsymbol{Q}$ рассматривается как упорядоченная последовательность байтов $q[i]$, а именно: $\boldsymbol{Q} = \{q[i]\}$, где $i = 0, 1, ..., 2050$. При шифровании используются подключи $Q[j]$, где $j = 0, 1, ..., 2047$ и $Q[j] = q[j+3] \| q[j+2] \| q[j+1] \| q[j]$.

2.7.3. Алгоритмы дискового шифрования

Алгоритм шифрования SPECTR-Z включает три раунда. Первый и третий раунды являются полными и соответствуют последовательному преобразованию всех 128 слов входного блока, начиная с первого слова. Второй раунд является сокращенным и соответствует преобразованию сокращенного блока данных, сформированного объединением 4-х начальных и 4-х конечных 32-битовых слов 512-байтового блока данных (рис. 2.2).

Рис. 2.2. Последовательность преобразования слов

Сокращенный раунд выполняется в виде 5 циклов шифрования 256-битового сокращенного блока данных. Каждый из циклов включает 8 шагов преобразования

32-битовых слов, которые выполняются аналогично шагам преобразования в полном раунде. Второй раунд состоит в последовательном преобразовании следующих слов: T_{124}, T_{125}, T_{126}, T_{127}, T_0, T_1, T_2, T_3. После каждого, кроме последнего, цикла шифрования указанных восьми слов осуществляется обмен значениями следующих пар слов: $T_0 \leftrightarrow T_{127}$, $T_1 \leftrightarrow T_{126}$, $T_2 \leftrightarrow T_{125}$, $T_3 \leftrightarrow T_{124}$. Полный раунд шифрования состоит в следующем. Блок открытого текста ***T*** разбивается на 128 32-битовых слов T_i: $T = \{T_i\}$, $i = 0, 1, ..., 127$. Слова преобразуемого блока данных преобразуются в следующей последовательности: $T_0, T_1, ..., T_{127}$. Процедура зашифрования блока данных ***Encrypt_Z*** приведена ниже.

Процедура *Encrypt_Z* (зашифрование 512-байтового блока данных)

Процедура ***Encrypt_Z*** состоит из следующих шагов:

1. Выполнить полный раунд зашифрования.
2. Выполнить сокращенный раунд зашифрования.
3. Выполнить полный раунд зашифрования.

Полный раунд зашифрования

1. Установить значение счетчика $i := 0$ и вычислить начальные значения внутренних переменных $R := Q[9]$, $V := Q[7]$, $Y := Q[3]$, $U := Q[9]$, $N := Q[5]$, $n := N +_{11} 0$.
2. Осуществить преобразования:
   - $N := n \oplus R$; $V := V +_{32} N$;
   - $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>11>}$;
   - $N := n \oplus V$; $Y := Y +_{32} N$;
   - $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>11>}$;
   - $N := n +_{32} Y$; $n := N +_{11} 0$;
   - $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$; $R := 0T_i$.
3. Осуществить преобразование очередного слова текста:

   $T_i := ((T_i -_{32} V) \oplus U)^{<<<V} -_{32} Y$.
4. Прирастить $i := i + 1$. Если $i \neq 128$, то перейти к шагу 2, иначе **СТОП**.

Сокращенный раунд зашифрования

1. Установить значение внешнего счетчика $j := 5$.
2. Установить значение внутреннего счетчика i:=124 и вычислить начальные значения внутренних переменных $R := Q[9]$, $V := Q[7]$, $Y := Q[3]$, $U := Q[9]$, $N := Q[5]$, $n := N +_{11} 0$.

3. Осуществить преобразования:
   - $N := n \oplus R$; $V := V +_{32} N$;
   - $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>>>11}$;
   - $N := n \oplus V$; $Y := Y +_{32} N$;
   - $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>>>11}$;
   - $N := n +_{32} Y$; $n := N +_{11} 0$;
   - $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$; $R := T_i$.
4. Осуществить преобразование очередного слова текста:
   $T_i := ((T_i -_{32} Y)^{>>>V} \oplus U) -_{32} V$.
5. Прирастить $i := i + 1 \bmod 128$. Если $i \neq 4$, то перейти к шагу 3.
6. Уменьшить $j := j - 1$. Если $j = 0$, то **СТОП**.
7. Обменять значения начальных и конечных слов текста:
   $T_0 \leftrightarrow T_{127}$, $T_1 \leftrightarrow T_{126}$, $T_2 \leftrightarrow T_{125}$, $T_3 \leftrightarrow T_{124}$.
8. Перейти к шагу 2.

Расшифрование выполняется с помощью процедуры ***Decrypt_Z***.

Процедура *Decrypt_Z*

Процедура ***Decrypt_Z*** состоит из следующих шагов:

1. Выполнить полный раунд расшифрования.
2. Выполнить сокращенный раунд расшифрования.
3. Выполнить полный раунд расшифрования.

Полный раунд расшифрования

1. Установить значение счетчика $i := 0$ и вычислить начальные значения внутренних переменных $R := Q[9]$, $V := Q[7]$, $Y := Q[3]$, $U := Q[9]$, $N := Q[5]$, $n := N +_{11} 0$.
2. Осуществить преобразования:
   - $N := n \oplus R$; $V := V + N$;
   - $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>11>}$;
   - $N := n \oplus V$; $Y := Y +_{32} N$;
   - $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>11>}$;
   - $N := n +_{32} Y$; $n := N +_{11} 0$;
   - $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$.

3. Осуществить преобразование очередного слова текста:

   $T_i := ((T_i +_{32} Y)^{>>>V} \oplus U) +_{32} V.$

4. Присвоить значение $R := T_i$ и прирастить $i := i + 1$. Если $i \neq 128$, то перейти к шагу 2, иначе **СТОП**.

Сокращенный раунд расшифрования

1. Установить значение внешнего счетчика $j := 5$.
2. Установить значение внутреннего счетчика $i := 124$ и вычислить начальные значения внутренних переменных $R := Q[9]$, $V := Q[7]$, $Y := Q[3]$, $U := Q[9]$, $N := Q[5]$, $n := N +_{11} 0$.
3. Осуществить преобразования:
   - $N := n \oplus R$; $V := V +_{32} N$;
   - $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>>>11}$;
   - $N := n \oplus V$; $Y := Y +_{32} N$;
   - $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>>>11}$;
   - $N := n +_{32} Y$; $n := N +_{11} 0$;
   - $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$.
4. Осуществить преобразование очередного слова текста:

   $T_i := ((T_i +_{32} V) \oplus U)^{<<<V} +_{32} Y.$

5. Присвоить значение $R := T_i$ и прирастить $i := i + 1 \bmod 128$. Если $i \neq 4$, то перейти к шагу 3.
6. Уменьшить $j := j - 1$. Если $j = 0$, то **СТОП**.
7. Обменять значения начальных и конечных слов текста:

   $T_0 \leftrightarrow T_{127}$, $T_1 \leftrightarrow T_{126}$, $T_2 \leftrightarrow T_{125}$, $T_3 \leftrightarrow T_{124}$.

8. Перейти к шагу 2.

Отметим, что структуры процедур зашифрования и расшифрования идентичны. В обоих случаях, например, преобразование слов в полных раундах осуществляется, начиная со слова T_0, в очередности T_0, T_1, T_2, ..., T_{127}. По аналогии с тем, что подключи расширенного ключа связаны сложной функциональной зависимостью с элементами секретного ключа, значения "аккумулирующих" переменных являются функцией расширенного ключа и подблоков, преобразованных на предыдущих шагах. Такой механизм определяет тот факт, что подключи $Q[j]$ не используются в уравнении непосредственного преобразования слов входного блока (шаг 3 алгоритма полного раунда и шаг 3 алгоритма сокращенного раунда). Кроме того, в непосредственном преобразовании слов входного блока в каждом раунде участвуют три "аккумулирующих" переменных, которые принимают псевдослучайные значения. Эти факторы определяют эффективность преобразований

шифра SPECTR-Z, который сочетает в себе высокую стойкость и высокое быстродействие.

Если рассмотреть шифрование двух текстов, различающихся только в слове T_i, где $i < 127$, то легко видеть, что при преобразовании слова T_{i+1} уже в первом раунде, по крайней мере, значение одной из переменных U, V, или Y изменяется, что приводит к лавинообразному изменению параметров U, V, Y и n на следующих элементарных шагах преобразования.

Если два входных блока отличаются в последнем слове T_{127} входного блока, то лавинное нарастание влияния этого изменения начнется с шага, соответствующего шифрованию слова T_0 в первом цикле второго раунда.

Таким образом, особенностью шифра SPECTR-Z является то, что сильный лавинный эффект имеет место при переходе от одного преобразуемого слова к другому в пределах одного раунда шифрования. Сам раунд шифрования представляет собой последовательное повторение некоторой итеративной функции шифрования над всеми словами входного сообщения. Сокращенный раунд шифрования используется для того, чтобы обеспечить сильное рассеивающее влияние битов последнего 32-битового слова на шифртекст.

Схема программного шифра, показанная на рис. 2.1, применительно к алгоритму SPECTR-Z может быть более подробно представлена на рис. 2.3, где выделен механизм формирования "аккумулирующих" переменных, который может быть интерпретирован как автомат, обладающий, по крайней мере, 2^{107} различными состояниями, определяемыми четверкой значений переменных U, V, Y и n.

Первые три из этих переменных непосредственно участвуют в преобразовании слов входного текста, а последняя является внутренним параметром, который опосредованно влияет на ход процесса шифрования. Благодаря большому числу внутренних состояний этого автомата, использованию нескольких раундов шифрования и зависимости перехода из одного состояния в другое от преобразуемых слов, обеспечивается уникальная последовательность переходов для каждого входного сообщения. Отметим, что полный раунд шифрования представляет собой некоторый механизм поточного шифрования с использованием влияния предыдущих слов на процедуру шифрования последующих.

Важная особенность использованного механизма сцепления состоит в том, что он осуществляется с помощью четырех параллельных механизмов, а именно, через переменные U, V, Y и n. Последним определяется существенность влияния режима сцепления и вычислительная неосуществимость нахождения (без знания ключа) последовательностей преобразуемых слов, которые бы могли перевести автомат из заданного исходного состояния за несколько шагов преобразования в заданное конечное состояние.

Рис. 2.3. Обобщенная структура преобразований в шифре SPECTR-Z

2.7.4. Оценка стойкости

Криптоалгоритм SPECTR-Z является стойким к известным аналитическим атакам, включая линейный и дифференциальный криптоанализ. Стойкость криптоалгоритма обеспечивается не количеством раундов, а самой структурой процедур преобразования и большим числом преобразуемых слов в двух основных раундах шифрования. При разработке шифра SPECTR-Z для сопоставительной оценки стойкости различных вариантов криптосхем с псевдослучайной выборкой подключей была использована атака на основе аппаратных ошибок, впервые подробно описанная в работе [46]. При этом рассматривался вариант случайных ошибок, т. е. предполагалось, что атакующий не может сформировать ошибку, имеющую нужное значение (не может инвертировать биты в заданных разрядах). Кроме того, полагалось, что от введения ошибки в значение параметра заверше-

ния процедуры шифрования входного текста защита должна обеспечиваться дополнительными механизмами уже при конкретной реализации алгоритма, например, путем дублирования условия выхода из процедуры шифрования. Это замечание является общим для всех известных алгоритмов шифрования блочного типа.

Криптоалгоритм SPECTR-Z также является стойким к атакам, основанным на генерации случайных аппаратных ошибок, которые преднамеренно вводятся атакующим в регистры, содержащие значения преобразуемых подблоков данных, путем внешнего физического воздействия на устройство шифрования. По нашим оценкам его стойкость к такой атаке составляет не менее 10^{30} операций, тогда как многие другие широко используемые шифры (RC5, DES, ГОСТ 28147-89) обеспечивают вычислительную стойкость к данной атаке не более 10^9 операций. Достаточно высокая стойкость к данному виду анализа достигается за счет следующих факторов:

- в шифртексте значения номеров выбираемых подключей никогда не присутствуют (номера формируются как значения внутренней переменной механизма шифрования);
- ошибка, вводимая на последнем раунде в какое-либо слово, размножается механизмом формирования переменных U, V, Y и n, входя в каждую из последних, причем изменения в n приводят к изменению в выборке подключей; данные изменения трудно дифференцировать в их интегральном влиянии на следующее слово;
- переменная U, формируемая с использованием последней выборки подключа, не входит в преобразование слов входного текста как операнд последней операции, что существенно размывает статистику разностей подключей, поскольку на нее накладывается значение переменной Y (при зашифровании) или V (при расшифровании), которое является неизвестным;
- выявление неоднородностей в статистике разностей между значениями подключей существенно усложнено также и тем, что после наложения подключа на исходное значение переменной U, на нее накладывается значение R, после чего осуществляется операция циклического сдвига на число битов, которое атакующему не известно.

Для получения обобщенной оценки минимального уровня криптостойкости шифра SPECTR-Z воспользуемся комбинаторно-вероятностной моделью, использованной ранее для блочных шифров с аналогичной структурой преобразований [91]. В алгоритме SPECTR-Z переменные U, V, Y и n изменяются при переходе от одного преобразуемого слова к другому, принимая значения, образующие псевдослучайную последовательность. Значения переменных U, V и Y на шаге преобразования i-го слова формируются в зависимости от комбинаций из i подключей ($i = 1, 2, ..., 128$), причем только число различных возможных значений M_i этих переменных зависит от i (для данного i значения U, V и Y зависят от входного блока). Легко показать, что $M_0 = 1 = 2^0$, $M_1 \approx 2^{11}$, $M_2 \approx 2^{22}$ и $M_i \approx 2^{32}$ для

$i = 3, 4, ..., 128$. Таким образом, на шагах с номерами $i = 3, 4, ..., 128$ возможно примерно 2^{96} различных наборов $\{Y, U, V\}$. Реализация конкретного набора зависит от входного блока и ключа шифрования. Рассматриваемый алгоритм разработан в соответствии со следующим критерием — *процедуры преобразования должны быть составлены так, чтобы изменение любого бита входного сообщения приводило к изменению выборки подключей*. Этот критерий гарантирует, что для всех различных входных сообщений будут генерироваться уникальные последовательности наборов $\{Y_{(i)}, U_{(i)}, V_{(i)}\}$, где индексом (i) отмечены значения переменных на шагах, соответствующих преобразованию слова с номером i.

Схематично процедуры одного раунда шифрования можно представить так — в зависимости от значений текущих преобразуемых входных слов осуществляется выборка подключей, применяемых для формирования значения переменных U, V и Y. Эти переменные используются для преобразования следующих слов. Алгоритм составлен таким образом, что изменение любого бита в произвольном входном блоке данных приведет к изменению выборки подключей и изменению значений ключевых переменных, а следовательно, каждое входное сообщение преобразуется с помощью уникальных последовательностей значений переменных U, V и Y. При этом в первом и, частично, втором раунде (в первом цикле) на некоторых шагах преобразования для специально подобранных входных сообщений значения U, V и Y могут совпадать предопределенно. Для этого надо выбрать два входных сообщения, отличающиеся только в слове T_{127}. Что касается последних четырех циклов второго раунда и всех шагов третьего раунда, то значения переменных для любой пары входных сообщений могут совпасть только случайно.

Определение значений подключей $Q[j]$ связано с нахождением значений U, V и Y на определенных шагах преобразования (например, на двух ближайших последовательных шагах преобразования слов), поэтому фиксирование значений U, V и Y на определенных шагах является предварительным условием вычисления подключей $Q[j]$. Под *фиксированием* будем понимать нахождение таких слов в одном или различных входных сообщениях, которые были преобразованы с использованием значений U, V и Y, связанных каким-либо условием. Например, для двух различных слов T и T', соответствующие им пары значений каждой из "аккумулирующих" переменных равны ($U = U'$, $V = V'$ и $Y = Y'$), составляют заданную разность ($U \oplus U' = \text{const}_1$, $V \oplus V' = \text{const}_2$ и $Y \oplus Y' = \text{const}_3$) или связаны линейным соотношением. Заметим, понятие фиксирования включает в себя любые наперед заданные зависимости между значениями "аккумулирующих" переменных, соответствующих выбираемой паре слов, т. е. применение дифференциального и линейного криптоанализа охватывается комбинаторно-вероятностной моделью как частный случай.

Уравнение, описывающее шифрование отдельного слова T (принадлежащего средней части входного блока данных, т. е. исключая 4 начальных и 4 конечных слова), в общем виде задается выражением: $C = f(T, U_1, V_1, Y_1, U_3, V_3, Y_3)$, где индекс соответствует номеру раунда шифрования. Анализ экспериментальной ста-

тистики последовательности значений, принимаемых переменными U, V и Y, подтверждает, что эти переменные принимают псевдослучайные значения.

Для получения обобщенных минимальных оценок сделаем ряд допущений, которых в реальных условиях атакующий не имеет. Будем предполагать, что решение системы уравнений, соответствующих преобразованию двух отдельных слов, является проблемой с низкой сложностью, если эти слова преобразованы с помощью фиксированных наборов значений "аккумулирующих" переменных U_1, V_1, Y_1, U_3, V_3, Y_3. Будем полагать, что сложность криптоанализа обусловлена только выявлением пары слов, которые соответствуют условию фиксирования.

Это соответствует достаточно общему принципу нападения на шифры — распознаванию (при атаке на основе известного исходного текста) или заданию (при атаке на основе подобранных текстов) некоторых ожидаемых связей между неизвестными параметрами. При этом поиск пар слов, удовлетворяющих условию фиксирования, можно охарактеризовать с некоторых достаточно общих позиций:

1. Атакующий должен выработать некоторый критерий для распознавания слов, удовлетворяющих условию фиксирования. Такой критерий может быть связан с использованием статистики некоторых вычислений, осуществляемых на основе какого-то предположения. Будем полагать, что атакующий обладает простым и эффективным критерием, который для двух заданных слов позволяет с минимальной трудоемкостью (за одну условную операцию) установить, соответствуют ли эти слова условию фиксирования.
2. Атакующий заранее не может знать номера слов, которые с высокой вероятностью соответствуют принятому условию фиксирования, но он может выбирать для опробования слова с определенными номерами, для которых условие фиксирования может выполняться с более высокой вероятностью.
3. С целью поиска пары слов, соответствующих условию фиксирования, атакующий осуществляет опробование пар слов, соответствующих повышенной вероятности появления условия фиксирования.

При атаке на основе известных текстов для опробуемых пар слов T и T' значения U_1, V_1, Y_1, U_3, V_3, Y_3 в общем случае различны. В случае атаки на основе специально подобранных текстов можно подбирать такие пары входных блоков, для которых на каждом заданном шаге первого раунда формируются одинаковые значения переменных U_1, V_1, Y_1 (эти переменные будут изменяться при переходе от одного слова к другому, но в первом раунде для таких пар входных блоков это будет происходить синхронно). Подобные пары входных сообщений легко получаются путем модифицирования в $\boldsymbol{T}$ слова T_i, где $0 < i < 127$. В полученной таким образом паре текстов $\boldsymbol{T}$ и $\boldsymbol{T}'$ слова T_j, где $0 \le j \le i$, в первом раунде будут преобразовываться с помощью одинаковых значений U_1, V_1, Y_1. Однако, после выполнения сокращенного раунда любое изменение в $\boldsymbol{T}'$ приведет к изменению всех битов слов T_0, T_1, T_2, T_3, T_{124}, T_{125}, T_{126}, T_{127} с вероятностью 0,5. Ход шифрования

в третьем раунде будет связан с псевдослучайными значениями разностей $U_{3(i)} - U_{3(i)}'$, $V_{3(i)} - V_{3(i)}'$, $Y_{3(i)} - Y_{3(i)}'$. В паре текстов $\boldsymbol{T}$ и $\boldsymbol{T}'$ с наибольшей пользой для атакующего может быть использован факт повторения в первом раунде значений U_1, V_1, Y_1 для слов T_j и T'_j, где $5 \le j \le 123$. (Рассмотрение четырех начальных и четырех конечных слов для анализа менее эффективно, поскольку начальные и конечные слова дополнительно преобразуются во втором раунде.)

С учетом этого, имеем два варианта обобщенной комбинаторно-вероятностной модели (КВМ):

- для атаки на основе известных исходных текстов (КВМ-1);
- для атаки на основе специально подобранных текстов (входных текстов или шифртекстов) (КВМ-2).

Отметим, что во втором варианте сложность криптоанализа в случае подобранных шифртекстов имеет примерно то же значение, что и в случае подобранных исходных текстов, поскольку алгоритмы зашифрования и расшифрования практически идентичны. КВМ-1 и КВМ-2 отличаются вероятностью того, что произвольные два слова были преобразованы с использованием одинаковых значений "аккумулирующих" переменных. Такую пару слов будем называть искомой. Данная вероятность для КВМ-1: $p_1 = M^{-6}$, а для КВМ-2 имеем $p_2 = M^{-3}$, где $M = 2^{-32}$ — число различных значений, принимаемых переменными U, V, Y.

Если имеется некоторый объем известных или подобранных текстов, в которых содержится L слов (относящихся к словам с индексами $i = 4, 5, ..., 123$), то можно выделить C_L^2 различных их пар, которые в первом приближении можно считать независимыми. С учетом последнего получаем вероятность того, что среди данных L слов найдется искомая пара слов, определяемая как $P = 1 - A_M^L M^{-L}$. Для случая $P < 0{,}6$ можно воспользоваться приближенными формулами $P_1 \approx p_1 C_L^2$ для КВМ-1 и $P_2 \approx p_2 C_L^2$ для КВМ-2. Для $P_1 = P_2 = 0{,}5$ легко получить значения L_1 и L_2, которые можно принять за минимальное количество известных или подобранных слов, и среди них с вероятностью 0,5 найдется искомая пара слов: $L_1 \approx p_1^{-0,5}$ и $L_2 \approx p_2^{-0,5}$. Эти значения соответствуют числу различных зашифрованных входных блоков данных, равному $L_1' > 2^{-7} L_1$ для КВМ-1 и $L_2' > 2^{-7} L_2$ для КВМ-2. Если такой объем материала доступен криптоаналитику, то он может, применяя критерий распознавания повторов, найти искомую пару.

Сложность поиска этой пары W мы примем за сложность криптоанализа, т. е. за стойкость алгоритма. На самом деле рассмотрение отдельных пар слов не может обеспечить идентификацию повтора, но в нашей модели мы принимаем это допущение в пользу атакующего, поскольку нас интересует нижняя оценка стойкости алгоритма. Будем полагать, что атакующий имеет какое-то дополнительное условие, которое он проверяет для различных пар слов, выбираемых им случайно. Чтобы найти искомую пару слов с вероятностью 0,5, необходимо проверить половину возможных сочетаний C_L^2. Полагая, что одну проверку атакующий

осуществляет за одну операцию, легко получить сложность криптоанализа: $W_1 = 0{,}5p_1^{-1}$ и $W_2 = 0{,}5p_2^{-1}$.

Численные значения обобщенных оценок приведены в табл. 2.1. Полученные результаты показывают, что криптоаналитик в лучшем случае может вычислить ключ шифрования, если он будет обладать подобранными текстами объемом более, чем 2^{18} Гбайт, и соответствующими им шифртекстами. При этом он должен выполнить не менее 2^{95} операций. Данные оценки показывают, что алгоритм SPECTR-Z может быть применен для безопасного зашифрования всех данных на современных встраиваемых носителях информации большого объема.

Таблица 2.1. *Численные значения обобщенных оценок криптоанализа*

Вариант атаки	P	*L*, слов	L′, блоков	*W*, операций
КВМ-1	2^{-192}	2^{96}	$>2^{89}$	2^{191}
КВМ-2	2^{-96}	2^{48}	$>2^{41}$	2^{95}

КВМ дает достаточно хорошую обобщенную оценку с точки зрения разработчика, поскольку не завышает стойкости. Сопоставление результатов, полученных по КВМ-1, с результатами специальных криптоаналитических исследований, выполненных для алгоритма системы "Кобра", подтверждает это. Действительно, экспертный анализ с помощью известных методов криптоанализа дал значения $L \approx 6 \cdot 10^{13}$ байт и $W \approx 4 \cdot 10^{15}$ операций, тогда как КВМ-1 дает величины существенно меньшие: $L \approx 10^{7}$ байт и $W \approx 10^{14}$ операций. Этот факт и то, что для шифра SPECTR-Z не было найдено вариантов криптоанализа, дающих значение стойкости ниже величины, определяемой по КВМ, позволяет допустить принятие оценки, получаемой по КВМ.

2.7.5. Алгоритм файлового шифрования

Для осуществления файлового шифрования в криптосистеме SPECTR-Z использован механизм преобразования, который удовлетворяет следующим требованиям:

- выполнение шифрующих процедур в масштабе реального времени при их программной реализации;
- формирование по секретному ключу и случайно формируемой метке файла временного уникального ключа файлового шифрования;
- обеспечение возможности произвольного доступа ко всем байтам зашифрованного файла (независимое шифрование каждого байта);
- осуществление файлового шифрования при каждой процедуре записи файла;
- неизменность длины и системных атрибутов файла.

Ключ файлового шифрования представляет собой некоторый расширенный ключ, формируемый по секретному ключу. При этом используется алгоритм предвычислений, который не допускает вычисления секретного ключа даже при условии, что расширенный ключ известен. Секретный ключ является общим для шифрования всех файлов данного пользователя и формируется на этапе предвычисления при начальной загрузке ЭВМ. Ключ имеет размер 1024 байта и представляет собой совокупность 256 32-битовых подключей $Q[j]$, $j = 0, 1, ..., 255$.

Общая схема шифрования файла состоит в выработке восьмибитовых элементов ключевой гаммы в зависимости от порядкового номера байта в файле и дополнительного локального ключа с последующим наложением элементов гаммы на соответствующие байты файла. Длина локального ключа составляет 64 бита. Локальный ключ формируется при открытии файла в зависимости от секретного ключа файлового шифрования и 64-битовой метки, предполагаемой известной нарушителю. Данная метка генерируется по случайному закону и присваивается каждому файлу (в качестве одного из атрибутов), для которого назначается режим шифрования. При этом вероятность формирования одинаковых меток для различных файлов пренебрежимо мала (порядка 2^{-64}) .

Пусть $M = U \| V$ — 64-битовая метка, представленная в виде конкатенации двух 32-битовых слов $U = u_4 \| u_3 \| u_2 \| u_1$ и $V = v_4 \| v_3 \| v_2 \| v_1$. Ниже приводится алгоритм формирования локального ключа.

Формирование локального ключа

Алгоритм формирования локального ключа состоит из следующих шагов:

1. Установить счетчик $i = 1$.
2. Выполнить преобразования:
   - $U := U + Q[v_1]$; $V := V + Q[u_1]$;
   - $U := U \oplus Q[v_2]$; $V := V \oplus Q[u_2]$;
   - $U := (U + V)^{<16<}$; $V := V + Q[u_1]$;
   - $V := V^{<16<}$; $U := U + Q[v_1]$;
   - $V := V \oplus Q[u_2]$; $U := (U \oplus Q[v_2])^{<13<}$;
   - $V := (V + U)^{<13<}$.
3. Если $i < 5$, то прирастить $i := i + 1$ и перейти к шагу 2.
4. **СТОП**.

Значение $R = U \| V$ после выполнения пяти раундов преобразования является локальным ключом. Шифрование файла осуществляется в соответствии с приводимым ниже алгоритмом, в котором использованы следующие обозначения: $N = n_4 \| n_3 \| n_2 \| n_1$ — номер текущего преобразуемого байта, представленный в виде конкатенации восьмибитовых чисел, $R = r_8 \| r_7 \| r_6 \| r_5 \| r_4 \| r_3 \| r_2 \| r_1$ —

локальный ключ, представленный в виде конкатенации восьмибитовых подключей, $F = f_4 \| f_3 \| f_2 \| f_1$ — 32-битовая переменная, j — восьмибитовый номер выбираемого подключа.

Шифрование текущего байта

Алгоритм шифрования текущего байта шифруемого файла состоит из следующих шагов:

1. Выполнить преобразования:
   - $j := n_1 + r_5$; $F := Q[j] \oplus (r_4 \| r_3 \| r_2 \| r_1)$;
   - $j := (j \oplus n_2) + (f_1 \oplus r_6)$; $F := \{(F + Q[j]) \oplus (r_8 \| r_7 \| r_6 \| r_5)\}^{>5>}$;
   - $j := (j + r_7) \oplus (f_1 + n_3)$; $F := (F + Q[j])^{>6>}$;
   - $j := (j + n_4) \oplus (f_1 + r_8)$; $F := (F \oplus Q[j])^{>7>}$;
   - $j := j + r_2$; $F := F + Q[j]$;
   - $j := j + r_3$; $F := F \oplus Q[j]$;
   - $(f_2 \| f_1) := (f_4 \| f_3) \oplus (f_2 \| f_1)$.
2. Сформировать текущий восьмибитовый элемент ключевой гаммы: $g := f_1 + f_2$.
3. Преобразовать текущий байт t_N:

   $t_N := t_N \oplus g$.
4. **СТОП**.

Данный алгоритм используется как для зашифрования, так и для расшифрования файлов. Он обеспечивает скорость преобразования около 80 Мбит/с для микропроцессора Pentium-II 266. Очевидно, что данный алгоритм предоставляет возможность независимого шифрования каждого байта шифруемого файла. Это создает высокую гибкость при работе с файлами большого размера, например, с базами данных. Вырабатываемая ключевая гамма $g(N)$ фактически является уникальной для каждого файла, поскольку она генерируется в зависимости от локального ключа, формируемого по случайной метке.

Отметим некоторые особенности указанного алгоритма файлового шифрования. Если какой-либо файл известен криптоаналитику в исходном и зашифрованном виде, то он без труда вычислит соответствующую этому файлу ключевую гамму, однако по ключевой гамме вычислительно сложно определение ключа файлового шифрования (даже в случае подобранного исходного текста). В силу этого раскрытие одних файлов не приводит к нарушению секретности других, а атака на основе шифрования специально подобранных файлов не дает криптоаналитику дополнительных возможностей для вычисления секретного ключа. При хищении носителей информации нарушитель легко может узнать значение метки, однако он не может вычислить локальный ключ, поскольку для этого требуется знание ключа файлового шифрования. Алгоритм файлового шифрования является стой-

ким к известным методам криптоанализа, включая атаку на основе генерации случайных аппаратных ошибок [46].

2.7.6. Преобразование данных в загрузочном секторе

Для расшифрования (зашифрования) загрузчика операционной системы в криптосистеме SPECTR-Z разработан мини-алгоритм, реализуемый в виде программного криптомодуля объемом около 100 байт, что позволило разместить его внутри загрузчика системы защиты (размер которого составляет всего 512 байт).

Общая криптосхема мини-алгоритма включает два этапа:

1. Предвычисления, выполняемые для построения 1024-байтового расширенного ключа $K[j]$.
2. Процедуры расшифрования **D**, использующие расширенный ключ. Расширенный ключ представляется в виде последовательности 32-битовых подключей $\{K[j]\}$, где $j = 0, 1, 2, ..., 255$.

Для выполнения предвычислений используется построение 1024-байтового вспомогательного ключа L путем повторения пароля необходимое число раз при одновременном модифицировании байтов пароля. Затем формируется $K:=L$. После этого 35 раз выполняются следующие преобразования $L := \mathbf{D}_K(L)$, $K := L$, где под $\mathbf{D}_K(L)$ подразумевается результат применения процедуры **D** с ключом K к блоку L. В итоге получаем расширенный ключ $K = \{K[j]\}$, $j = 0, 1, ..., 255$, который в дальнейшем используется в качестве ключа в процедуре **D** для расшифрования загрузочного сектора. Процедура **D** реализуется путем разбиения шифруемого сообщения на 64-битовые блоки и их последовательного преобразования. Блоки данных представляются в виде $T = A \| B$, где $A=a_4 \| a_3 \| a_2 \| a_1$, $B = b_4 \| b_3 \| b_2 \| b_1$, a_i и b_i — восьмибитовые подблоки, знак "||" обозначает операцию конкатенации. В качестве процедуры расшифрования 64-битовых блоков данных используется следующий алгоритм.

Процедура *D* (мини-алгоритм расшифрования)

Мини-алгоритм расшифрования состоит из следующих шагов:

1. Установить счетчик $r = 1$.
2. Выполнить преобразования:
   - $A := A + K[b_1]$; $B := B \oplus K[a_1]$;
   - $A := A \oplus B$; $A := A^{>3>}$;
   - $B := B^{>3>}$; $B := B + A$.
3. Если $r < 35$, то прирастить $r := r + 1$ и перейти к шагу 2.
4. **СТОП**.

Значение $A \parallel B$ после 35 раундов шифрования является выходом мини-алгоритма. Процедура зашифрования загрузчика операционной системы выполняется при установке системы защиты или смене секретного ключа и легко строится по алгоритму расшифрования.

Мини-алгоритм обеспечивает высокую стойкость к атакам на основе известного и подобранного текста объемом несколько килобайт. В данном механизме найден удачный компромисс между размером модуля и стойкостью. Следует отметить, что у потенциального криптоаналитика нет достаточного объема соответствующих пар блоков входного текста и шифртекста для реализации статистических методов криптоанализа. Кроме того, атакующему, ввиду особенностей применения мини-алгоритма системы защиты, крайне сложно ввести для расшифрования (или зашифрования) какой-либо выбранный им текст. Использование шифрования загрузчика операционной системы позволило обеспечить высокую защищенность от программных закладок. Поскольку для алгоритма мини-шифра важным является размер кода, а не скорость шифрования, то стойкость мини-шифра может быть увеличена, если в качестве одной из базовых операций использовать операцию умножения и/или увеличить число раундов преобразования.

Таким образом, в системе SPECTR-Z криптоалгоритм дискового шифрования гармонично дополнен системой скоростного файлового шифрования и мини-шифром, позволяющим контролировать первоначальную загрузку ЭВМ, что образует стойкую систему многоуровневого шифрования данного программного полнофункционального СЗИ НСД.

2.8. Программная хэш-функция с гибким входом

Системы контроля целостности информации, хранимой на физических носителях и передаваемой по телекоммуникационным каналам, и системы электронной цифровой подписи (ЭЦП) используют в качестве составной части алгоритмы защитного контрольного суммирования (АЗКС). Эти алгоритмы представляют собой некоторый криптографический механизм, позволяющий вычислить для сообщения M произвольной длины контрольный код фиксированного размера l сравнительно малого размера ($l = 64$ — 256 бит). Данный код является "слепком" сообщения M. В зависимости от решаемой задачи в качестве АЗКС могут быть использованы различные алгоритмы, обеспечивающие влияние каждого входного бита данных на значение контрольного кода. Например, могут быть использованы алгоритмы, зависящие от секретного ключа, достоинством которых является сравнительная простота построения при обеспечении высокой скорости вычисления контрольных сумм и стойкости к преднамеренным искажениям данных. Наиболее общим и важным случаем АЗКС являются хэш-функции $\mathbf{H}(M)$, которые не используют секретных параметров и в то же время обеспечивают малую вероятность пропуска факта преднамеренного модифицирования информа-

ции, основанного на анализе свойств хэширующего преобразования. Эта ситуация имеет место в системах ЭЦП, где вместо подписывания сообщений (и электронных документов) используется подписывание соответствующих им значений хэш-функций. Важность построения стойких хэш-функций связана с тем, что ряд атак на ЭЦП основан на нападениях на используемые хэш-функции. Широкое применение программных реализаций ЭЦП делает актуальным разработку скоростных программных хэш-функций. К хэш-функциям предъявляются следующие требования:

- вычислительно неосуществимо нахождение сообщения M, хэш-функция от которого была бы равна случайно выбранному значению h, т. е. такого M, для которого выполняется соотношение $h = \mathbf{H}(M)$;
- вычислительно неосуществимо нахождение двух сообщений M_1 и $M_2 \neq M_1$ с равными значениями хэш-функции, т. е. удовлетворяющих условию $\mathbf{H}(M_1) = \mathbf{H}(M_2)$.

Один из способов построения хэш-функций состоит в использовании некоторого итеративного механизма на базе стойкого блочного шифра $\mathbf{E}$, а именно $h_i = \mathbf{E}(h_{i-1}, M_i)$, $i = 1, 2, ..., n$, где h_0 — некоторое специфицированное начальное значение, M_i — блоки сообщения $M = (M_1, M_2, ..., M_n)$ одинаковой длины l. Если длина сообщения не является кратной l, то последний блок дополняется по оговоренному правилу до длины l. Значение $h = h_n$ принимается за значение хэш-функции, которую можно записать в виде $h = \mathbf{H}(h_0, M)$, что подчеркивает зависимость h от начального параметра h_0. В таких схемах построения стойкость хэш-функций не превышает стойкости базовой итеративной функции $\mathbf{E}$, что делает необходимым применение криптографически стойких базовых функций.

Общей атакой на хэш-функции является атака, использующая "парадокс дней рождения" и имеющая трудоемкость около $2^{l/2}$ вычислений функции $\mathbf{E}(h_{i-1},M_i)$, независимо от выбранной раундовой функции (при этом используется память объемом порядка $2^{l/2}l/8$ байт, где l — длина хэш-кода). Это обусловливает необходимость применения хэш-функций со значением $l \geq 128$ бит. Современные блочные шифры являются стойкими к атаке на основе известного исходного текста, поэтому если использовать блок M_i в качестве ключа, то по входу h_{i-1} функции $\mathbf{E}$ и выходу h_i вычислительно неосуществимо нахождение значения M_i.

В качестве базовых функций можно использовать 128-битовые блочные шифры RC6, MARS, Twofish и др. С целью использования 64-битовых блочных шифров, например DES, ГОСТ 28147-89 были предложены специальные схемы построения l-битовых хэш-функций на основе $(l/2)$-битовых блочных шифров [83].

Практическая задача проверки целостности больших массивов в компьютерных системах данных делает актуальной разработку скоростных программно-ориентированных хэш-функций. Для ее решения могут быть использованы блочные программные шифры, основанные на выборке подключей в зависимости от преобразуемых данных. Данный механизм преобразования фактически реализует

операции подстановки (размера 11×32 и 8×32), зависящие от преобразуемых данных. В свою очередь, последние могут быть интерпретированы как фиксированные операции подстановок увеличенного размера. Именно увеличение размера фактически реализуемых подстановок лежит в основе стойкости указанных шифров.

Отметим, что под подстановками размера $q \times p$ здесь понимается некоторое функциональное преобразование $\mathbf{F}(x)$, реализуемое с помощью таблиц замены $\{Q[j]\}$, где $j = 0, 1, ..., 2^q - 1$, которые содержат 2^q p-битовых двоичных векторов $Q[j]$. Операция подстановки реализуется как выборка элемента таблицы с номером x, т. е. в качестве $\mathbf{F}(x)$ берется значение $Q[x]$. В программных шифрах таблица замены формируется в зависимости от секретного ключа и служит расширенным ключом шифрования, а выборка ее элементов имеет псевдослучайный характер.

Непосредственное использование шифров с псевдослучайной выборкой в качестве рассмотренной выше функции **E** и блоков M_i в качестве ключа неприемлемо, поскольку в шифрах подобного типа используется сравнительно сложный этап предвычислений, на котором по секретному ключу строится расширенный ключ шифрования, что существенно ограничивает скорость хэширования (вычисления значения хэш-функции). Рассматриваемая скоростная программная хэш-функция основана на раундовой функции, базирующейся на операциях подстановок, зависящих от преобразуемых данных и осуществляющих одностороннее преобразование $y = f(x)$, т. е. такое преобразование, что для случайного выбранного значения y вычислительно сложно найти такое значение x, при котором будет выполняться условие $f(x) = y$. Операции подстановок, зависящие от преобразуемых данных, реализуются через выборку, зависящую от хэшируемых данных, 32-битовых элементов из известной таблицы. Используемые процедуры преобразования подобны процедурам, используемым в псевдовероятностных шифрах, за исключением того, что односторонний характер преобразования обеспечивается отменой начальной инициализации переменных перед выполнением каждого раунда шифрования.

2.8.1. Схемы хэш-функций

Функцию $f(x)$ можно использовать в достаточно простых схемах построения итеративных хэш-функций, показанных на рис. 2.4 и соответствующих следующим формулам преобразования:

- $h_i = h_{i-1} \oplus f(M_i)$ (рис. 2.4, *а*);
- $h_i = h_{i-1} \oplus f(M_i) \oplus M_i$ (рис. 2.4, *б*);
- $h_i = h_{i-1} \oplus f(M_i \oplus h_{i-1})$ (рис. 2.4, *в*);

где M_i, $i = 1, 2, ..., n$ — блоки данных и $h_0 = (00\ldots0)$.

Для схем на рис. 2.4, *а* и 2.4, *в* принципиальным является использование односторонней функции $f(x)$, поскольку применение вместо нее шифрующего блоч-

ного преобразования с некоторым специфицированным фиксированным ключом является недопустимым ввиду появления возможности легкого модифицирования сообщений с сохранением значения хэш-функции. В случае схемы, представленной на рис. 2.4, *б*, такая замена не является критической, однако и в этом случае она может привести к потенциальным слабостям. Схемы построения хэш-функций на основе итеративного использования блочного одностороннего преобразования не являются столь многообразными, как в случае применения блочного шифрующего преобразования **E**, имеющего вход для данных и вход для ключа. Другие схемы построения хэш-функций на основе функции $f(x)$ связаны с использованием двух функций подобного типа в одном раунде хэширования, что снижает скорость, однако принципиального дополнительного усиления не дает. Далее предполагается, что рассматриваемые блочные односторонние функции применяются в соответствии с указанными выше формулами.

Рис. 2.4. Схемы итеративных хэш-функций на основе односторонних функций

2.8.2. Построение функции *f* (*x*)

В рассматриваемой хэш-функции на основе псевдослучайных выборок используется таблица ***Q***, задаваемая в виде последовательности из 2051 байта. Данная таблица может храниться в памяти постоянно или вырабатываться при инициализации криптоалгоритма с помощью следующего алгоритма.

Процедура *Table_Q*

Алгоритм задания таблицы ***Q*** состоит из следующих шагов:

1. Установить счетчик $i = 0$.
2. Вычислить 32-байтовое число $Z_i = (a^{23+i} \bmod Q)^{17} \bmod R$.
3. Прирастить $i := i + 1$. Если $i \neq 64$, то перейти к шагу 2.
4. Сформировать 2051-байтовое число $S = q_2 \| q_1 \| q_0 \| Q_{63} \| \ldots \| Q_1 \| Q_0$, где $q_2 \| q_1 \| q_0$ — младшие 24 разряда числа Q_0.

5. Число S интерпретируется как последовательность $\boldsymbol{Q}$, включающая 2051 байт $\{q[i]\}$, где $i = 0, 1, ..., 2050$.
6. **ВЫХОД:** $\bar{\mathbf{Q}} = \{q[0], q[1], \ldots, q[2050]\}$.

Таблица $\boldsymbol{Q}$ представляет собой упорядоченную последовательность байтов $q[i]$: $\boldsymbol{Q} = \{q[i]\}$, где $i = 0, 1, ..., 2050$. При хэшировании используются 32-битовые элементы таблицы $Q[j] = q[j] \parallel q[j+1] \parallel q[j+2] \parallel q[j+3]$, где $j = 0, 1, ..., 2047$. Последовательность $\{q[i]\}$ предполагается общедоступной. Таблица $\boldsymbol{Q}$ может генерироваться с помощью процедуры ***Table_Q*** или задаваться непосредственно табличным способом, т. е. храниться в памяти ЭВМ постоянно. В последнем случае нет необходимости выполнять вычисления, задаваемые процедурой ***Table_Q***.

Блоки хэшируемых данных M_i длины $32m$ бит, где $m \geq 4$, будем рассматривать как конкатенацию 32-битовых слов $T_0, T_1, ..., T_{m-1}$, значения которых используем для управления выборкой элементов $Q[j]$. Для обеспечения основных требований к хэш-функциям в раундовом преобразовании применим механизм, обеспечивающий влияние текущего преобразуемого слова на все последующие шаги выборки и на преобразование всех последующих слов текущего блока данных. Это реализуется за счет использования рекурсивных формул для вычисления текущих значений переменных процесса раундового хэширования. Используются четыре 32-битовые переменные, через которые реализуется эффективный механизм "продленного" влияния преобразуемых слов на выборку элементов из таблицы $\boldsymbol{Q}$. Именно данный механизм определяет "псевдослучайный" характер выборки и ее принципиальное отличие как простой операции подстановки.

Раундовая хэш-функция

Алгоритм формирования раундовой хэш-функции состоит из следующих шагов:

1. Задать значение $m \geq 4$, установить значение счетчика циклов $j := 6$ и вычислить начальные значения внутренних переменных $R := Q[0]$, $V := Q[1]$, $Y := Q[2]$, $U := Q[3]$, $N := Q[4]$.
2. Инициализировать внутренний счетчик $i := 1$.
3. Выполнить процедуру ***Change_NVYU***.

   {*Change_NVYU*:

   - $N := N \oplus R$; $V := V +_{32} N$;
   - $N := N \otimes F$; $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>>>11}$;
   - $N := N \oplus V$; $Y := Y +_{32} N$;
   - $N := N \otimes F$; $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>>>11}$;
   - $N := N +_{32} Y$; $N := N \otimes F$; $n := N +_{11} 0$;
   - $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$.
   - **END**}

4. Осуществить преобразование очередного слова текста: $T_i := (T_i -_{32} V) \oplus U$.
5. Преобразовать переменную R: $R := R +_{32} T_i$.
6. Завершить преобразование слова T_i: $T_i := T_i^{<<<V} -_{32} Y$.
7. Прирастить $i := i + 1$. Если $i \neq m$, то перейти к шагу 3.
8. Прирастить $j := j + 1$. Если $j \neq 0$, то перейти к шагу 2.
9. **СТОП**.
10. **ВЫХОД**: значение хэш-функции равное $T_{m-1} \| T_{m-2} \| \ldots \| T_1 \| T_0$.

Данная процедура включает преобразования, аналогичные используемым в алгоритме шифрования SPECTR-Z [2], однако для получения высокой сложности обращения раундовой функции инициализация переменных R, V, Y, U и N осуществляется только один раз, а именно в начале первого цикла преобразований. В последующих пяти циклах раундовой функции в качестве начальных значений этих переменных берутся их финальные значения в предыдущем цикле. При составлении алгоритма хэширования данных возможны следующие два варианта:

- инициализация переменных R, V, Y, U и N производится перед осуществлением каждого текущего раундового преобразования;
- инициализация переменных R, V, Y, U и N осуществляется только перед вычислением первого раундового значения хэш-функции, а для каждой последующей итерации в качестве начальных значений R, V, Y, U и N берутся их финальные значения на предыдущем раунде вычисления хэш-функции.

Оба варианта являются приемлемыми с точки зрения обеспечения криптографической стойкости, хотя второй вариант при малом значении m (например, $m = 64$) к некоторым типам атак представляется более стойким. Скорость хэширования данных для микропроцессора Pentium-II 266 МГц при использовании этого алгоритма составляет около 50 Мбит/с. Стойкость функции $f(x)$ связана со следующими факторами:

- используются табличные преобразования над векторами достаточно большой длины (11 бит);
- число возможных комбинаций значений переменных U, V, Y, N и R процесса преобразования достаточно велико и составляет 2^{155};
- табличные подстановки выполняются в зависимости от преобразуемых данных, причем текущее преобразуемое слово влияет на подстановки, выполняемые на всех последующих шагах преобразования, включая шаги, относящиеся ко всем последующим циклам.

Процедуры преобразования составлены так, что для двух блоков данных, различающихся в слове T_i, при преобразовании слова T_{i+1} в переменных U, V, Y, N и R появляются различия, которые затем лавинообразно нарастают при переходе к следующим элементарным шагам преобразования, описываемым пунктами 3 и 4

приведенного выше алгоритма. Из этого свойства следует, что изменение одного бита в блоке данных приводит во втором и последующих циклах вычисления раундовой функции к "непредсказуемому" изменению выборки элементов $Q[n]$, которые формируют значения переменных U, V, и Y. Используемые при преобразовании слов T_i и переменных Y и U операции циклического сдвига на элементарных шагах второго и последующих циклов зависят по сложному закону от каждого бита входного блока данных, что также вносит вклад в повышении стойкости хэш-функции.

Рассмотренный алгоритм хэширования выгодно отличается параметризованным входом, что позволяет преобразовывать блоки данных различной длины, например 128, 256 и 512 бит, в зависимости от конкретных приложений. Данная хэш-функция свободно может быть превращена в АЗКС, использующий секретный параметр. Это легко осуществляется путем задания процедур преобразования таблицы $\boldsymbol{Q}$ в зависимости от секретного ключа. Высокая скорость при программной реализации, стойкость и гибкость входа используемой итеративной хэш-функции делают ее привлекательной для практического использования.

2.9. Программный шифр с гибким входом

Представляет интерес построение программного шифра с гибким входом. Это может быть сделано на основе алгоритма системы SPECTR-Z, причем переменная длина блока должна относиться к той части блока, которая преобразуется в двух раундах. Это позволит получить высокие скорости для блоков большого размера. При малом размере блока скорость будет в несколько раз меньше, поскольку для начальных и конечных слов входного блока по соображениям стойкости число раундов не может быть задано меньше шести.

Рассматриваемый ниже программный блочный алгоритм SPECTR-F представляет собой модифицированную версию алгоритма SPECTR-Z, которая отличается тем, что размер входного блока не является фиксированным, а может быть задан от 128 бит и больше, но кратно 32 битам. Благодаря этому, SPECTR-F обладает необходимой гибкостью для оптимизации выбора длины блока в тех или иных приложениях. Это позволяет получить существенный выигрыш по скорости при шифровании больших блоков данных (до 300 Мбит/с для современных массовых процессоров), а при скорости шифрования около 100 Мбит/с — обеспечить возможность осуществления (в случае необходимости) шифрования блоками по 128 бит.

При разработке криптосистемы SPECTR-F были использованы те же конструктивные критерии, что и для SPECTR-Z. Для обеспечения большей гибкости в алгоритме SPECTR-F применена переменная длина входного блока. В качестве размера входного блока данных для алгоритма SPECTR-F было выбрано параметризованное значение $32m$ (бит), где m является натуральным числом, причем $m \geq 4$.

В криптосистеме SPECTR-F, также как и в других шифрах с расширенным ключом, предполагается, что последний вырабатывается в зависимости от секретного ключа сравнительно небольшой длины с помощью специальной процедуры, выполняемой как предвычисления. Шифр SPECTR-F реализуется в виде двух подпрограмм: модуля инициализации, который предназначен для однократного запуска, и резидентного модуля шифрования, обслуживающего запросы других программ на зашифрование и расшифрование данных.

2.9.1. Предвычисления

Для выполнения предвычислений секретный ключ (входной параметр) повторяется необходимое число раз до получения последовательности из 2051 байта, которую обозначим $\boldsymbol{Q}' = \{q_0', q_1', ..., q'_{2050}\}$. Далее формируется вспомогательный ключ $\boldsymbol{H} = \boldsymbol{Q}' \oplus \boldsymbol{Z}$, где $\boldsymbol{Z}$ — последовательность, формируемая с помощью процедуры ***Table_Z***. Затем несколько раз осуществляются преобразования ключа $\boldsymbol{Q}'$ с помощью процедуры ***Encrypt512*** и ключа $\boldsymbol{H}$. Расширенный ключ $\boldsymbol{Q}$ формируется в соответствии с приведенной ниже процедурой ***FormKey***, в которой $\boldsymbol{Q}'$ интерпретируется как последовательность четырех 512 байтовых блоков $\boldsymbol{Q}^{(1)}$, $\boldsymbol{Q}^{(2)}$, $\boldsymbol{Q}^{(3)}$, $\boldsymbol{Q}^{(4)}$ и трех байтов, т. е. $\boldsymbol{Q}' = \{\boldsymbol{Q}^{(1)}, \boldsymbol{Q}^{(2)}, \boldsymbol{Q}^{(3)}, \boldsymbol{Q}^{(4)}, q^{(0)}, q^{(1)}, q^{(2)}\}$, где $q^{(0)}=q'_{2048}$, $q^{(1)}=q'_{2049}$ и $q^{(2)}=q'_{2050}$. Ключ шифрования $\boldsymbol{Q}$ представляет собой последовательность байтов q_i: $\boldsymbol{Q} = \{q_i\}$, где i=0,1, …, 2050. При шифровании данных используются подключи $Q_j = q_{j+3} \parallel q_{j+2} \parallel q_{j+1} \parallel q_j$, где j = 0, 1, ..., 2047.

Процедура *Table_Z*

Алгоритм выполнения процедуры ***Table_Z*** состоит из следующих шагов:

1. Установить счетчик $i = 0$.
2. Вычислить 32-байтовое число $Z_i' = (a^{23+i} \bmod P)^{17} \bmod R$.
3. Прирастить $i := i + 1$. Если $i \neq 64$, то перейти к шагу 2.
4. Сформировать 2051-байтовое число $S = z_2' \parallel z_1' \parallel z_0' \parallel Z_{63}' \parallel \ldots \parallel Z_0'$, где $z_2' \parallel z_1' \parallel z_0' = Z_0' +_{24} 0$.
5. Представить S в виде последовательности байтов $\boldsymbol{Z} = \{z_0, z_1, ..., z_{2050}\}$.

Процедура *FormKey*

Алгоритм выполнения процедуры ***FormKey*** состоит из следующих шагов:

1. Установить параметр m = 128 и взять $\boldsymbol{H}$ в качестве ключа шифрования.
2. Преобразовать $\boldsymbol{Q}^{(1)}$: $\boldsymbol{Q}^{(1)} := \boldsymbol{Encrypt512_H}\,(\boldsymbol{Q}^{(1)})$.
3. Преобразовать $\boldsymbol{Q}^{(2)}$: $\boldsymbol{Q}^{(2)} := \boldsymbol{Encrypt512_H}\,(\boldsymbol{Q}^{(2)} \oplus \boldsymbol{Q}^{(1)})$.
4. Преобразовать $\boldsymbol{Q}^{(3)}$: $\boldsymbol{Q}^{(3)} := \boldsymbol{Encrypt512_H}\,(\boldsymbol{Q}^{(3)} \oplus \boldsymbol{Q}^{(2)})$.

5. Преобразовать $\boldsymbol{Q}^{(4)}$: $\boldsymbol{Q}^{(4)} := \boldsymbol{Encrypt512_H}\,(\boldsymbol{Q}^{(4)} \oplus \boldsymbol{Q}^{(3)})$.
6. Сформировать расширенный ключ: $\boldsymbol{Q} = \{\boldsymbol{Q}^{(1)}, \boldsymbol{Q}^{(2)}, \boldsymbol{Q}^{(3)}, \boldsymbol{Q}^{(4)}, q^{(0)}, q^{(1)}, q^{(2)}\}$.

2.9.2. Алгоритмы преобразования

Алгоритм шифрования SPECTR-F включает два полных и четыре сокращенных раунда шифрования. Блок открытого текста $\boldsymbol{T}$ разбивается на четыре 32-битовых слова T_i: $T = \{T_i\}$, где $i = 0, 1, ..., m - 1$ ($m \geq 4$). Значение натурального числа m устанавливается в зависимости от области применения. В каждом раунде осуществляется последовательное преобразование входных 32-битовых слов $T_0, T_1, T_2, ..., T_{m-1}$. Отметим, что при $m = 4$ полный и сокращенный раунды идентичны. После каждого раунда, кроме последнего, осуществляется обмен значениями в следующих парах слов: $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$. Алгоритмы преобразования включают следующие две типовые процедуры: ***Initialize*** и ***Change_NVYU***.

Процедура *Initialize*

Алгоритм выполнения процедуры ***Initialize*** состоит из следующих шагов:

1. Установить значение внутреннего счетчика $i := 0$ и начальные значения переменных $R := Q[9]$, $V := Q[7]$, $Y := Q[3]$, $U := Q[9]$, $N := Q[5]$.
2. **END**.

Процедура *Change_NVYU*

Алгоритм выполнения процедуры ***Change_NVYU*** состоит из следующих шагов:

1. $N := N \oplus R$; $V := V +_{32} N$;
2. $N := N \otimes F$; $n := N +_{11} 0$; $V := (V +_{32} Q[n])^{>>>11}$;
3. $N := N \oplus V$; $Y := Y +_{32} N$;
4. $N := N \otimes F$; $n := N +_{11} 0$; $Y := (Y +_{32} Q[n])^{>>>11}$;
5. $N := N +_{32} Y$; $N := N \otimes F$; $n := N +_{11} 0$;
6. $U := ((U \oplus Q[n]) +_{32} R)^{>>>V}$.
7. **END**.

Шифр SPECTR-F описывается следующими алгоритмами.

Процедура зашифрования в четырех сокращенных раундах

Алгоритм выполнения процедуры зашифрования в четырех сокращенных раундах состоит из следующих шагов:

1. Установить значение внешнего счетчика $j := 0$.
2. Выполнить процедуру ***Initialize***.

3. Выполнить процедуру ***Change_NVYU***.
4. Осуществить преобразование очередного слова текста:
   $T_i := (T_i -_{32} V) \oplus U$.
5. Преобразовать переменную R: $R := R +_{32} T_i$.
6. Завершить преобразование слова T_i: $T_i := T_i^{<<<V} -_{32} Y$.
7. Прирастить $i := i + 1$. Если $i \neq 4$, то перейти к шагу 3.
8. Выполнить операцию обмена значениями в следующих парах слов:
   $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. Прирастить $j := j + 1$. Если $j \neq 4$, то перейти к шагу 2.
10. **СТОП**.

Процедура расшифрования в сокращенных раундах

Алгоритм выполнения процедуры расшифрования в сокращенных раундах состоит из следующих шагов:

1. Установить значение внешнего счетчика $j := 0$.
2. Выполнить процедуру ***Initialize***.
3. Выполнить процедуру ***Change_NVYU***.
4. Осуществить преобразование очередного слова текста:
   $T_i := (T_i +_{32} Y)^{>>>V}$.
5. Преобразовать переменную R: $R := R +_{32} T_i$.
6. Завершить преобразование слова T_i: $T_i := (T_i \oplus U) +_{32} V$.
7. Прирастить $i := i + 1$. Если $i \neq 4$, то перейти к шагу 3.
8. Выполнить операцию обмена значениями в следующих парах слов:
   $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. Прирастить $j := j + 1$. Если $j \neq 4$, то перейти к шагу 2.
10. **СТОП**.

Процедура *Encrypt512*: первый (полный) раунд зашифрования

Алгоритм выполнения процедуры ***Encrypt512*** состоит из следующих шагов:

1. Выполнить процедуру ***Initialize***.
2. Выполнить процедуру ***Change_NVYU***.
3. Осуществить преобразование очередного слова текста: $T_i := (T_i -_{32} V) \oplus U$.

4. Преобразовать переменную R: $R := R +_{32} T_i$.
5. Завершить преобразование слова T_i: $T_i := T_i^{<<<V} -_{32} Y$.
6. Прирастить $i := i + 1$. Если $i \neq m$, то перейти к шагу 2.
7. Если $m > 4$, то преобразовать слова T_2 и T_3 в следующей очередности:

   $T_2 := T_2 \oplus T_{m-2}$ и $T_3 := T_3 \oplus T_{m-1}$.
8. Обменять значениями слова: $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. **СТОП.**

Процедура шестого (полного) раунда (зашифрование)

Алгоритм выполнения процедуры шестого (полного) раунда (зашифрование) состоит из следующих шагов:

1. Выполнить процедуру ***Initialize***.
2. Если $m > 4$, то преобразовать слова T_3 и T_2 в следующей очередности:

   $T_3 := T_3 \oplus T_{m-1}$ и $T_2 := T_2 \oplus T_{m-2}$.
3. Выполнить процедуру ***Change_NVYU***.
4. Осуществить преобразование очередного слова текста: $T_i := (T_i -_{32} V) \oplus U$.
5. Преобразовать переменную R: $R := R +_{32} T_i$.
6. Завершить преобразование слова T_i: $T_i := T_i^{<<<V} -_{32} Y$.
7. Прирастить $i := i + 1$. Если $i \neq m$, то перейти к шагу 2.
8. **СТОП**.

Процедура первого (полного) раунда (расшифрование)

Алгоритм выполнения процедуры первого (полного) раунда (расшифрование) состоит из следующих шагов:

1. Выполнить процедуру ***Initialize***.
2. Выполнить процедуру ***Change_NVYU***.
3. Осуществить преобразование очередного слова текста: $T_i := (T_i +_{32} Y)^{>>>V}$.
4. Преобразовать переменную R: $R := R +_{32} T_i$.
5. Завершить преобразование слова T_i: $T_i := (T_i \oplus U) +_{32} V$.
6. Прирастить $i := i + 1$. Если $i \neq m$, то перейти к шагу 2.
7. Если $m > 4$, то преобразовать слова T_2 и T_3 в следующей очередности:

   $T_2 := T_2 \oplus T_{m-2}$ и $T_3 := T_3 \oplus T_{m-1}$.

8. Обменять значениями слова: $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. **СТОП**.

Процедура шестого (полного) раунда (расшифрование)

Алгоритм выполнения процедуры шестого (полного) раунда (расшифрование) состоит из следующих шагов:

1. Выполнить процедуру ***Initialize***.
2. Если $m > 4$, то преобразовать слова T_3 и T_2 в следующей очередности:
   $T_3 := T_3 \oplus T_{m-1}$ и $T_2 := T_2 \oplus T_{m-2}$.
3. Выполнить процедуру ***Change_NVYU***.
4. Осуществить преобразование очередного слова текста:
   $T_i := (T_i +_{32} Y)^{>>>V}$.
5. Преобразовать переменную R: $R := R +_{32} T_i$.
6. Завершить преобразование слова T_i: $T_i := (T_i \oplus U) +_{32} V$.
7. Прирастить $i := i + 1$. Если $i \neq m$, то перейти к шагу 2.
8. **СТОП**.

Схема преобразований исходного текста показана на рис. 2.5.

Криптоалгоритм SPECTR-F ориентирован на программную реализацию. Скорость шифрования зависит от размера входного блока. При $m = 4$ скорость минимальна и составляет около 50 Мбит/с (для процессора Pentium 266). С увеличением размера блока скорость увеличивается, принимая значение около 140 Мбит/с при $m \geq 32$.

Благодаря использованию операций обмена значениями, каждое из слов T_0, T_1, T_2 и T_3 в четырех сокращенных раундах преобразуется с использованием набора из 27 подключей, выбираемых в зависимости от преобразуемых данных. Выходные значения крайних слов T_0 и T_3 зависят дополнительно от 3 подключей, которые входят в процедуры их преобразования трижды. Однако данная тройка подключей является фиксированной для всех блоков данных и рассматривается как вносящая небольшой эффект в повышение стойкости. Основным фактором повышения стойкости является задание динамической выборки 27 подключей для каждого преобразуемого слова. В процедуру преобразования слов с индексами $i = 4, 5, ..., m - 1$ входят $6i$ "переменных" подключей (причем более половины из этих подключей входят многократно).

Подбором специального подмножества текстов атакующий может задать ход преобразования в первом раунде одинаковым для каждого текста из такого подмножества. Использовать этот факт в целях анализа преобразованных значений слов T_0, T_1, T_2 и T_3 для раскрытия подключей представляется существенно более

сложным по сравнению с анализом слов T_4, T_5, …, T_{m-1}, поскольку первые преобразуются при выполнении всех шести раундов, а последние — только в первом и последнем раундах. Это сделано для получения высокой скорости при шифровании блоков данных большого размера. Атаки, связанные с использованием анализа T_4, T_5, ..., T_{m-1}, могут быть положены в основу варианта комбинаторно-вероятностной модели, учитывающего особенности шифра SPECTR-F.

Рис. 2.5. Схема очередности преобразования 32-битовых слов в шифре SPECTR-F

Криптоалгоритм SPECTR-F является стойким к известным криптоаналитическим атакам, включая линейный и дифференциальный криптоанализ. Несмотря на малое число полных раундов преобразования, данный алгоритм обладает высокой стойкостью благодаря специфике преобразований, характерной для криптосистем с большим размером входного текста и основанной на выборке подключей в зависимости от преобразуемых данных. При разработке шифра SPECTR-F для сопоставительной оценки стойкости различных вариантов криптосхем с псевдослучайной выборкой подключей была использована атака [46], учитывающая возможность внешнего физического воздействия со стороны атакующего на устройство шифрования. При этом рассматривался вариант формирования *случайных* ошибок в регистрах, содержащих шифруемые данные. Криптоалгоритм SPECTR-F оказался стойким к подобной атаке, что обеспечивается теми же ме-

ханизмами, что и в случае шифра SPECTR-Z (это обусловлено сходством механизма преобразований в этих двух шифрах).

Для обобщенной оценки минимального уровня криптостойкости шифра SPECTR-F в случае выбора параметра $m \geq 5$ (т. е. при размере входного блока 20 байтов и более) можно воспользоваться комбинаторно-вероятностной моделью (КВМ), использованной выше для оценки алгоритма SPECTR-Z. Значения переменных U, V и Y на шаге преобразования i-го слова формируются в зависимости от комбинаций из i сменяемых подключей ($i = 0, 1, 2, ..., m - 1$), при этом мощность множества значений M_i, принимаемых этими переменными, зависит от i (для данного i значения U, V и Y зависят от входного блока). Легко показать, что $M_0 = 1$, $M_1 \approx 2^{11}$, $M_2 \approx 2^{21}$, $M_3 \approx 2^{30}$ и $M_i \approx 2^{32}$ для $i \geq 4$. Таким образом, слова с номерами $i \geq 4$ преобразуются с использованием одного из возможных $\approx 2^{96}$ различных наборов $\{Y, U, V\}$, зависящих от входного блока и ключа шифрования. Алгоритм составлен так, что *процедуры преобразования обеспечивают влияние любого бита входного сообщения на выборку подключей.* Этот критерий гарантирует, что для всех различных входных сообщений будут генерироваться уникальные последовательности наборов $\{Y_{(i)}, U_{(i)}, V_{(i)}\}$ где индексом (i) отмечены значения переменных на i-м шаге преобразования.

Только в первом и частично втором раунде (в первом цикле) на некоторых шагах преобразования для специально подобранных входных сообщений значения переменных U, V и Y могут совпадать предопределенно. Криптоаналитик это легко задаст, однако ему не известны конкретные значения переменных. Для этого он может выбрать два входных сообщения, отличающиеся только в слове T_{m-1}. Что касается последних четырех раундов, значения переменных для любой пары входных сообщений могут совпасть только случайно.

Определение значений подключей $Q[j]$ связано с нахождением значений U, V и Y на определенных шагах преобразования (например, на двух ближайших последовательных шагах преобразования слов), поэтому фиксирование значений U, V и Y на определенных шагах является предварительным условием вычисления подключей $Q[j]$. Под фиксированием будем понимать нахождение таких слов в одном или различных входных сообщениях, которые были преобразованы с использованием значений U, V и Y, связанных каким-либо условием. Например, для двух различных слов T и T', соответствующие им пары значений каждой из "аккумулирующих" переменных, использованные при их преобразовании, равны (т. е. $U = U'$, $V = V'$ и $Y = Y'$), составляют заданную разность ($U \oplus U' = \text{const}_1$, $V \oplus V' = \text{const}_2$ и $Y \oplus Y' = \text{const}_3$) или связаны линейным соотношением. Заметим, что понятие фиксирования включает в себя любые заданные зависимости между значениями "аккумулирующих" переменных, соответствующих выбираемой паре слов, т. е. применение дифференциального и линейного криптоанализа охватывается комбинаторно-вероятностной моделью как частный случай.

Уравнение, описывающее шифрование слов T, имеющих на входе номера $i \geq 4$ (эти слова преобразуются только на первом и шестом раундах, тогда как осталь-

ные слова преобразуются во всех шести раундах), в общем виде задается выражением: $C = f(T, U_1, V_1, Y_1, U_6, V_6, Y_6)$, где индекс соответствует номеру раунда шифрования. Анализ экспериментальной статистики последовательности значений, принимаемых переменными U, V и Y, подтверждает, что эти переменные принимают псевдослучайные значения.

Для получения обобщенных минимальных оценок будем рассматривать наиболее слабое звено преобразований, а именно слова с номерами $i \geq 4$, подвергающиеся минимальному преобразованию, и сделаем ряд допущений, которых в реальных условиях атакующий не имеет. Будем предполагать, что решение системы уравнений, соответствующих преобразованию двух отдельных слов, является проблемой с низкой сложностью, если эти слова преобразованы с помощью фиксированных наборов значений "аккумулирующих" переменных U_1, V_1, Y_1, U_6, V_6, Y_6. Для получения численных оценок стойкости воспользуемся комбинаторно-вероятностной моделью. Сложностью решения уравнений, соответствующих условию фиксирования, будем пренебрегать, полагая, что сложность криптоанализа обусловлена только выявлением пары слов, которые соответствуют условию фиксирования. Это соответствует достаточно общему принципу нападения на шифры — распознаванию (при атаке на основе известного исходного текста) или заданию (при атаке на основе подобранных текстов) некоторых ожидаемых связей между неизвестными параметрами.

При атаке на основе известных текстов для опробуемых пар слов T и T' значения U_1, V_1, Y_1, U_6, V_6, Y_6 в общем случае различны. В случае атаки на основе специально подобранных текстов можно подбирать такие пары входных блоков, для которых на каждом заданном шаге первого раунда формируются одинаковые значения переменных U_1, V_1, Y_1 (эти переменные будут изменяться при переходе от одного слова к другому, но в первом раунде для таких пар входных блоков это будет происходить синхронно). Эти пары входных сообщений легко получаются путем модифицирования в $\boldsymbol{T}$ слова T_i, где $0 < i < m - 1$. Обозначим через $\boldsymbol{T}'$ модифицированный текст. В полученной таким образом паре текстов $\boldsymbol{T}$ и $\boldsymbol{T}'$ слова T_j, где $0 \leq j \leq i$, в первом раунде будут преобразовываться с помощью одинаковых значений U_1, V_1, Y_1.

Однако после выполнения четырех последующих раундов любое изменение в $\boldsymbol{T}'$ приведет к изменению всех битов слов T_0, T_1, T_2, T_3 с вероятностью 0,5. Ход шифрования в третьем раунде будет связан с псевдослучайными значениями разностей $U_6 - U_6'$, $V_6 - V_6'$ и $Y_6 - Y_6'$. В паре текстов $\boldsymbol{T}$ и $\boldsymbol{T}'$ с наибольшей пользой для атакующего может быть использован факт повторения в первом раунде значений U_1, V_1, Y_1 для слов T_j и T'_j, где $5 \leq j \leq m - 1$. (Анализ первых четырех слов представляется более сложным, поскольку они дополнительно преобразуются во втором, третьем, четвертом и пятом раундах.) Нетрудно видеть, что оценка стойкости алгоритма SPECTR-F такая же, как и для SPECTR-Z (см. табл. 3.1).

Разность двух значений каждой из переменных U, V, Y, примененных для преобразования двух соседних слов в последовательности T_4, T_5, …, T_{m-1}, является, по-

жалуй, "наименее псевдослучайной", однако использование этого факта для выполнения криптоанализа существенно осложнено тем, что значения этих слов после первого раунда шифрования являются неизвестными. Например, фиксируя текст $\boldsymbol{T}$ и выбирая множество соответствующих ему текстов $\boldsymbol{T}'$, можно попытаться набрать статистику изменения пары соседних слов в последовательности T_4, T_5, ..., T_i, однако значения этих слов после первого раунда шифрования являются неизвестными, поскольку неизвестны значения троек переменных U_1, V_1, Y_1, которые были использованы на первом раунде при преобразовании соответствующих слов и которые изменяются при переходе от одного слова к другому. Причем, благодаря формированию этих переменных с использованием механизма сцепления, значение текущего слова влияет на преобразование всех последующих.

Приращения значений переменных U, V, Y при выполнении шестого раунда (т. е. приращения U_6, V_6, Y_6, которые имеют место при переходе от одного слова к другому) зависят не только от трех текущих выбираемых подключей, но и от всех предыдущих слов и ранее выбранных подключей, причем эта зависимость задана через две 32-битовые переменные N и R, которые входят в выражения для преобразования этих переменных. Значения переменных $N_{(4)}$ и $R_{(4)}$, используемые при преобразовании слова T_4, являются псевдослучайными, поскольку они формируются в процессе выполнения первых пяти раундов.

На всех последующих шагах преобразования слов T_5, …, T_{m-1} значения переменных N и R взаимно однозначно зависят от $N_{(4)}$ и $R_{(4)}$, которые вместе задают 64 независимых бита. С учетом того, что при переходе от одного слова к другому используются три новых подключа (мощность множества реализуемых значений каждого из них равна 2^{11}), можно предположить, что для такой схемы анализа понадобится статистика чрезвычайно большого объема, связанная с влиянием на ход шифрования некоторого псевдослучайного "обобщенного" 97-битового параметра. Таким образом, этот вариант атаки достаточно хорошо соответствует рассматриваемой КВМ-2, которую можно охарактеризовать 96-битовым псевдослучайным параметром, представляющим собой тройку переменных (U_6, V_6, Y_6).

ГЛАВА 3

Управляемые перестановочные операции

В данной главе используется следующая терминология.

Пусть GF(2) — поле из двух элементов {0,1} с бинарными операциями сложения "⊕" и умножения "·". Множество всех векторов длины n следующего вида: $X = (x_1, x_2, \ldots, x_n)$, где $x_i \in \mathrm{GF}(2)\ \forall i \in \{1, 2, \ldots, n\}$, обозначим через **X**. Мощность множества **X** обозначим через #**X** (#**X** = 2^n). В зависимости от заданных на множестве операций, оно обладает определенными свойствами и имеет специальные обозначения. Так, например, множество **X** с операцией "·" умножения вектора на скаляр $\alpha \in \mathrm{GF}(2)$, а именно: $\alpha \cdot X = (\alpha x_1, \alpha x_2, \ldots, \alpha x_n)$, и бинарной операцией "⊕" *поразрядного сложения* векторов: $X \oplus V = (x_1 \oplus v_1, x_2 \oplus v_2, \ldots, x_n \oplus v_n)$, является *линейным векторным пространством* (**X**, ⊕, ·) [22], для которого будем использовать обозначение $\mathrm{GF}(2)^n$.

В большинстве случаев для множества **X** в книге используется обозначение $\mathrm{GF}(2)^n$, хотя в некоторых случаях нет необходимости требовать выполнения всех свойств линейного пространства. В других случаях с введением дополнительных операций множество **X** приобретает новые свойства, для характеристики которых в математике предусмотрены специальные термины. Так, например, используя упрощенную формулировку, если на множестве **X** помимо бинарной операции поразрядного сложения (⊕) задана бинарная операция *поразрядного умножения векторов* (⊗), а именно $X \otimes V = (x_1 v_1, x_2 v_2, \ldots, x_n v_n)$, то такое множество является *кольцом* (**X**, ⊕, ⊗).

Замечание

При обозначении двоичных векторов с использованием прописных букв, например X, как правило, подразумевается, что данный вектор рассматривается в качестве переменной величины, в то время как строчная буква x означает некоторое фиксированное значение вектора X. Однако для указания конкретного разряда рассматриваемого вектора в обоих случаях используется строчная буква с нижним индексом (например, x_n).

Для некоторых фиксированных векторов введем специальные обозначения с использованием прописных букв:

- E_i — вектор, содержащий единицу только в i-м разряде, а остальные — нули ($i \in \{1, 2, \ldots, n\}$);
- E_0 — вектор, не содержащий единиц (также будем использовать обозначение e_0);
- D_i — вектор, содержащий один ноль в i-м разряде, а остальные единицы ($i \in \{1, 2, \ldots, n\}$);
- D_0 — вектор, не содержащий нулей (также будем использовать обозначение d_0);
- $\overline{X}$ — дополнение вектора X, то есть вектор $\overline{X} = X \oplus D_0$.

Вектор E_0 является нейтральным по сложению и умножению на скаляр ($X \oplus E_0 = X$ и $\alpha \cdot E_0 = E_0$), а вектор D_0 — нейтральным для операции поразрядного умножения ($X \otimes D_0 = X$) $\forall X \in \mathrm{GF}(2)^n$ и $\forall \alpha \in \mathrm{GF}(2)$.

Ниже приводятся определения наиболее часто встречающихся терминов, связанных с преобразованиями над линейными векторными пространствами.

Конкатенацией $X \| V$ векторов X и V ($X \in \mathrm{GF}(2)^n$ и $V \in \mathrm{GF}(2)^m$) называется отображение $\mathrm{GF}(2)^n \times \mathrm{GF}(2)^m \to \mathrm{GF}(2)^{n+m}$, а именно:

$$Y = X \| V = (x_1, x_2, \ldots, x_n, v_1, v_2, \ldots, v_m).$$

Очевидно, что для произвольной функции $\mathbf{F}(Y)$ справедливы равенства:

$$\mathbf{F}(Y) = \mathbf{F}(X \| V) = \mathbf{F}(X, V).$$

Булевой функцией называется отображение $\mathbf{f}$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)$.

Сбалансированной называется булева функция $\mathbf{f}(X)$, для которой

$$\#\{X \in \mathrm{GF}(2)^n \mid \mathbf{f}(X) = 1\} = \sum_x \mathbf{f}(x) = 2^{n-1}.$$

Векторной булевой функцией называется отображение $\mathbf{F}$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$, а именно:

$$\mathbf{F}(X) = (\mathbf{f}_1(X), \mathbf{f}_2(X), \ldots, \mathbf{f}_m(X)),$$

где $\mathbf{f}_i$ — образующие булевы функции $i \in \{1, 2, \ldots, m\}$, $X \in \mathrm{GF}(2)^n$.

Весом Хэмминга называется целочисленная функция **WH**, значение которой равно числу ненулевых компонентов вектора X. Очевидно, что

$$0 \le \mathbf{WH}(X) \le n,\ \mathbf{WH}(E_i) = 1,\ \mathbf{WH}(E_0) = 0,\ \mathbf{WH}(D_i) = n - 1,\ \mathbf{WH}(D_0) = n$$

и

$$\mathbf{WH}(\overline{X}) = n - \mathbf{WH}(X).$$

Расстоянием Хэмминга между векторами X и Y ($X, Y \in \mathrm{GF}(2)^n$) называется целочисленная функция **DH**, значение которой равно числу ненулевых компонентов вектора $X \oplus Y$, т. е. $\mathbf{DH}(X, Y) = \mathbf{WH}(X \oplus Y)$. Поэтому для обеих функций часто используется одно и то же обозначение **WH**.

Функцией четности вектора называется булева функция $\mathbf{WH}_{\oplus}$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)$, а именно:

$$\mathbf{WH}_{\oplus}(X) = x_1 \oplus x_2 \oplus \ldots \oplus x_n.$$

Скалярным произведением ($\langle X, Y \rangle$) называется целочисленная функция, значение которой равно числу ненулевых компонентов вектора $X \otimes Y$, т. е. $\langle X, Y \rangle = \mathbf{WH}(X \otimes Y)$.

Скалярным произведением ($X \bullet Y$) называется булева функция следующего вида:

$$X \bullet Y = x_1 y_1 \oplus x_2 y_2 \oplus \ldots \oplus x_n y_n.$$

Непосредственно из определений скалярного произведения $X \bullet Y$ и функции четности $\mathbf{WH}_{\oplus}$ справедливо равенство $X \bullet Y = \mathbf{WH}_{\oplus}(X \otimes Y)$, которое будет активно использоваться в настоящей главе при доказательстве различных утверждений, связанных с управляемыми перестановками.

3.1. Понятие управляемой перестановочной операции

Прежде чем приступить к изучению *управляемых перестановочных операций*, поясним существующие различия между понятиями *подстановка* и *перестановка*, которые, в отличие от классической математики, являются существенными для криптографических приложений.

В классической математике понятие "подстановка" вводится следующим образом.

Определение 3.1

Подстановкой σ конечного множества Ω, состоящего из n различных элементов, называется биекция (взаимно однозначное отображение) множества Ω на себя.

Подстановки, соответствующие данному определению, будем называть σ-подстановками. Один из вариантов наглядного представления подстановки σ для $\Omega = \{1, 2, \ldots, n\}$ имеет следующий вид:

$$\boldsymbol{\sigma} = \begin{pmatrix} 1 & 2 & \ldots & n \\ \sigma_1 & \sigma_2 & \ldots & \sigma_n \end{pmatrix}, \tag{3.1}$$

причем в нижней строке указаны значения $\sigma_i \in \{1, 2, \ldots, n\}$ ($\sigma_i \neq \sigma_j$ при $i \neq j$), на которые необходимо заменить элементы верхней строки. Значение n называется длиной подстановки. Например:

$$\boldsymbol{\sigma} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 1 & 4 \end{pmatrix}. \tag{3.2}$$

Множество всех $\boldsymbol{\sigma}$-подстановок длины n принято обозначать символом S_n. Очевидно, что $|S_n| = n!$.

Существуют различные способы представления подстановок. Другой способ задания $\boldsymbol{\sigma}$-подстановок — циклический. Например, для (3.2) подстановка $\boldsymbol{\sigma}$ имеет следующие шесть равнозначных циклических представлений:

$$\boldsymbol{\sigma} = (1,2,3)(4) = (2,3,1)(4) = (3,1,2)(4) = (4)(1,2,3) = (4)(2,3,1) = (4)(3,1,2).$$

Наконец, рассматривая подстановку как функцию от целочисленного аргумента, используют запись $\boldsymbol{\sigma}(i)$, где $\boldsymbol{\sigma}(i) = \sigma_i \ \forall i \in \{1, 2, \ldots, n\}$.

На множестве подстановок S_n в соответствии с общим правилом композиции отображений: $(\boldsymbol{\sigma}' \circ \boldsymbol{\sigma}'')(i) = \boldsymbol{\sigma}''(\boldsymbol{\sigma}'(i))$ легко вводится групповая операция умножения $\circ_{\boldsymbol{\sigma}} = \circ$, причем множество S_n является группой подстановок. В частности, единичный элемент группы принято обозначать символом $\boldsymbol{e}$ ($\boldsymbol{\sigma} \circ \boldsymbol{e} = \boldsymbol{\sigma} = \boldsymbol{e} \circ \boldsymbol{\sigma} \ \forall \boldsymbol{\sigma} \in S_n$), для которого $\boldsymbol{e}(i) = i \ \forall i \in \{1, 2, \ldots, n\}$, а обратный элемент — символом $\boldsymbol{\sigma}^{-1}$

$$\boldsymbol{\sigma} \circ \boldsymbol{\sigma}^{-1} = \boldsymbol{e} = \boldsymbol{\sigma}^{-1} \circ \boldsymbol{\sigma}.$$

Множество S_n, рассматриваемое с естественной операцией умножения его элементов, называется *симметрической группой степени n*.

В криптографии понятие подстановки связано с *процессом замены* одних элементов другими и существенно отличается от классического определения. Требование биекции не является обязательным. Не является обязательным и требование совпадения алфавитов (области определения отображения и области значений отображения). Более того, существуют подстановочные шифры, в которых каждому элементу α исходной последовательности соответствует свое множество значений $\boldsymbol{\Omega}_\alpha$ из алфавита замены $\boldsymbol{\Omega}'$. Однако в данной книге термин "подстановка" применительно к криптографии мы будем использовать в соответствии со следующим определением.

Определение 3.2

Подстановкой типа $\boldsymbol{s}$ или $\boldsymbol{s}$-подстановкой называется отображение $\boldsymbol{s}$: $\boldsymbol{\Omega} \to \boldsymbol{\Omega}'$, где $\boldsymbol{\Omega}$ и $\boldsymbol{\Omega}'$ — конечные множества и $\forall \omega \in \boldsymbol{\Omega}$ существует единственный элемент $\omega' \in \boldsymbol{\Omega}'$: $\omega' = \boldsymbol{\sigma}(\omega)$.

Для $\boldsymbol{\Omega} = \{\omega_1, \omega_2, \ldots, \omega_n\}$ постановка $\boldsymbol{s}$ имеет вид:

$$s = \begin{pmatrix} \omega_1 & \omega_2 & \ldots & \omega_n \\ \omega'_1 & \omega'_2 & \ldots & \omega'_n \end{pmatrix}, \tag{3.3}$$

где $\omega'_1, \omega'_2, \ldots, \omega'_n \in \boldsymbol{\Omega}'$.

Очевидно, что в последовательности $\omega'_1, \omega'_2, \ldots, \omega'_n$ не обязательно все значения должны быть различны.

В частности, в алгоритме DES подстановки такого типа (***s***-подстановки) реализованы в виде специальных устройств, получивших название **S**-блоков. На вход **S**-блока поступает 6-разрядный двоичный вектор $\omega \in GF(2)^6$, а на выходе формируется 4-разрядный двоичный вектор $\omega' \in GF(2)^4$. Такое построение **S**-блока обусловлено необходимостью аппаратной реализации алгоритма, когда каждому значению разрядов входа или выхода соответствует наличие или отсутствие токовой посылки, а весь **S**-блок реализуется в виде электронной схемы. Поэтому правомерно рассматривать **S**-блок в качестве *операционного* блока, реализующего заданную подстановочную операцию.

То есть **S**-блок реализует подстановку ***s***: $GF(2)^6 \to GF(2)^4$, и каждому из 64 векторов $\omega_1, \omega_2, ..., \omega_{64}$ поставлен в соответствие один из 16 двоичных векторов $\omega'_1, \omega'_2, \ldots, \omega'_{16}$. В общем случае **S**-блок, реализующий отображение $GF(2)^n \to GF(2)^m$ (подстановку типа ***s***), называется **S**-блоком размера $n \times m$.

Случай $\mathbf{\Omega'} = \mathbf{\Omega}$ соответствует подстановке типа **σ**, при этом определение 3.2 эквивалентно определению 3.1. Отметим, что **σ**-подстановки широко используются во многих современных блочных шифрах, рассчитанных на программную реализацию (Rijndael [61], ГОСТ 28147-89 [10] и др.). Однако в иностранной литературе для **σ**-подстановки обычно используется термин *"permutation one to one"*, то есть "перестановка один в один", поскольку нижняя строка в двухстрочной записи подстановки является перестановкой элементов верхней строки. С другой стороны, под *перестановкой* обычно понимается процесс (действие), заключающийся в изменении порядка следования элементов исходной последовательности. Соответственно в криптографии существует понятие *шифр перестановки*.

Так, если $Q = \omega_1, \omega_2, \ldots, \omega_n$ — конечная последовательность элементов ω_i ($i = 1, 2, \ldots, n$) из множества $\mathbf{\Omega''}$, именуемого алфавитом, то действие шифра перестановки применительно к последовательности Q можно представить в виде двух строк

$$\begin{pmatrix} \omega_1 & \omega_2 & \ldots & \omega_n \\ \omega_{i_1} & \omega_{i_2} & \ldots & \omega_{i_n} \end{pmatrix}, \tag{3.4}$$

где *нижняя строка указывает измененный порядок следования элементов последовательности Q*. Так, например, если $i_1 = 3$, то независимо от природы элементов ω_i первым элементом преобразованной последовательности всегда будет элемент ω_3. Необходимо отметить, что, во-первых, множества $\{i_1, i_2, \ldots, i_n\}$ и $\{1, 2, \ldots, n\}$ совпадают, а во-вторых, не обязательно все элементы ω_i должны быть различны. Например, в качестве исходной последовательности Q может рассматриваться двоичный вектор длины n, т. е. $\mathbf{\Omega''} = GF(2)$.

В случае бесконечной исходной последовательности перестановка также может быть задана в виде двух строк, а именно:

$$\begin{pmatrix} \omega_1 & \omega_2 & \dots & \omega_n \dots \\ \omega_{i_1} & \omega_{i_2} & \dots & \omega_{i_n} \dots \end{pmatrix},$$

причем в данном случае множество индексов верхней строки и множество индексов нижней строки совпадают с натуральным рядом $N = \{1, 2, 3, \dots\}$.

Независимо от природы элементов алфавита $\mathbf{\Omega''}$ перестановка, рассматриваемая как некоторое преобразование, однозначно определяется взаимным соответствием индексов верхней и нижней строк. Для обозначения перестановки будем использовать букву греческого алфавита $\boldsymbol{\pi}$. Так, например, для (3.4) перестановка $\boldsymbol{\pi}$ имеет вид:

$$\boldsymbol{\pi} = \begin{pmatrix} i_1 & i_2 & \dots & i_n \\ 1 & 2 & \dots & n \end{pmatrix}, \tag{3.5}$$

где j-й столбец ($1 \le j \le n$) означает, что элемент ω с индексом i_j исходной последовательности должен быть переставлен на j-е место преобразованной последовательности. Очевидно, что при таком представлении перестановки $\boldsymbol{\pi}$ взаимное расположение столбцов может быть произвольным. Поэтому обычно используют запись, в которой верхняя строка индексов упорядочена по возрастанию, а именно:

$$\boldsymbol{\pi} = \begin{pmatrix} 1 & 2 & \dots & n \\ \pi_1 & \pi_2 & \dots & \pi_n \end{pmatrix}, \tag{3.6}$$

при этом нижняя строка указывает номера мест, на которые необходимо переставить элементы первой строки. То есть π_i — номер места, на которое необходимо переставить элемент ω с индексом i.

Так, например, если (3.4) для последовательности $Q = \omega_1, \omega_2, \omega_3, \omega_4$ имеет вид:

$$\begin{pmatrix} \omega_1 & \omega_2 & \omega_3 & \omega_4 \\ \omega_3 & \omega_1 & \omega_2 & \omega_4 \end{pmatrix},$$

то тогда представления для (3.5) и (3.6) следующие:

$$\boldsymbol{\pi} = \begin{pmatrix} 3 & 1 & 2 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix} \text{ и } \boldsymbol{\pi} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 1 & 4 \end{pmatrix}. \tag{3.7}$$

Заметим, что в (3.5) и (3.6) множества верхних и нижних строк совпадают с множеством $\{1, 2, \dots, n\}$. То есть перестановка $\boldsymbol{\pi}$ является биекцией (взаимно однозначным отображением) множества $\{1, 2, \dots, n\}$ на себя. Но тогда множество всех перестановок $\boldsymbol{\pi}$ длины n совпадает с S_n. При рассмотрении перестановки как функции от целочисленного аргумента, а именно $\boldsymbol{\pi}(i)$, где $\boldsymbol{\pi}(i) = \boldsymbol{\pi}_i$ $\forall i \in \{1, 2, \dots, n\}$, на множестве перестановок S_n в соответствии с общим правилом композиции отображений: $(\boldsymbol{\pi}' \circ \boldsymbol{\pi}'')(i) = \boldsymbol{\pi}''(\boldsymbol{\pi}'(i))$, операция умножения пере-

становок $\circ_\pi = \circ$ совпадает с групповой операцией $\circ_\sigma$, определенной на множестве подстановок S_n. Точнее группа перестановок $(S_n, \circ_\pi)$ изоморфна группе подстановок $(S_n, \circ_\sigma)$, а понятия симметрическая группа перестановок S_n и симметрическая группа подстановок S_n являются синонимами. Именно этим обстоятельством объясняется "несущественность" различий между терминами перестановка $\boldsymbol{\pi}$ и подстановка $\boldsymbol{\sigma}$ в классической математике.

Вернемся к обсуждению принципов построения **S**-блоков в DES, важных для введения понятий управляемой перестановочной операции и управляемой подстановочной операции.

Особенностью каждого **S**-блока является тот факт, что для каждого из четырех значений (x_1, x_6) входного бинарного вектора $X = (x_1, x_2, x_3, x_4, x_5, x_6) \in \mathrm{GF}(2)^6$ в **S**-блоке реализована некоторая подстановка $\boldsymbol{\sigma}^{(j)}$: $\{(x_2, x_3, x_4, x_5)\} \to \mathrm{GF}(2)^4$, $\boldsymbol{\sigma}^{(j)} \in S_{16}$. Данная особенность позволяет рассматривать двоичный вектор (x_1, x_6) в качестве *управляющего вектора (управляющего кода)*, а **S**-блок — в качестве *управляемой подстановочной операции*. Следует отметить, что для случайно выбранной подстановки $\boldsymbol{s}$ (**S**-блока размера $n \times m$, $n > m$) не всегда существует возможность выбрать $n - m$ разрядов в качестве управляющего вектора. Однако, как показывает практика, наилучшими криптографическими свойствами обладают подстановки, в которых такое разделение возможно. В последнем случае, можно говорить, что **S**-блок является объединением (блоком) 2^{n-m} постановок, каждая из которых принадлежит S_{2^m}.

При построении **S**-блока соответствие между значением управляющего вектора V и выбранной $\boldsymbol{\sigma}$-подстановкой определяется разработчиком с учетом дополнительных требований. Так, если каждой подстановке $\boldsymbol{\sigma} \in S_{2^n}$ присвоить свой уникальный номер, например $\boldsymbol{\sigma}^{(j)}$ ($j = 1, 2, \ldots, 2^n!$), тогда соответствие между управляющим вектором $V \in \mathrm{GF}(2)^m$ и выбранной подстановкой $\boldsymbol{\sigma}^{(j)}$ может быть задано в виде некоторой функции $j = \alpha(V)$, причем разным векторам V могут соответствовать одинаковые подстановки $\boldsymbol{\sigma}^{(j)}$. Так как число различных значений вектора V длины m равно 2^m, следовательно, в **S**-блоке можно использовать не более 2^m различных $\boldsymbol{\sigma}$-подстановок. Таким образом, **S**-блок правомерно рассматривать в качестве операционного *блока управляемых подстановок*, для которого, используя более удобную систему обозначений, введем следующее определение.

Определение 3.3

Отображение $\mathbf{S}_{n/m}(X, V)$ $(\mathrm{GF}(2)^n \times \mathrm{GF}(2)^m \to \mathrm{GF}(2)^n)$, являющееся объединением 2^m подстановок $\boldsymbol{\sigma}^{(j)} \in S_{2^n}$, называется *блоком управляемых подстановок*, если для каждого фиксированного значения $V \in \mathrm{GF}(2)^m$ задана некоторая подстановка $\boldsymbol{\sigma}^{(\alpha(V))} \in S_{2^n}$, такая, что $\mathbf{S}_{n/m}(X, V) = \boldsymbol{\sigma}^{(\alpha(V))}(X)$.

Наряду с обозначением $\mathbf{S}_{n/m}(X, V)$ будем использовать обозначение $\mathbf{S}(X, V)$ и $\mathbf{S}^{(V)}(X)$.

Необходимо отметить, что введенное выше обозначение $\mathbf{S}_{n/m}$ является не только более удобным, но и логичным при изучении управляемых операций. При этом соответствие между параметрами **S**-блока размера $n' \times m'$, введенного в DES, и параметрами блока управляемых подстановок $\mathbf{S}_{n/m}$ следующее: $n' = n + m$, $m' = n$.

В общем случае при построении блока управляемых подстановок длина подстановки, а также число используемых подстановок не обязательно должны быть степенью числа 2. Действительно, пусть даны k фиксированных подстановок $\boldsymbol{\sigma}^{(0)}, \boldsymbol{\sigma}^{(1)}, \ldots, \boldsymbol{\sigma}^{(k-1)}$ длины n, принадлежащих S_n. Тогда параметрическое преобразование $\mathbf{S}(i, j)$, заключающееся в применении подстановки $\boldsymbol{\sigma}^{(j)}$ к аргументу i, где $i \in \{1, 2, \ldots, n\}$, можно рассматривать в качестве блока управляемых подстановок.

Определение 3.4

Для заданного множества подстановок $\boldsymbol{\sigma}^{(0)}, \boldsymbol{\sigma}^{(1)}, \ldots, \boldsymbol{\sigma}^{(k-1)}$ длины n *блоком управляемых подстановок* называется отображение $\mathbf{S}(i, j)$ такое, что для любого $i \in \{1, 2, \ldots, n\}$ и для каждого фиксированного значения j ($j = 0, 1, \ldots, k-1$) имеет место равенство $\mathbf{S}(i, j) = \boldsymbol{\sigma}^{(j)}(i)$.

В частности, для случая $k = 2^m$ параметр j может быть представлен двоичным вектором $V \in \mathrm{GF}(2)^m$, для которого $j = \alpha(V) = /V/ = v_1 + v_2 2^1 + \ldots + v_m 2^{m-1}$ и $\boldsymbol{\sigma}^{(j)} = \boldsymbol{\sigma}^{(/V/)}$.

Определение 3.5

Отображение $\boldsymbol{\Psi}$, соответствующее конкретному значению управляющего кода V, называется *модификацией* и обозначается как $\boldsymbol{\Psi}_V$.

Например, $\mathbf{S}_V$ является *модификацией* для $\mathbf{S}_{n/m}(X, V)$. Для каждого фиксированного значения вектора $V \in \mathrm{GF}(2)^m$ и $\mathbf{S}_V = \boldsymbol{\sigma}_V$, где $\boldsymbol{\sigma}_V = \boldsymbol{\sigma}^{(\alpha(V))}$ — фиксированная подстановка, соответствующая вектору V.

Определение 3.6

Две модификации $\boldsymbol{\Psi}_V$ и $\boldsymbol{\Psi}_{V'}$, соответствующие управляющим векторам V и $V' \neq V$, называются *эквивалентными*, если $\forall X$ выполняется равенство $\boldsymbol{\Psi}_V(X) = \boldsymbol{\Psi}_{V'}(X)$.

Определение 3.7

Модификация $\boldsymbol{\Psi}_V$ называется *уникальной*, если $\forall V' \neq V$ выполняется неравенство $\boldsymbol{\Psi}_V \neq \boldsymbol{\Psi}_{V'}$.

То есть, модификация называется *уникальной*, если не существует другой эквивалентной ей модификации.

Итак, в обозначении $\mathbf{S}_{n/m}$ первый параметр n указывает размерность преобразуемого двоичного вектора (размерность входа и размерность выхода совпадают), а второй параметр m — размерность управляющего вектора. Так, например, в раундовой функции DES каждый из восьми **S**-блоков, выполняющих в совокуп-

ности параллельное преобразование над 32-разрядным двоичным вектором входных данных, реализует блок управляемых подстановок $\mathbf{S}_{4/2}$, два стоящих рядом **S**-блока — блок $\mathbf{S}_{8/4}$, а все восемь — блок $\mathbf{S}_{32/16}$. В частности, раундовая функция в DES является управляемой подстановочной операцией с числом модификаций 2^{16}.

С ростом значений n и m аппаратная реализация базовых **S**-блоков усложняется, поскольку необходимо реализовать 2^m подстановок, каждая из которых принадлежит S_{2^n}. В программной реализации обычно используются таблицы, при этом размеры памяти в ЭВМ также ограничивают предельно допустимые значения параметров n и m. Разработанные в последнее десятилетие методы анализа свойств реализуемых отображений и преобразований в большинстве случаев опираются именно на возможность полного перебора значений входных и выходных параметров базовых **S**-блоков, что, в свою очередь, позволяет вычислять элементы секретного ключа по частям [44, 45, 86]. Поэтому представляют интерес отображения и преобразования, исключающие возможность такого перебора.

Рассмотрим перестановку

$$\boldsymbol{\pi} = \begin{pmatrix} 1 & 2 & \dots & n \\ \pi_1 & \pi_2 & \dots & \pi_n \end{pmatrix}.$$

Аппаратная реализация перестановки $\boldsymbol{\pi}$ заключается в проволочной коммутации разрядов входа и выхода, поэтому фиксированную перестановку часто называют *фиксированным коммутатором*. Поскольку каждый разряд входа и выхода перестановки $\boldsymbol{\pi}$ является элементом электрической цепи, фактически выполняется перемещение элементов множества GF(2) из разряда i в разряд π_i, т. е. перестановка $\boldsymbol{\pi}$ задает биективное подстановочное отображение (преобразование) $\boldsymbol{\sigma}_{\boldsymbol{\pi}}$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$. Операционный блок, реализующий данное преобразование, получил название **P**-блока, поэтому наряду с обозначением $\boldsymbol{\sigma}_{\boldsymbol{\pi}}$ будем также использовать обозначение $\mathbf{P}_{\boldsymbol{\pi}}$. Достоинством данной операции является простота реализации, а также отсутствие временной задержки ($t(\boldsymbol{\pi}) = 0$). Фиксированные коммутаторы (перестановки) активно используются практически в каждом блочном алгоритме (например, в DES $n = 32, 64$).

Определение 3.8

Подстановку $\sigma_{\boldsymbol{\pi}}$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$) будем называть *подстановкой с коммутацией разрядов*, для которой введем обозначение $\mathbf{P}_{\boldsymbol{\pi}}$, если $\forall i$ ($i = 1, 2, \dots, n$) $y_{\pi_i} \equiv x_i$, где $Y = \mathbf{P}_{\boldsymbol{\pi}}(X)$ и $X, Y \in \mathrm{GF}(2)^n$.

Заметим, что каждая подстановка $\mathbf{P}_{\boldsymbol{\pi}}$ реализует биективное преобразование, сохраняющее вес Хэмминга $\forall X \in \mathrm{GF}(2)^n$, а именно:

$$\mathbf{WH}(\mathbf{P}_{\boldsymbol{\pi}}(X)) = \mathbf{WH}(X). \tag{3.8}$$

Действительно

$$\mathbf{WH}(\mathbf{P}_{\boldsymbol{\pi}}(X)) = \sum_{j=1}^{n} y_j = \sum_{j=1}^{n} y_{\pi_j} = x_j = \mathbf{WH}(X),$$

где $y_j = y_{\pi_j} = x_i \ \forall j = 1, 2, \ldots, n$.

Очевидно, что тождественной перестановке $\mathbf{e} \in S_n$ соответствует тождественная подстановка $\mathbf{P}_{\mathbf{e}} = \mathbf{E} \in S_{2^n}$.

Определение 3.9

Подстановка $\mathbf{P}_{\boldsymbol{\pi}}^{-1} \overset{def}{=} (\mathbf{P}_{\boldsymbol{\pi}})^{-1}: (\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n)$ называется *обратной* для $\mathbf{P}_{\pi}$, если выполняется условие $\mathbf{P}_{\boldsymbol{\pi}}^{-1} \circ \mathbf{P}_{\boldsymbol{\pi}} = \mathbf{P}_{\boldsymbol{\pi}} \circ \mathbf{P}_{\boldsymbol{\pi}}^{-1} = \mathbf{E}$, где $\mathbf{E}$ — тождественное преобразование на $\mathrm{GF}(2)^n$.

Легко проверить, что для каждой подстановки $\mathbf{P}_{\boldsymbol{\pi}}$ существует ей обратная, а именно:

$$\mathbf{P}_{\boldsymbol{\pi}}^{-1} = \mathbf{P}_{\boldsymbol{\pi}^{-1}}. \tag{3.9}$$

Покажем, что $\mathbf{P}_{\pi}^{-1} \circ \mathbf{P}_{\pi} = \mathbf{E}$. Действительно, если $Y = \mathbf{P}_{\pi}^{-1}(X)$ и $Y' = \mathbf{P}_{\pi}(Y)$, тогда $\forall i = 1, 2, \ldots, n \quad y_i \equiv x_{\pi_i}$ и $y'_{\pi_i} \equiv y_i \equiv x_{\pi_i}$, поскольку для $\boldsymbol{\pi}^{-1}\ \pi_i \to i$ и $i \to \pi_i$ для $\boldsymbol{\pi}$. Но запись $y'_{\pi_i} \equiv x_{\pi_i} \forall i = 1, 2, \ldots, n$ означает, что $Y' = X$. Таким образом

$$(\mathbf{P}_{\boldsymbol{\pi}}^{-1} \circ \mathbf{P}_{\boldsymbol{\pi}})(X) = \mathbf{P}_{\boldsymbol{\pi}}(\mathbf{P}_{\boldsymbol{\pi}^{-1}}(X)) = \mathbf{P}_{\boldsymbol{\pi}}(Y) = Y' = X = \mathbf{E}(X).$$

Аналогично доказывается $\mathbf{P}_{\pi} \circ \mathbf{P}_{\pi}^{-1} = \mathbf{E}$.

Но тогда множество $\boldsymbol{G} = \{\mathbf{P}_{\pi}\}$ является подгруппой в S_{2^n}, при этом свойства подстановок $\mathbf{P}_{\pi}$ существенным образом зависят от свойств перестановок $\boldsymbol{\pi}$. Достаточно отметить, что подгруппа преобразований $\boldsymbol{G} = \{\mathbf{P}_{\pi}\}$ изоморфна группе S_n. Соответственно $|\boldsymbol{G}| = n!$. Различие между $\mathbf{P}_{\boldsymbol{\pi}} \in \boldsymbol{G}$ и $\boldsymbol{\pi} \in S_n$ с точки зрения изучения математических свойств заключается лишь в том, что в первом случае входными и выходными значениями являются бинарные векторы из $\mathrm{GF}(2)^n$, а во втором — номера разрядов j ($j = 1, 2, \ldots, n$). При этом между номерами разрядов j и двоичными векторами E_j ($E_j \in \mathrm{GF}(2)^n$, $\mathbf{WH}(E_j) = 1$, $e_j = 1$) можно установить взаимно однозначное соответствие, а именно $E_j \leftrightarrow j$, при этом достаточно очевидно, что $E_s = \mathbf{P}_{\pi}(E_j) \Leftrightarrow s = \boldsymbol{\pi}(j)$.

Покажем, что подгруппа $\boldsymbol{G} \subset S_{2^n}$ является *группой изоморфизмов* кольца $(\mathrm{GF}(2)^n, \oplus, \otimes)$ на себя [22]. То есть $\forall X, Y \in \mathrm{GF}(2)^n$ и $\forall \mathbf{P}_{\boldsymbol{\pi}} \in \boldsymbol{G}$ справедливы равенства:

$$\mathbf{P}_\pi(X \oplus Y) = \mathbf{P}_\pi(X) \oplus \mathbf{P}_\pi(Y). \tag{3.10}$$

$$\mathbf{P}_\pi(X \otimes Y) = \mathbf{P}_\pi(X) \otimes \mathbf{P}_\pi(Y). \tag{3.11}$$

$$\mathbf{P}_\pi(E_0) = E_0. \tag{3.12}$$

Действительно, если $Z = \mathbf{P}_\pi(X)$, $Z' = \mathbf{P}_\pi(Y)$ и $Z'' = \mathbf{P}_\pi(X \oplus Y)$, тогда $\forall i = 1, 2, \ldots, n$ $z_{\pi_i} \equiv x_i$, $z'_{\pi_i} \equiv y_i$ и $z''_{\pi_i} \equiv (X \oplus Y)_i \equiv x_i \oplus y_i \equiv z_{\pi_i} \oplus z'_{\pi_i}$. То есть $\mathbf{P}_\pi(X \oplus Y) = \mathbf{P}_\pi(X) \oplus \mathbf{P}_\pi(Y)$. Если дополнительно $Z''' = \mathbf{P}_\pi(X \otimes Y)$, тогда $\forall i = 1, 2, \ldots, n$ $z'''_{\pi_i} \equiv (X \otimes Y)_i \equiv x_i \otimes y_i \equiv z_{\pi_i} \otimes z'_{\pi_i}$. Следовательно, $\mathbf{P}_\pi(X \otimes Y) = \mathbf{P}_\pi(X) \otimes \mathbf{P}_\pi(Y)$.

Поскольку формула (3.8), а именно $\mathbf{WH}(\mathbf{P}_\pi(X)) = \mathbf{WH}(X)$, верна $\forall X \in \mathrm{GF}(2)^n$, она верна и для вектора E_0: $\mathbf{WH}(\mathbf{P}_\pi(E_0)) = \mathbf{WH}(E_0) = 0$. Следовательно, $\mathbf{P}_\pi(E_0) = E_0$. То есть действительно $\mathbf{P}_\pi$ является изоморфизмом кольца $(\mathrm{GF}(2)^n, \oplus, \otimes)$ на себя.

Пусть даны k фиксированных перестановок $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(k-1)}$ длины n, принадлежащих S_n. Аналогично операционному блоку управляемых подстановок $\mathbf{S}(i, j)$, параметрическое преобразование $\pi(i, j)$, заключающееся в применении перестановки $\pi^{(j)}$ к аргументу i, где $i \in \{1, 2, \ldots, n\}$, можно рассматривать в качестве операционного блока управляемых перестановок.

Определение 3.10

Для заданного множества перестановок $\pi^{(0)}, \pi^{(1)}, \ldots, \pi^{(k-1)}$ длины n *блоком управляемых перестановок* называется отображение $\pi(i, j)$ такое, что $\forall i \in \{1, 2, \ldots, n\}$ и для каждого фиксированного значения j ($j = 0, 1, \ldots, k-1$) имеет место равенство $\pi(i, j) = \pi^{(j)}(i)$.

В частности, для случая $k = 2^m$ параметр j может быть представлен двоичным вектором $V \in \mathrm{GF}(2)^m$, а именно $j = \alpha(V)$, например:

$$j = \alpha(V) = /V/ = v_1 + v_2 2^1 + \ldots + v_m 2^{m-1} \text{ и } \boldsymbol{\sigma}^{(j)} = \boldsymbol{\sigma}^{(/V/)}.$$

При этом отображение $\boldsymbol{\pi}(i, V) \overset{def}{=} \boldsymbol{\pi}^{(\alpha(V))}(i)$ для каждого фиксированного значения $V \in \mathrm{GF}(2)^m$ и $\forall i \in \{1, 2, \ldots, n\}$ также называется *блоком управляемых перестановок*, а $\pi_V = \pi^{(\alpha(V))}$, соответственно, модификацией отображения $\pi(i, V)$.

Поскольку каждая перестановка $\pi^{(j)}$ задает биективное подстановочное преобразование $\mathbf{P}_{\pi(j)}$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$, следовательно, *блок управляемых перестановок* $\pi(i, j)$ формирует *блок управляемых подстановок* $\mathbf{P}(X, j) = \mathbf{P}^{(j)}(X)$, где $X \in \mathrm{GF}(2)^n$. Для случая $j = \alpha(V)$ вместо $\mathbf{P}_{\pi(j)} = \mathbf{P}_{\pi(\alpha(V))}$ будем использовать обозначение $\mathbf{P}_V$.

Однако сформированный таким образом операционный *блок управляемых подстановок* $\mathbf{P}(X, V)$ будем называть *блоком управляемых перестановок*, подчеркивая тем самым специфические особенности формирования такого блока. Приве-

денные ниже определения позволяют увидеть существующие различия между $\pi_{n/m}(i, V)$ и $\mathbf{P}_{n/m}(X, V)$.

Определение 3.11

Отображение $\pi_{n/m}(i, V)$ $(\{1, 2, ..., n\} \times \mathrm{GF}(2)^m \to \{1, 2, ..., n\})$, являющееся объединением 2^m перестановок $\pi_V = \pi^{(j)} \in S_n$, называется $\pi_{n/m}$ *блоком управляемых перестановок*, если для каждого фиксированного значения $V \in \mathrm{GF}(2)^m$ задана некоторая перестановка $\pi_V = \pi^{(\alpha(V))} \in S_n$, такая, что $\pi_{n/m}(i, V) = \pi_V(i) = \pi^{(\alpha(V))}(i)$.

Определение 3.12

Отображение $\mathbf{P}_{n/m}(X, V)$ $(\mathrm{GF}(2)^n \times \mathrm{GF}(2)^m \to \mathrm{GF}(2)^n)$, являющееся объединением 2^m подстановок $\mathbf{P}_{\pi(j)} \in S_{2^n}$, называется $\mathbf{P}_{n/m}$ *блоком управляемых перестановок*, если для каждого фиксированного значения $V \in \mathrm{GF}(2)^m$ задана некоторая перестановка $\pi_V = \pi^{(\alpha(V))} \in S_n$, такая, что $\mathbf{P}_{n/m}(X, V) = \mathbf{P}_V(X) = \mathbf{P}_{\pi^{(\alpha(V))}}(X) = \mathbf{P}_{\pi_V}(X)$.

В необходимых случаях наряду с обозначениями $\mathbf{P}_{n/m}(X, V)$ и $\pi_{n/m}(i, V)$ будем использовать обозначение $\mathbf{P}(X, V)$ и $\pi(i, V)$ или $\mathbf{P}^{(V)}(X)$ и $\pi^{(V)}(i)$.

Модификация $\mathbf{P}_V$ называется *уникальной*, если $\forall\ V' \neq V$ выполняется неравенство $\mathbf{P}_V \neq \mathbf{P}_{V'}$. Очевидно, что число различных модификаций $\mathbf{P}_V$ не превосходит значение

$$\min(n!, 2^m).$$

Понятие блока $\pi_{n/m}$ введено для изучения свойств операционного блока $\mathbf{P}_{n/m}$. При этом все вышесказанное в отношении зависимости свойств подстановки $\mathbf{P}_\pi$ от свойств перестановки π распространяется и на блоки $\mathbf{P}_{n/m}$ и $\pi_{n/m}$. В частности, для блока $\mathbf{P}_{n/m}$ и $\forall v \in \mathrm{GF}(2)^m$, $\forall X, Y \in \mathrm{GF}(2)^n$ формулы (3.8), (3.10) и (3.11) имеют вид:

$$\mathbf{WH}(\mathbf{P}(X, v)) = \mathbf{WH}(X); \tag{3.13}$$

$$\mathbf{P}(X, v) \oplus \mathbf{P}(Y, v) = \mathbf{P}(X \oplus Y, v); \tag{3.14}$$

$$\mathbf{P}(X, v) \otimes \mathbf{P}(Y, v) = \mathbf{P}(X \otimes Y, v). \tag{3.15}$$

Соответственно, $E_s = \mathbf{P}_V(E_j) \Leftrightarrow s = \pi_V(j)$, где $\mathbf{P}_V$ и π_V — модификации блоков $\mathbf{P}_{n/m}$ и $\pi_{n/m}$.

Поскольку для каждого фиксированного значения $V \in \mathrm{GF}(2)^m$ блок управляемых перестановок $\mathbf{P}_{n/m}$ реализует биективное преобразование $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$, для криптографических приложений представляет интерес вопрос построения и использования операционного блока, реализующего все *обратные* (по первому параметру) *преобразования* $\mathbf{P}_{n/m}^{-1} = (\mathbf{P}_{n/m})^{-1}$. Очевидно, что такой операционный блок существует и также является *блоком управляемых перестановок*. Действительно, так как каждая модификация $\mathbf{P}_V$ является σ-подстановкой с коммутацией

разрядов $\mathbf{P}_{\pi_V}$, следовательно, для нее существует обратная σ-подстановка $\mathbf{P}_{\pi_V}^{-1} = \mathbf{P}_{\pi_V^{-1}}$, также являющаяся подстановкой с коммутацией разрядов, для которой введем обозначение $\mathbf{P}_V^{-1}$. Тогда объединение 2^m σ-подстановок $\mathbf{P}_V^{-1} \in S_{2^n}$ и является искомым блоком управляемых перестановок $(\mathbf{P}_{n/m})^{-1}$.

Определение 3.13

Отображение $\mathbf{P}_{n/m}^{-1}(X, V)$ $(\mathrm{GF}(2)^n \times \mathrm{GF}(2)^m \to \mathrm{GF}(2)^n)$, являющееся объединением 2^m подстановок $\mathbf{P}_V^{-1} = \mathbf{P}_{\pi_V^{-1}} \in S_{2^n}$, называется *обратным блоком управляемых перестановок по отношению к блоку* $\mathbf{P}_{n/m}(X, V)$, или просто *обратным преобразованием*, если блок $\mathbf{P}_{n/m}(X, V)$ $(\mathrm{GF}(2)^n \times \mathrm{GF}(2)^m \to \mathrm{GF}(2)^n)$ является объединением 2^m подстановок $\mathbf{P}_V = \mathbf{P}_{\pi_V} \in S_{2^n}$.

Действительно, $\forall \mathbf{P}_V \in \mathbf{P}_{n/m}$ $\mathbf{P}_V \circ \mathbf{P}_V^{-1} = \mathbf{P}_{\pi_V} \circ \mathbf{P}_{\pi_V^{-1}} = \mathbf{P}_{(\pi_V \circ \pi_V^{-1})} = \mathbf{P}_{\mathbf{e}} = \mathbf{E}$, где $\mathbf{e}$ — тождественная перестановка, а $\mathbf{E}$ — тождественное преобразование. И $\forall V \in \mathrm{GF}(2)^m$

$$(\mathbf{P}_{n/m} \circ \mathbf{P}_{n/m}^{-1})(X,V) = \mathbf{P}_{n/m}^{-1}(\mathbf{P}_{n/m}(X,V),V) = \mathbf{P}_V^{-1}(\mathbf{P}_V(X)) = (\mathbf{P}_V \circ \mathbf{P}_V^{-1})(X) = \mathbf{E}(X) = X.$$

В дальнейшем на схемах, отражающих алгоритм блочного шифрования (или его фрагмент), для операционного блока управляемых перестановок $\mathbf{P}_{n/m}$ будем использовать обозначение, представленное на рис. 3.1.

Рис. 3.1. Блок управляемых перестановок $\mathbf{P}_{n/m}$

При синтезе операционных блоков $\mathbf{P}_{n/m}$ для их последующего использования в алгоритмах блочного шифрования определяющим является следующий набор параметров:

- длина преобразуемого блока данных n и разрядность управляющего кода m;
- число различных модификаций и уникальность модификации для каждого значения управляющего кода;
- время выполнения преобразования $t(\mathbf{P}_{n/m})$ — *см. разд. 3.2*;
- сложность аппаратной реализации $C_\Omega(\mathbf{P}_{n/m})$ — *см. разд. 3.2*;
- сложность аппаратной реализации обратного преобразования — *см. разд. 3.2*;

- выполнение условия равновероятного смещения — *см. разд. 3.3*;
- порядок блока h — *см. разд. 3.3*;
- формирование модификаций, относящихся к определенному классу (например, полноцикловые перестановки, инволюции, не содержащие единичных циклов, и др.) — *см. разд. 3.3*;

и т. д.

В данной главе рассматриваются различные конструкции $\mathbf{P}_{n/m}$-блоков, а также анализируются перечисленные выше параметры с целью выбора оптимального варианта. Кроме того, в *разд. 3.4* для специальных линейных характеристик в методе линейного криптоанализа, предложенного в 1993 году Мацуи (Matsui), подробно обсуждаются способы их вычисления.

В *разд. 3.5* рассматриваются вопросы, связанные с формированием управляющих векторов для блоков управляемых перестановок.

Замечание

В прикладных задачах, связанных с обработкой данных на вычислительной технике, наибольший интерес представляет случай, когда n является степенью числа 2. Поэтому, в дальнейшем, если не оговорено иначе, будем полагать, что $n = 2^k$.

3.2. Разработка операционных блоков управляемых перестановок с заданными параметрами скорости и схемотехнической сложности

На процесс синтеза алгоритмов блочного шифрования существенное влияние оказывают параметры скорости работы алгоритма и сложности реализации алгоритма. На решение одной и той же задачи можно затратить больше времени, используя меньшее количество ресурсов, и наоборот, увеличив ресурсы, можно сократить время решения задачи. Понятия времени и сложности реализации неразрывно связаны с *теорией сложности вычислений*, основоположником которой является Клод Шеннон [35, 36], показавший важность булевой алгебры для анализа и синтеза релейных переключательных схем. Моделирование вычислительных процессов часто выполняется в виде логических (комбинационных) схем, для реализации которых используется некоторый базис простейших логических элементов *(вентилей)*, реализующих логические операции. Базисы бывают полные и неполные. Например, логические операции "И" (&), "ИЛИ" ($\vee$), "НЕ" ($\bar{\ }$) образуют полный базис, позволяющий реализовать любую булеву функцию. То есть множество логических операций $\mathbf{\Omega} = \{\&, \vee, \bar{\ }\}$ является полным базисом. $\mathbf{\Omega} = \{\oplus, \&, 1\}$ и $\mathbf{\Omega} = \{\oplus, \&, \vee, \bar{\ }\}$ — другие варианты полных базисов [34].

Определение 3.14

Под сложностью схемотехнической реализации (комбинационной сложностью) $C_\Omega(\mathbf{F})$ будем понимать количество *вентилей*, реализующих схему **F**.

Необходимо отметить, что комбинационная сложность существенным образом зависит от базиса **Ω**. Кроме того, одна и та же логическая схема может иметь различные реализации. В настоящей книге оценка сложности рассматриваемых комбинационных схем дана для базиса $\mathbf{\Omega} = \{\&, \vee, \bar{\ }\}$.

Замечание

К количественным оценкам схемотехнической реализации необходимо относиться только как к качественным характеристикам, поскольку для получения точных оценок необходимы специальные исследования с учетом постоянно развивающейся элементной базы.

Для оценки скоростных параметров схемы **F** обычно используют время, затрачиваемое на выполнение самой медленной базовой операции, которое называют *тактом задержки* τ. Например, $\tau_\Omega = t_\&$ — для базиса $\Omega = \{\&, \vee, \bar{\ }\}$ и $\tau_\Omega = t_\oplus$ — для $\Omega = \{\oplus, \&, 1\}$.

Определение 3.15

Максимальная длина маршрута $d_\Omega(\mathbf{F})$ (маршрут — число *вентилей*, которое проходит один из сигналов от входа до выхода) называется *глубиной схемы*.

Говорят, что схема **F** имеет d_Ω тактов задержки, если глубина схемы равна d_Ω, при этом *время выполнения* схемы $t_\Omega(\mathbf{F}) \le d_\Omega \cdot \tau_\Omega$. В необходимых случаях для *времени выполнения* схемы можно получить точное значение, являющееся максимальным значением, вычисленным по всем маршрутам.

3.2.1. Первый *(формальный)* вариант синтеза блока управляемых перестановок $\mathbf{P}_{n/m}$

Данный вариант представлен на рис. 3.2. Схема работает следующим образом.

Двоичный дешифратор $\mathbf{D}_m$, на вход которого поступает m разрядный управляющий вектор $V = (v_1, v_2, \ldots, v_m)$, формирует двоичный вектор $U = \mathbf{D}_m(V)$ длины 2^m с весом Хэмминга $\mathbf{WH}(U) = 1$, т. е. значение 1 вырабатывается только в одном разряде, а именно:

$$u_s = \begin{cases} 1, & \text{если } s = 1 + v_1 + v_2 2 + \ldots + v_m 2^{m-1} \\ 0, & \text{в остальных случаях} \end{cases}.$$

Коммутация разрядов входа и разрядов выхода по заданной фиксированной перестановке $\boldsymbol{\pi}^{(s-1)}$ реализуется в соответствии со схемой, представленной на рис. 3.3, *а*, в которой разряды вектора X поступают на вход n логических элемен-

тов "И" одновременно с управляющим сигналом u_s. Коммутация выполняется только в том случае, если значение одноразрядного управляющего сигнала $u_s = 1$.

Рис. 3.2. Формальная схема синтеза блока управляемых перестановок

Рис. 3.3. Схема реализации управляемого выбора фиксированной модификации $\pi^{(s-1)}$: *а* — общий вид, *б* — пример для блока $\pi_{4/4}$

Пусть, например, в блоке управляемых перестановок $\boldsymbol{\pi}_{4/4}$ вектору $V = (1,1,1,0)$ соответствует модификация

$$\boldsymbol{\pi}^{(7)} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 1 & 3 & 2 \end{pmatrix}.$$

Реализация модификации $\boldsymbol{\pi}^{(7)}$ приведена на рис. 3.3, *б*.

Схему, представленную на рис. 3.2, будем называть *матричной структурой*. Анализируя скоростные параметры такой реализации, видим, что $t(\mathbf{P}_{n/m}) = t(\mathbf{D}_m) + t_{\&}$, где $t(\mathbf{P}_{n/m})$ — время выполнения схемы, $t(\mathbf{D}_m)$ — время формирования управляющего вектора ($t(\mathbf{D}_m) \approx \log_2 m \cdot t_{\&}$), а $t_{\&}$ — время выполнения операции "И". При синтезе конкретных схем алгоритмов преобразования данных в тех случаях, когда можно обеспечить заблаговременное формирование вектора U, время задержки меньше $(\log_2 m + 1)\tau$, где $\tau = t_{\&}$. В общем случае

$$\tau \le t(\mathbf{P}_{n/m}) \le (\log_2 m + 1)\tau. \tag{3.16}$$

Сложность реализации блока $\mathbf{P}_{n/m}$ с матричной структурой:

$$C_\Omega(\mathbf{P}_{n/m}) = \min(n2^m, n^2) + C_\Omega(\mathbf{D}_m). \tag{3.17}$$

Поясним формулу для $C_\Omega(\mathbf{P}_{n/m})$. Значение $n2^m$ получается следующим образом: для каждой из 2^m модификации $\boldsymbol{\pi}^{(s)}$ (рис. 3.3, *а*) необходимо n вентилей, реализующих конъюнкцию &.

Однако при $n \ge 2^m$ достаточно использовать n^2 вентилей $f_{\&}^{(ij)}(u)$, индексы i и j которых обозначают конкретную пару разрядов входа и выхода, причем вентиль $f_{\&}^{(ij)}$ "замыкает" цепь, если $u = u^{(ij)} = 1$, где

$$u^{(ij)} = \sum_{s=1}^{2^m} u_s \cdot \xi^{(ij)}(\pi^{(s)}) \tag{3.18}$$

и

$$\xi^{(ij)}(\pi^{(s)}) = \begin{cases} 1, & \text{если } \pi^{(s)}(i) = j \\ 0, & \text{в противном случае} \end{cases}.$$

Поскольку в сумме (3.18) $u_s = 1$ только для одного s, следовательно, дополнительных вентилей типа $f_\vee$ для реализации функции (3.18) не требуется.

Сложность реализации двоичного дешифратора, согласно [34], находится в пределах $2^m + m - 2 \le C_\Omega(\mathbf{P}_{n/m}) \le 2^m + (m-2)2^{\lceil m/2 \rceil}$, где $\lceil m/2 \rceil$ — округление до ближайшего целого числа, большего либо равного $m/2$. Поэтому комбинационная сложность реализации блока управляемых перестановок $\mathbf{P}_{n/m}$ находится в пределах

$$\min(n \cdot 2^m, n^2) + 2^m + m - 2 \le C_\Omega(\mathbf{P}_{n/m}) \le \min(n \cdot 2^m, n^2) + 2^m + (m-2)2^{\lceil m/2 \rceil}.$$

При $m \ge \log_2 n$ и больших n значение $C_\Omega(\mathbf{P}_{n/m}) \ge n^2 + 2^m + m - 2 \approx n^2 + 2^m$, что ведет к резкому усложнению и удорожанию электронной схемы блока.

Достоинством матричной структуры являются *переменные* скоростные параметры блока, выполняющего операцию перестановки с временем задержки не более чем за $\log_2 m + 1$ тактов задержки τ, причем минимальное время задержки равно τ. Для небольших значений n ($n \le 8$) и числа модификаций 2^m, сравнимого с n, такая реализация оправдана. Вопрос конкретных рекомендаций по использова-

нию этой технологии синтеза блоков с заданными параметрами рассматривается ниже. Здесь же необходимо отметить следующее.

Замечание

Свойства блока управляемых перестановок с матричной структурой зависят от конкретного набора заданных перестановок (модификаций), а также от их взаимного расположения, так как предметом аналитических исследований являются не только значения векторов входа и выхода, но и значения управляющего вектора.

Поэтому с ростом значения m возрастает сложность формирования блока $\mathbf{P}_{n/m}$, обладающего необходимыми свойствами.

3.2.2. Второй *(экономичный)* вариант реализации блока управляемых перестановок $\mathbf{P}_{n/m}$

Очевидно, что композиция двух или более **P**-блоков также является **P**-блоком, при этом общее время задержки не превосходит сумму задержек на каждом блоке. Используя данный композиционный принцип, как будет показано далее, можно формировать блоки с приемлемой схемотехнической сложностью реализации и небольшим числом тактов задержки.

Необходимо отметить, что в электротехнике широкое распространение получили так называемые переключаемые сети, в которых каждый бит вектора V управляет *элементарным переключателем*, а вся сеть реализует биективную коммутацию между разрядами входа и разрядами выхода для каждого значения управляющего вектора. В качестве примера на рис. 3.4 представлена четырехслойная переключаемая *R-сеть*, название и рисунок которой заимствованы из работы Порца [109].

При такой структуре каждое значение управляющего вектора V реализует взаимно однозначную коммутацию между разрядами входа и разрядами выхода, т. е. формирует перестановку $\boldsymbol{\pi}_V$ длины n. Соответственно, переключаемая сеть с m-разрядным управляющим вектором V ($V \in \mathrm{GF}(2)^m$) формирует 2^m перестановок и является одновременно $\boldsymbol{\pi}_{n/m}$-блоком и $\mathbf{P}_{n/m}$-блоком. Поэтому переключаемые сети также называют *перестановочными* сетями. Заметим, что в отличие от *первого* варианта синтеза блоков управляемых перестановок с матричной структурой, в котором управляющий вектор лишь *выбирает* фиксированную перестановку $\boldsymbol{\pi}_V$, в *перестановочной* сети управляющий вектор ее *формирует*. Именно поэтому в определениях блоков $\mathbf{P}_{n/m}$ и $\boldsymbol{\pi}_{n/m}$ *(см. разд. 3.1)* отсутствует ссылка на способ формирования модификаций $\mathbf{P}_V$ и π_V.

В общем случае переключаемая сеть размера $n \times n$ имеет послойную структуру, в которой перед первым слоем и после последнего, а также между соседними слоями используются фиксированные коммутаторы, а каждый слой является объединением параллельно расположенных и не пересекающихся переключаемых сетей (блоков) меньшего размера. При такой конструкции время задержки равно сумме задержек на каждом слое.

Рис. 3.4. *R*-сеть с 16 разрядами входа/выхода

Рис. 3.5. Общий вид $\mathbf{P}_{n/m}$-блока с послойной структурой

Наибольший интерес представляют сети (рис. 3.5), в которых n является степенью числа 2, т. е. $n = 2^k$, а каждый слой является объединением $n/2$ параллельно расположенных элементарных переключателей $\mathbf{P}_{2/1}$. Для такого слоя, также являющегося блоком управляемых перестановок, мы будем использовать обозначение $\mathbf{L}_n$ или $\mathbf{L}_{n(V)}$, а сам слой будем называть *активным*. Термин "активный" используется в противовес фиксированной коммутации между такими слоями.

В простейшем блоке $\mathbf{P}_{2/1}$ (см. рис. 3.6, *а*), именуемом в электротехнике *элементарным переключателем*, в зависимости от значения управляющего бита v два бита информации либо переставляются $y_1 = x_2$ и $y_2 = x_1$ (например, при значении управляющего бита $v = 1$), либо не переставляются $y_1 = x_1$ и $y_2 = x_2$ ($v = 0$). Очевидно, что $\mathbf{P}_{2/1}^{-1} = \mathbf{P}_{2/1}$. Соответственно блок $\mathbf{L}_n$ реализует подстановочное преобразование $Y = \mathbf{L}_{n(V)}(X)$, где $X, Y \in \mathrm{GF}(2)^n$, $V \in \mathrm{GF}(2)^{n/2}$, причем $\forall i$ ($i = 1, 2, \ldots, n/2$) $y_{2i-1} = x_{2i-1+v_i}$ и $y_{2i} = x_{2i-v_i}$. Очевидно также, что $\mathbf{L}_{n(V)}^{-1} = \mathbf{L}_{n(V)}$.

Для рассматриваемой схемы число слоев $l = \dfrac{2m}{n}$, время выполнения преобразования $t(\mathbf{P}_{n/m}) = l\tau$, а сложность реализации $C_\Omega(\mathbf{P}_{n/m}) = mC_\Omega(\mathbf{P}_{2/1})$.

3.2.3. Построение обратного преобразования $\mathbf{P}_{n/m}^{-1}$

Для *формального варианта* построения обратного преобразования $\mathbf{P}_{n/m}^{-1}$ достаточно к каждой модификации $\mathbf{P}_V$ взять обратную модификацию $\mathbf{P}_V^{-1}$ и объединить их в блок. Для блоков $\mathbf{P}_{n/m}$ с послойной структурой (см. рис. 3.5) с теоретической точки зрения существует очевидное схемотехнически простое решение, соответствующее *второму варианту*: в блоке $\mathbf{P}_{n/m}$ достаточно поменять местами вход и выход. Если представить управляющий вектор V в виде конкатенации l векторов размера $\dfrac{m}{l}$ ($V = V_1 \parallel V_2 \parallel \ldots \parallel V_l$, где $V_i \in \mathrm{GF}(2)^{n/2}$), и учитывая возможность декомпозиции блока $\mathbf{P}_{n/m}$ по слоям, а именно:

$$\mathbf{P}_{n/m(V)} = \mathbf{P}_{\boldsymbol{\pi}^{(0)}} \circ \mathbf{L}_{n(V_1)} \circ \mathbf{P}_{\boldsymbol{\pi}^{(1)}} \circ \mathbf{L}_{n(V_2)} \circ \mathbf{P}_{\boldsymbol{\pi}^{(2)}} \circ \ldots \circ \mathbf{P}_{\boldsymbol{\pi}^{(l-1)}} \circ \mathbf{L}_{n(V_l)} \circ \mathbf{P}_{\boldsymbol{\pi}^{(l)}}, \tag{3.19}$$

где $\mathbf{L}_{n(V_i)}$ — i-й слой блоков $\mathbf{P}_{2/1}$, *обратное преобразование* $\mathbf{P}_{n/m(V)}^{-1}$ имеет вид:

$$\mathbf{P}_{n/m(V)}^{-1} = (\mathbf{P}_{\boldsymbol{\pi}^{(l)}})^{-1} \circ \mathbf{L}_{n(V_l)}^{-1} \circ (\mathbf{P}_{\boldsymbol{\pi}^{(l-1)}})^{-1} \circ \ldots \circ (\mathbf{P}_{\boldsymbol{\pi}^{(1)}})^{-1} \circ \mathbf{L}_{n(V_1)}^{-1} \circ (\mathbf{P}_{\boldsymbol{\pi}^{(0)}})^{-1}. \tag{3.20}$$

Так как $\mathbf{L}_{n(V_l)}^{-1} = \mathbf{L}_{n(V_l)}$ и $(\mathbf{P}_{\boldsymbol{\pi}^{(j)}})^{-1} = \mathbf{P}_{\boldsymbol{\pi}^{(j)}}^{-1}$, следовательно

$$\mathbf{P}_{n/m(V)}^{-1} = \mathbf{P}_{\boldsymbol{\pi}^{(l)}}^{-1} \circ \mathbf{L}_{n(V_l)} \circ \mathbf{P}_{\boldsymbol{\pi}^{(l-1)}}^{-1} \circ \ldots \circ \mathbf{P}_{\boldsymbol{\pi}^{(1)}}^{-1} \circ \mathbf{L}_{n(V_1)} \circ \mathbf{P}_{\boldsymbol{\pi}^{(0)}}^{-1}. \tag{3.21}$$

Получаем, что $\mathbf{P}^{-1}_{n/m(V)} = \mathbf{P}'_{n/m(U)}$, причем в блоке $\mathbf{P}'_{n/m(U)}$ управляющий вектор U имеет следующий вид:

$$U = V_l \parallel V_{l-1} \parallel \ldots \parallel V_1,$$

то есть конкатенация векторов размера $\frac{m}{l}$ имеет обратный порядок по сравнению с вектором V. Поэтому для обеспечения возможности использования формального обозначения $\mathbf{P}^{-1}_{n/m(V)}$ примем следующее соглашение, при котором обратное преобразование $\mathbf{P}^{-1}_{n/m(V)}$ выполняется при том же самом значении управляющего вектора $V = V_1 \parallel V_2 \parallel \ldots \parallel V_l$.

Соглашение

В блоках $\mathbf{P}^{-1}_{n/m}$ с послойной структурой используется нумерация блоков $\mathbf{P}_{2/1}$ *по слоям слева направо снизу вверх.*

Действительно, при таком соглашении сначала в управлении блоком $\mathbf{P}^{-1}_{n/m(V)}$ участвует вектор V_l, затем V_{l-1} и т. д. Необходимо отметить, что блок $\mathbf{P}^{-1}_{n/m}$ наследует все основные свойства, присущие блоку $\mathbf{P}_{n/m}$ (уникальность модификаций, свойство равновероятного смещения и порядок блока, сохранение цикловой структуры, как показано в *разд. 3.3*). Правомерность такого утверждения следует непосредственно либо из общих свойств перестановок (уникальность модификаций, одинаковая цикловая структура), либо из инвариантности определений относительно входа и выхода (равновероятное смещение, порядок блока). Поэтому в криптографических приложениях блок $\mathbf{P}^{-1}_{n/m}$ целесообразно использовать наравне с блоком $\mathbf{P}_{n/m}$, как, например, в схеме патента РФ № 2140714 [1].

3.3. Синтез блоков управляемых перестановок с заданными алгебраическими и вероятностно-статистическими свойствами

3.3.1. Блоки равновероятного смещения и блоки порядка *h*

Определение 3.16

Блок управляемых перестановок $\mathbf{P}_{n/m}$ удовлетворяет условию *равновероятного смещения*, если при равновероятном распределении всех модификаций блока произвольно выбранный входной бит x_i, $i \in \{1, 2, \ldots, n\}$ в результате преобразования может оказаться на выходе в любом из n двоичных разрядов с одинаковой вероятностью.

Применительно к блоку $\pi_{n/m}$ данное условие определяется выражением $\Pr_V\{\boldsymbol{\pi}_{n/m}(i) = j\} = \dfrac{1}{n}$ $\forall i, j \in \{1, 2, \ldots, n\}$, где $\Pr_V$ — вероятность события $\{\pi_{n/m}(i) = j\}$, вычисленная по всем $V \in \mathbf{V}$.

Определение 3.17

Блок $\mathbf{P}_{n/m}$ называется блоком h-го *порядка*, если он позволяет переместить любые h битов на входе в любые h разрядов на выходе [18]. Для блока h-го *порядка* будем использовать обозначение $\mathbf{P}_{n/m}^{(h)}$.

В терминах блока $\pi_{n/m}$ данное определение равносильно следующему.

Определение 3.18

Блок $\pi_{n/m}$ называется блоком h-го *порядка*, если для любых, различающихся между собой, элементов $i_1, i_2, \ldots, i_h \in \{1, 2, \ldots, n\}$, и для любых, различающихся между собой, элементов $j_1, j_2, \ldots, j_h \in \{1, 2, \ldots, n\}$ существует $\pi_V \in \pi_{n/m}$ такая, что

$$\pi_V = \begin{pmatrix} i_1 & i_2 & \ldots & i_h \ldots & \ldots & \ldots \\ j_1 & j_2 & \ldots & j_h \ldots & \ldots & \ldots \end{pmatrix}.$$

Для блока $\pi_{n/m}$ h-го *порядка* будем использовать обозначение $\boldsymbol{\pi}_{n/m}^{(h)}$.

Очевидно, что $0 \le h \le n$.

Теорема 3.1

Пусть $\pi_{n/m}$ — блок равновероятного смещения. Тогда число модификаций кратно n, причем, если $n = 2^k$, тогда, во-первых, $m \ge k$; а во-вторых, $\pi_{n/m}$ является, по крайней мере, блоком первого порядка, т. е. $h \ge 1$.

Доказательство

Действительно, по определению блока равновероятного смещения

$$\Pr_V\{\boldsymbol{\pi}_{n/m}(i) = j\} = \frac{1}{n} \text{ и } \Pr_V\{V = v\} = \frac{1}{2^m},$$

а, с другой стороны,

$$\Pr_V\{\boldsymbol{\pi}_{n/m}(i) = j\} = \sum_v \theta_v(i, j) \Pr_V\{V = v\} = \frac{1}{2^m} \sum_v \theta_v(i, j),$$

где

$$\theta_v(i, j) = \begin{cases} 1, & \text{если} \quad \boldsymbol{\pi}_v(i) = j \\ 0, & \text{если} \quad \boldsymbol{\pi}_v(i) \ne j \end{cases}.$$

Отсюда

$$\sum_{v} \theta_v(i,\ j) = \frac{2^m}{n}.$$

Так как для каждой модификации $\boldsymbol{\pi}_v$ характеристическая функция $\theta_v(i, j)$ принимает одно из двух значений (ноль либо единица), то в левой части последнего уравнения — целое число, а значит и $\frac{2^m}{n}$ — целое число. То есть число модификаций кратно n. Если при этом $n = 2^k$, тогда $\frac{2^m}{n} = 2^{m-k} \geq 1$.

Следовательно, $m - k = \log_2(2^{m-k}) > 0$. Первая часть утверждения доказана.

Вторая часть утверждения достаточно очевидна. Действительно, так как $\forall i, j \in \{1, 2, \ldots, n\}\quad p_V\{\boldsymbol{\pi}_{n/m}(i) = j\} = \frac{1}{n} > 0$, то существует $\boldsymbol{\pi}_v \in \boldsymbol{\pi}_{n/m}$ такая, что $\boldsymbol{\pi}_V = \begin{pmatrix} i & \ldots & \ldots \\ j & \ldots & \ldots \end{pmatrix}$. То есть $h \geq 1$.

Теорема доказана полностью.

С точки зрения практических приложений представляет интерес условие $h = n$. Как отмечалось выше, блоки управляемых перестановок, удовлетворяющие данному условию, называются *блоками максимального порядка*. Такие блоки реализуют любую перестановку, что равносильно выражению $S_n \subseteq \boldsymbol{\pi}_{n/m}$.

Рассмотрим три примера блоков с матричной структурой и один с каскадной.

Пример 3.1. Элементарный блок $\mathbf{P}_{2/1}$ (рис. 3.6, *а*). В базисе функциональных элементов $\Omega = \{\vee, \&, \bar{\ }\}$ реализация такого блока приведена на рис. 3.6, *б*, где треугольником обозначена инверсия сигнала, соответствующая логической операции "отрицание" ($\bar{\ }$). Очевидно, что для $t(\mathbf{P}_{2/1}) = \tau$, а сложность реализации без учета инверсии $C_\Omega(\mathbf{P}_{2/1}) \approx 4$. Блок обладает следующими свойствами:

- является блоком равновероятного смещения;
- реализует все перестановки (тождественную и транспозицию), т. е. является блоком *максимального* порядка;
- все модификации уникальны;
- реализует инволюцию, т. е. $\mathbf{P}_{2/1}^{-1} = \mathbf{P}_{2/1}$.

Пример 3.2. Блоки $\mathbf{P}_{4/m}$. Для $n = 4$ существуют 24 различные перестановки. Поэтому, например, с помощью матричной структуры блока $\mathbf{P}_{4/4}$ невозможно реализовать все перестановки ($2^4 = 16 < n! = 24$), но можно выбрать любые 16 уникальных модификаций. С другой стороны, с использованием блока $\mathbf{P}_{4/5}$ можно

реализовать все перестановки ($2^5 = 32 > 24$), однако для выполнения свойства уникальности необходимо обеспечить формирование только 24-х значений управляющего вектора. Например, для $\mathbf{P}_{4/4}$ быстродействие $\tau \le t(\mathbf{P}_{4/4}) \le 3\ \tau$, а сложность реализации матричной структуры $C_\Omega(\mathbf{P}_{4/4}) \approx n^2 + 2^m = 32$. Экономичная схема БРС $\mathbf{P}_{4/4}$ (рис. 3.7) имеет быстродействие 2τ и $C_\Omega(\mathbf{P}_{4/4}) \approx 16$.

Рис. 3.6. Блок управляемых перестановок $\mathbf{P}_{2/1}$ (*а*) и его практическая реализация (*б*)

Рис. 3.7. Структура блока $\mathbf{P}_{4/4}$

Пример 3.3. Блок циклического сдвига $\mathbf{T}_n$ (рис. 3.8). Рассматриваемый блок $\mathbf{T}_n$ с матричной структурой, реализующий операцию циклического сдвига входных

данных за один такт, является частным случаем блока $\mathbf{P}_{n/m}$ при $m = \log_2 n$ и состоит:

- из входных и выходных шин, обозначенных на схеме соответственно вертикальными и горизонтальными параллельными сплошными линиями;
- из двоичного дешифратора, формирующего сигналы управления ключами; шина управления ключами обозначена пунктирными линиями;
- в местах пересечения входной и выходной шин установлены ключи (логические элементы "И"), которые замыкают или размыкают цепь в зависимости от того, какой сигнал подается на вход управления ключом.

Рис. 3.8. Управляемый блок циклического сдвига $\mathbf{T}_n$ с матричной структурой

При $m = \log_2 n$ сложность реализации схемы $C_\Omega(\mathbf{T}_n) \approx n^2 + 2^m = n^2 + n$. Блок $\mathbf{T}_n$ является блоком равновероятного смещения первого порядка с минимально возможной длиной управляющего вектора, причем все модификации уникальны.

Существуют и другие конструкции блоков равновероятного смещения.

Пример 3.4. В работе [18] наряду с *матричной* структурой блока $\mathbf{T}_n$ рассматривалась схемотехнически экономичная *каскадная* структура (рис. 3.9), являющаяся аналогом переключаемой сети, со следующими особенностями:

- блок состоит из $m = \log_2 n$ слоев, каждый из которых включает n блоков $\mathbf{P}_{2/1}$, размещенных параллельно, причем в каждом блоке $\mathbf{P}_{2/1}$ используется только один выход ($C_\Omega(\mathbf{P}_{2/1}) \approx 2$), соединенный с двумя входами разных блоков $\mathbf{P}_{2/1}$ следующего слоя;
- каждый слой управляется одним битом управляющего вектора, причем s-й слой $\mathbf{T}^s_{n/1}$ выполняет циклический сдвиг данных на значение $v_s 2^{s-1}$.

Рис. 3.9. Каскадная структура блока циклического сдвига $\mathbf{T}_n$

Суммарная величина r циклического сдвига вычисляется по формуле:

$$r = v_1 + 2v_2 + 4v_3 + \qquad + 2^{m-1}v_m. \tag{3.22}$$

Сложность реализации каскадной структуры $C_\Omega(\mathbf{T}_n) \approx 2n \log_2 n$, время выполнения $t(\mathbf{T}_n) = \tau \log_2 n$.

Прежде чем переходить к описанию конкретных схем реализации блоков равновероятного смещения, блоков максимального порядка и блоков h-го порядка, обратим внимание на переключаемые сети Клоса [39, 40, 58], поскольку большинство из рассматриваемых далее схем блоков управляемых перестановок, имеющих самостоятельные названия, являются сетями Клоса.

Сети Клоса

В описании переключаемой трехуровневой сети Клоса, названной по фамилии своего автора, под блоком размера $r \times r$ подразумевается коммутатор, имеющий

r входов, r выходов и позволяющий выполнять произвольные коммутации между входами и выходами. В первой половине XX века в телефонии аналогом такого блока, в сущности, являлся коммутационный узел (штырьковый коммутатор), причем число входов превосходило число выходов. Однако в терминах, важных с точки зрения криптографических приложений, будем полагать, что число входов и число выходов совпадает, причем каждый такой блок является блоком управляемых перестановок.

Итак, пусть $n = rs$, где r и s — целые положительные числа.

Определение 3.19

Переключаемая сеть, в которой первый уровень состоит из s блоков размера $r \times r$, второй — из r блоков размера $s \times s$, третий уровень повторяет первый, а коммутация между соседними уровнями реализована по принципу "каждый блок с каждым имеет ровно одну коммутацию", называется *трехуровневой сетью Клоса с параметрами* (r, s) [39]. Соответственно, *двухуровневая сеть Клоса* — это сеть, содержащая только первые два уровня.

Для трехуровневой сети Клоса с параметрами (r, s) введем обозначение $\mathbf{C}_{<r,\, s,\, r>/m}$, где $rs = n$, а для двухуровневой сети Клоса с параметрами (r, s) — обозначение $\mathbf{C}_{<r,\, s>/m}$.

Пример трехуровневой сети Клоса представлен на рис. 3.10.

Рис. 3.10. Трехуровневая сеть Клоса

Отметим, что большинство рассматриваемых в книге блоков управляемых перестановок являются сетями Клоса. Достаточно очевидно, что свойства сети Клоса полностью определяются свойствами блоков размера $r \times r$ и $s \times s$.

Так, например,

- если каждый из блоков размера $r \times r$ и $s \times s$ является блоком равновероятного смещения, то уже двухуровневая сеть Клоса является блоком равновероятного смещения;
- если каждый из блоков размера $r \times r$ и $s \times s$ является блоком первого порядка, тогда двухуровневая сеть Клоса является блоком первого порядка;
- если каждый из блоков размера $r \times r$ и $s \times s$ является блоком максимального порядка, тогда трехуровневая сеть Клоса является блоком максимального порядка.

Данные утверждения приведены без доказательств, поскольку имеется достаточно публикаций, посвященных этому вопросу. Для $n = 2^k$ и $k \geq 2$ последовательность двухуровневых сетей Клоса задается формулой:

$$\mathbf{C}_{<2^{k-i},2^i>/m}, \quad 1 \leq i \leq k-1. \tag{3.23}$$

Рекурсивная модель блоков равновероятного смещения и блоков первого порядка с послойной структурой

На рис. 3.11 представлены два варианта рекурсивных моделей синтеза блоков $\mathbf{P}_{2n/2m+n}$ и $\mathbf{P}'_{2n/2m+n}$ из двух блоков $\mathbf{P}_{n/m}$ и, соответственно, двух блоков $\mathbf{P}'_{n/m}$, и n блоков $\mathbf{P}_{2/1}$, для которых будем использовать специальное обозначение $\mathbf{R}_{2n/2m+n}$ и $\mathbf{R}'_{2n/2m+n}$.

Рис. 3.11. Построение блоков $\mathbf{R}_{2n/2m+n}$ (*а*) и $\mathbf{R}'_{2n/2m+n}$ (*б*)

Свойства блоков $\mathbf{R}_{2n/2m+n}$ и $\mathbf{R}'_{2n/2m+n}$ зависят от свойств блоков $\mathbf{P}_{n/m}$ и $\mathbf{P}'_{n/m}$. Так, например, если блоки $\mathbf{P}_{n/m}$ и $\mathbf{P}'_{n/m}$ являются блоками равновероятного смещения, то и блоки $\mathbf{R}_{2n/2m+n}$ и $\mathbf{R}'_{2n/2m+n}$ также являются блоками равновероятного смещения.

Если ограничиться условием, что блоки $\mathbf{P}_{n/m}$ и $\mathbf{P}'_{n/m}$ являются блоками первого порядка, тогда блоки $\mathbf{R}_{2n/2m+n}$ и $\mathbf{R}'_{2n/2m+n}$ также являются блоками первого порядка.

Доказательство данных утверждений достаточно очевидно, поскольку:

- первая из рассматриваемых конструкций является двухуровневой сетью Клоса с параметрами (2, n);
- вторая — двухуровневой сетью Клоса с параметрами (n, 2).

И остается воспользоваться свойствами двухуровневой сети Клоса, для которой данные утверждения справедливы. Действительно, поскольку каждый блок $\mathbf{P}_{2/1}$ является блоком равновероятного смещения, а блоки $\mathbf{P}_{n/m}$ и $\mathbf{P}'_{n/m}$ являются блоками равновероятного смещения (или блоками первого порядка), следовательно, блоки $\mathbf{R}_{2n/2m+n}$ и $\mathbf{R}'_{2n/2m+n}$ также являются блоками равновероятного смещения (или блоками первого порядка).

Очевидно, что если в качестве блока $\mathbf{P}'_{n/m}$ использовать блок $\mathbf{P}^{-1}_{n/m}$, то модель, представленная на рис. 3.11, *б*, является блоком $\mathbf{R}^{-1}_{2n/2m+n}$ для модели, приведенной на рис. 3.11, *а*. Дальнейшая детализация (уточнение) моделей (рис. 3.11) зависит от структуры блоков $\mathbf{P}_{n/m}$ и $\mathbf{P}'_{n/m}$.

Так, начиная с $n = 2$ и $m = 1$ и используя первый вариант, можно построить следующую последовательность рекурсивных блоков равновероятного смещения: $\mathbf{R}_{4/4}$, $\mathbf{R}_{8/12}$, $\mathbf{R}_{16/32}$, $\mathbf{R}_{32/80}$, $\mathbf{R}_{64/192}$ и т. д.

Специальное обозначение

С учетом важности свойств построенных таким образом блоков для них введем *специальное обозначение* $\boldsymbol{R}_k$, причем $n = 2^k$, разрядность управляющего вектора $m = \frac{n}{2}\log_2 n = k2^{k-1}$, а сами блоки будем называть *рекурсивными блоками равновероятного смещения.*

Отметим, что в рекурсивном блоке равновероятного смещения $\mathbf{R}_k$ число слоев $l = \log_2 n = k$, а каждая модификация уникальна. Приведенная выше последовательность рекурсивных блоков равновероятного смещения имеет следующий вид: $\mathbf{R}_2$, $\mathbf{R}_3$, $\mathbf{R}_4$, $\mathbf{R}_5$, $\mathbf{R}_6$, $\mathbf{R}_7$ и т. д. С другой стороны, используя второй вариант, можно построить последовательность рекурсивных блоков равновероятного смещения: $\mathbf{R}^{-1}_2, \mathbf{R}^{-1}_3, \mathbf{R}^{-1}_4, \mathbf{R}^{-1}_5, \mathbf{R}^{-1}_6, \mathbf{R}^{-1}_7$ и т. д.

Замечание

Хотя блок $\mathbf{P}_{2/1}$ нельзя представить рекурсивной моделью, поскольку он является тем "неделимым кирпичиком", с использованием которого формируются переключаемые сети, он обладает всеми свойствами, присущими блоку $\mathbf{R}_k$. Поэтому в необходимых случаях вместо $\mathbf{P}_{2/1}$ будем использовать $\mathbf{R}_1$.

На рис. 3.12 в качестве примера представлены блоки $\mathbf{R}_3$ и $\mathbf{R}^{-1}_3$.

Рис. 3.12. Структура блоков $\mathbf{R}_3$ (*а*), $\mathbf{R}_3^{-1}$ (*б*)

Сложность реализации $C_\Omega(\mathbf{R}_k) = C_\Omega(\mathbf{R}_k^{-1}) = mC_\Omega(\mathbf{P}_{2/1}) \cong 2n\log_2 n = k2^{k+1}$, а время выполнения $t(\mathbf{R}_k) = t(\mathbf{R}_k^{-1}) = kt$.

Сравнивая полученные оценки с аналогичными оценками блока циклического сдвига с каскадной структурой, правомерен следующий вывод.

Вывод

Скоростные параметры и схемотехническая сложность реализации управляемого блока циклического сдвига с каскадной структурой $\mathbf{T}_n$ и рекурсивного блока равновероятного смещения $\mathbf{R}_k$ совпадают, однако размерность управляющего вектора во втором случае в $n/2$ раз больше, что обеспечивает более качественное воплощение "свойства рассеивания" за счет увеличения числа реализуемых модификаций.

Необходимо отметить, что известны различные схемы реализации блоков равновероятного смещения с послойной структурой, поскольку они активно используются в алгоритмах вычисления быстрых преобразований Фурье, при транспонировании матриц, при вычислении полиномов, а также в методах сортировки. Все они имеют одинаковую сложность реализации, одинаковую временную задержку, а отличаются друг от друга только коммутацией между слоями. В частности, рассмотренная выше рекурсивная модель в [80] получила название *сети с базовой структурой* (baseline network). Отметим также, что приведенная на рис. 3.4 *R*-сеть отличается от рекурсивной только наличием нетождественного коммутатора после последнего активного слоя. Доказательство эквивалентности переключаемых сетей, а именно *Омега-сети Лурье* (Omega network of Lawrie), *сети Пиза* (Pease's indirect binary n-cube network) и *R-сети*, можно найти в работе С. Паркера [108]. В статье [108], помимо полученных результатов, представляет

интерес математическое представление фиксированных коммутаторов между слоями. Отметим также, что сеть Пиза в работе [80] получила название сети "бабочка" именно за сходство схемы коммутации с очертаниями этого насекомого. Однако мы ограничимся рассмотрением только рекурсивной модели, поскольку для доказательства некоторых криптографических свойств блоков управляемых перестановок она имеет наиболее наглядное представление.

Представление блока $\mathbf{R}_k$ двухуровневой сетью Клоса

Теорема 3.2

Блок равновероятного смещения $\mathbf{R}_{k(V)}$, удовлетворяющий условию $k \geq 2$, для любого i: $1 \leq i \leq k-1$ представим двухуровневой сетью Клоса $\mathbf{C}_{<2^{k-i},\, 2^i>/m(W)}$, в которой:

- первый уровень составляют 2^i блоков равновероятного смещения $\mathbf{R}_{k-i}$;
- второй уровень составляют 2^{k-i} блоков равновероятного смещения размера $\mathbf{R}_i$;
- взаимное расположение блоков $\mathbf{P}_{2/1}$ в первом и последнем активных слоях остается неизменным;
- управляющий вектор W является перестановкой управляющего вектора V, причем перестановка отдельных разрядов возможна только в пределах одного и того же активного слоя.

Доказательство

Для доказательства теоремы воспользуемся методом математической индукции.

Для $k = 2$ утверждение очевидно, поскольку условию $1 \leq i \leq k-1$ удовлетворяет только значение $i = 1$, и по построению структура двухуровневой сети Клоса $\mathbf{C}_{<2^1,\, 2^1>/4}$ совпадает со структурой $\mathbf{R}_2$ (см. рис. 3.7). Соответственно выполнены все дополнительные требования, предъявляемые к двухуровневой сети Клоса $\mathbf{C}_{<2^1,\, 2^1>/4(W)}$, в частности, взаимное расположение блоков $\mathbf{P}_{2/1}$ в первом и последнем активных слоях остается неизменным, а управляющий вектор W совпадает с управляющим вектором V.

Легко проверить справедливость теоремы и для $k = 3$.

Действительно, для $i = 2$ структура блока $\mathbf{C}_{<2^1,\, 2^2>/12}$ совпадает со структурой блока $\mathbf{R}_3$ (см. рис. 3.12, *а*) по построению, а для $i = 1$ структура блока $\mathbf{C}_{<2^2,\, 2^1>/12}$ совпадает со структурой блока $\mathbf{R}_3^{-1}$ (см. рис. 3.12, *б*).

Пусть управляющий вектор V блока $\mathbf{R}_3$ имеет вид:

$$V = (v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}, v_{12}).$$

Тогда управляющий вектор W блока $\mathbf{C}_{<2^2,\, 2^1>/12}$ имеет следующий вид:

$$W = (v_1, v_2, v_3, v_4, v_5, v_7, v_6, v_8, v_9, v_{10}, v_{11}, v_{12}).$$

Другими словами, для преобразования блока $\mathbf{R}_3$ к виду $\mathbf{C}_{<2^2,\,2^1>/12}$ во втором активном слое блока $\mathbf{R}_3$ достаточно поменять местами два блока $\mathbf{P}_{2/1}$ с управляющими разрядами v_6 и v_7, сохранив при этом коммутацию данных блоков с блоками $\mathbf{P}_{2/1}$ первого и третьего активных слоев. Неизменность взаимного расположения блоков $\mathbf{P}_{2/1}$ в первом и последнем активных слоях подтверждается совпадением четырех первых и, соответственно, четырех последних разрядов в векторах V и W.

Пусть теорема справедлива для любого k и любого i: $1 \le i \le k-1$.

Докажем утверждение теоремы для $k+1$ и любого j: $1 \le j \le (k+1)-1$.

Действительно, для $k+1$ и $j = k$ теорема верна, так как $\mathbf{R}_{k+1} = \mathbf{C}_{<2,2^k>/2m+n}$ по построению (рис. 3.13).

Рис. 3.13. Структура блоков $\mathbf{R}_{k+1}$

Докажем для значения j, удовлетворяющего условию: $1 \le j \le k-1$.

В блоке $\mathbf{R}_{k+1}$, по предположению индукции, каждый из двух блоков $\mathbf{R}_k$ для любого i ($1 \le i \le k-1$) представим двухуровневой сетью Клоса $\mathbf{C}_{<r,\,s>/m}$, где ($r = 2^{k-i}$, $s = 2^i$) с сохранением неизменности взаимного расположения блоков $\mathbf{P}_{2/1}$ в первом и последнем активных слоях. В частности, такое представление имеет место для любого $i = j$.

Пусть для заданного i такое представление приведено на рис. 3.14, в котором первый уровень состоит из n блоков $\mathbf{P}_{2/1}$, второй уровень — из $2s$ блоков $\mathbf{R}_{k-i}$, а третий уровень составляют $2r$ блоков $\mathbf{R}_i$.

Отметим, что коммутация между первым и вторым уровнями осталась неизменной, поскольку неизменным осталось взаимное расположение блоков $\mathbf{P}_{2/1}$ в первом активном слое каждой двухуровневой сети Клоса $\mathbf{C}_{<r,\,s>/m}$.

Осталось показать, что первые два уровня образуют s блоков равновероятного смещения $\mathbf{R}_{k+1-i}$. Для этого расположенные рядом блоки $\mathbf{P}_{2/1}$ первого уровня

сгруппируем в непересекающиеся классы по r в каждом. При такой группировке всего будет ровно $s = 2^i$ классов ($2^i = n / 2^{k-i} = 2^k / 2^{k-i}$).

Рис. 3.14. Модифицированная структура блоков $\mathbf{R}_{k+1}$

Рассмотрим, например, первые r блоков $\mathbf{P}_{2/1}$ при их нумерации слева направо. Каждый из них имеет коммутацию с первым и $s + 1$-м блоками равновероятного смещения второго уровня $\mathbf{R}_{k-i}$ (на рис. 3.14 рассматриваемые блоки имеют самую темную окраску). Рассматриваемые r блоков $\mathbf{P}_{2/1}$ первого уровня и два блока $\mathbf{R}_{k-i}$ второго уровня образуют блок $\mathbf{R}_{k+1-i}$.

Число таких блоков $\mathbf{R}_{k+1-i}$ равно количеству классов, то есть $s = 2^i$.

Наличие коммутации между вновь сформированными блоками $\mathbf{R}_{k+1-i}$ и блоками $\mathbf{R}_i$ по принципу "каждый с каждым" очевидно.

Действительно, с учетом общей размерности блока $\mathbf{R}_{k+1}$ и размерности блоков $\mathbf{R}_{k+1-i}$ и $\mathbf{R}_i$, отсутствие коммутации между какими-либо двумя блоками $\mathbf{R}_{k+1-i}$ и $\mathbf{R}_i$ противоречит условию равновероятного смещения для $\mathbf{R}_{k+1}$. После перегруппировки блоков второго уровня $\mathbf{R}_{k-i}$, как показано на рис. 3.15, блок $\mathbf{R}_{k+1}$ принимает окончательный вид двухуровневой сети Клоса $\mathbf{C}_{<2^{k+1-i},2^i>/2m+n}$, состоящей из блоков $\mathbf{R}_{k+1-i}$ и $\mathbf{R}_i$.

В заключение отметим, что при перегруппировке блоков второго уровня $\mathbf{R}_{k-i}$ взаимное расположение блоков $\mathbf{P}_{2/1}$ в каждом активном слое второго уровня изменилось, но только в пределах своего активного слоя, и не повлияло на взаимное расположение блоков $\mathbf{P}_{2/1}$ первого уровня.

Теорема доказана полностью.

Рис. 3.15. Двухуровневая сеть Клоса $\mathbf{C}_{<2^{k+1-i},2^i>/2m+n}$

Эквивалентность моделей $\mathbf{R}_k$ и $\mathbf{R}_k^{-1}$

Рассмотрение данного вопроса обусловлено необходимостью доказательства утверждения, что для рекурсивной модели $\mathbf{R}_k$ не только конструкция $\mathbf{R}_{k(V)} \circ \mathbf{R}_{k(U)}^{-1}$, но и конструкция $\mathbf{R}_{k(U)}^{-1} \circ \mathbf{R}_{k(V)}$ является блоком максимального порядка.

Доказательство утверждения для первой конструкции, получившей название *модернизированной схемы Бенеша*, рассматривается в *разд. 3.3.2*.

Вторая конструкция является следствием синтеза итеративных блочных шифров на основе базовой схемы патента РФ № 2140714 [1]. Для этой конструкции такой вопрос правомерен, поскольку конструкция $\mathbf{R}_{k(U)}^{-1} \circ \mathbf{R}_{k(V)}$ формально не является схемой Бенеша. Однако утвердительный ответ на поставленный вопрос можно получить, если, например, доказать эквивалентность блоков $\mathbf{R}_{k(U)}^{-1}$ и $\mathbf{R}_{k(V)}$. Разумеется, что при этом между векторами U и V должно существовать взаимно однозначное соответствие. И на самом деле такое соответствие существует, причем вектор U является результатом простой перестановки разрядов вектора V.

Теорема 3.3

Модели блоков $\mathbf{R}_k$ и $\mathbf{R}_k^{-1}$ эквивалентны с точностью до перестановки разрядов в управляющем векторе, то есть $\forall k \geq 1\ \exists \pi$: $U = \pi(V)$, $\pi \in S_n$ и $\mathbf{R}_{k(U)}^{-1} = \mathbf{R}_{k(V)}$.

Доказательство

Для доказательства теоремы воспользуемся методом математической индукции. Для $k = 1, 2, 3$ утверждение достаточно очевидно. Например, для блока $\mathbf{R}^{-1}_{3(U)}$ (см. рис. 3.12, *б*) с учетом особенностей нумерации разрядов в блоке $\mathbf{R}^{-1}_{3(U)}$ управляющий вектор U имеет вид:

$$U = (v_9, v_{10}, v_{11}, v_{12}, v_5, v_7, v_6, v_8, v_1, v_2, v_3, v_4).$$

При этом если блок $\mathbf{R}^{-1}_{3(U)}$ рассматривать как некоторый блок $\mathbf{R}'_{8/12(W)}$ (см. рис. 3.11, *б*), то тогда управляющий вектор W имеет вид:

$$W = (v_1, v_2, v_3, v_4, v_5, v_7, v_6, v_8, v_9, v_{10}, v_{11}, v_{12}),$$

т. е. отличается от вектора V только транспозицией шестого и седьмого разрядов.

Предполагая, что утверждение теоремы 3.3 справедливо для любого k, докажем теорему для $k + 1$.

Воспользуемся утверждением теоремы 3.2 для блока $\mathbf{R}_{k+1(V)}$.

С учетом теоремы 3.2 блок $\mathbf{R}_{k+1(V)}$ (рис. 3.16, *а*) можно представить двухуровневой сетью Клоса $\mathbf{C}_{<2^k,2>/2m+n(W)}$ (рис. 3.16, *б*), состоящей из блоков $\mathbf{R}_k$ и $\mathbf{R}_1 = \mathbf{P}_{2/1}$, причем взаимное расположение блоков $\mathbf{P}_{2/1}$ первого и последнего активных слоев остается неизменным. Управляющий вектор W блока $\mathbf{C}_{<2^k,2>/2m+n(W)}$ является перестановкой разрядов вектора V, причем перестановка отдельных разрядов возможна только в пределах одного и того же активного слоя. В частности, в управляющем векторе W значения разрядов, относящихся к первому и последнему активным слоям, совпадают с соответствующими разрядами управляющего вектора V.

Рис. 3.16. Структура блоков $\mathbf{R}_{k+1}$ (*а*) и $\mathbf{C}_{<2^k,2>/2m+n}$ (*б*)

Но по предположению индукции каждый из двух блоков $\mathbf{R}_k$ можно заменить блоком $\mathbf{R}_k^{-1}$ с соответствующей перестановкой разрядов в управляющих векторах блоков $\mathbf{R}_k$. Перестроенный таким образом блок $\mathbf{C}_{<2^k,2>/2m+n(W)}$, очевидно, является блоком $\mathbf{R}_{k+1}^{-1}$, что и доказывает утверждение теоремы.

Следствие 3.1

В представлении блока $\mathbf{R}_k$ двухуровневой сетью Клоса $\mathbf{C}_{<2^{k-i},2^i>/2m+n(W)}$, состоящей из блоков $\mathbf{R}_{k-i}$ и $\mathbf{R}_i$ ($1 \le i \le k-1$), любой из блоков $\mathbf{R}_{k-i}$ (соответственно, $\mathbf{R}_i$) можно заменить блоком $\mathbf{R}_{k-i}^{-1}$ (соответственно, $\mathbf{R}_i^{-1}$).

С учетом следствия 3.1, например, в качестве блока управляемых перестановок $\mathbf{P}_{64/192}$ можно использовать двухуровневую сеть Клоса $\mathbf{C}_{<8,8>/192}$, причем если в первом уровне использовать блоки $\mathbf{R}_3$, а во втором — блоки $\mathbf{R}_3^{-1}$, то такая структура обеспечивает симметрию относительно входа/выхода. То есть структура блока управляемых перестановок $\mathbf{P}_{64/192}$ и блока $\mathbf{P}_{64/192}^{-1}$ совпадает. Блок $\mathbf{P}_{64/192}$ с предложенной структурой используется в блочном шифре SPECTR-128, описание которого приведено в *гл. 5*.

3.3.2. Схемы реализации блоков максимального порядка

Универсальная схема

Базовой идеей данной схемы (рис. 3.17) [25] является представление перестановки в виде произведения транспозиций. Схема реализуема для любого значения n.

Блок α_j ($1 \le j \le n-1$) состоит из j блоков $\mathbf{P}_{2/1}$, в которых *только для данной схемы коммутация "накрест" выполняется при нулевом значении управляющего разряда*, и двоичного дешифратора $\mathbf{D}_k$, где k — ближайшее целое, большее либо равное $\log_2(j+1)$ (например, *см. разд. 3.2.1*). На вход дешифратора $\mathbf{D}_k$ поступает k-разрядный управляющий вектор $V^{(j)}$, где $V = V^{(n-1)} \| V^{(n-2)} \| \ldots \| V^{(2)} \| V^{(1)}$, а на выходе формируется двоичный вектор $U^{(j)}$ длины 2^k с весом Хэмминга равным единице, т. е. значение 1 формируется только в одном разряде, а именно:

$$u_s^{(j)} = \begin{cases} 1, & \text{если } s = 1 + v_1^{(j)} + v_2^{(j)} 2 + \ldots + v_k^{(j)} 2^{k-1} \\ 0, & \text{в остальных случаях} \end{cases}.$$

Блок α_j реализует тождественное преобразование, если для каждого блока $\mathbf{P}_{2/1}$ значение управляющего разряда $u_s^{(j)} = 0$, и транспозицию с участием $n-j$-го входного разряда вектора X. Например, блок α_{n-1} реализует тождественное преобразование, если $U^{(n-1)} = E_0 = (0, 0, \ldots, 0)$, и транспозицию с участием 1-го и s-го разрядов, если $u_s^{(j)} = 1$.

Рис. 3.17. Универсальная схема реализации блока максимального порядка

То есть блок α_{n-1} реализует коммутацию заданного вектором $V^{(n-1)}$ s-го разряда x_s входного вектора X с первым разрядом выхода y_1, а первый разряд x_1 входного вектора X с s-м разрядом входа блока α_{n-2}. Соответственно, блок α_{n-2} реализует коммутацию заданного вектором $V^{(n-2)}$ t-го разряда x_t входного вектора блока α_{n-2} со вторым разрядом выхода y_2, а второй разряд x_2 входного вектора блока α_{n-2} с t-м разрядом входа блока α_{n-3}, и т. д.

Последний блок α_1 выполняет коммутацию x_{n-1} в y_n (соответственно, x_n в y_{n-1}), если $V^{(1)} = v_m = 1$, и коммутацию x_{n-1} в y_{n-1} (соответственно, x_n в y_n), если $V^{(1)} = v_m = 0$.

Главным достоинством универсальной схемы является реализуемость для любого значения n, однако схема несимметричная и несинхронная. Схема представляет интерес в теоретическом плане, но для использования в практических приложениях, связанных с защитой информации, малопригодна по параметрам скорости и схемотехнической сложности. Сложность реализации такой схемы

$$C_\Omega(\mathbf{P}_{n/m}^{(n)}) = \frac{n(n-1)}{2} C_\Omega(\mathbf{P}_{2/1}) \approx 2n(n-1),$$

а время задержки — $t(\mathbf{P}_{n/m}^{(n)}) = (2n-3)\tau$.

Дополнительная сложность реализации обусловлена наличием двоичных дешифраторов, что усложняет реализацию равновероятного выбора модификаций. Для обеспечения уникальности каждой модификации необходимо в управляющем векторе $V = V^{(n-1)} \parallel \ldots \parallel V^{(2)} \parallel V^{(1)}$ обеспечить ровно $j + 1$ значение вектора $V^{(j)}$ для каждого j: $1 \le j \le n - 1$.

Схема Бенеша

Схема Бенеша [39] предназначена для синтеза блоков максимального порядка для $n = 2^k$, имеющих специальную структуру. Введем для них специальное обозначение $\mathbf{B}_k$. Начиная с элементарного блока максимального порядка $\mathbf{B}_1 = \mathbf{P}_{2/1}^2 = \mathbf{P}_{2/1}$, переход от блока $\mathbf{B}_k$ к блоку $\mathbf{B}_{k+1}$ реализуется в соответствии с рис. 3.18, *а*.

Схема Бенеша является частным случаем трехуровневой сети Клоса с параметрами (2, n, 2).

В блоке $\mathbf{B}_k$ число активных слоев l равно $2\log_2 n - 1 = 2k - 1$, а разрядность управляющего вектора m равна $n\log_2 n - \frac{n}{2} = k2^k - 2^{k-1} = 2^{k-1}(2k-1)$.

Очевидно, что структура блоков $\mathbf{B}_k^{-1}$ и $\mathbf{B}_k$ совпадает. То есть $\mathbf{B}_{k(V)}^{-1} = \mathbf{B}_{k(U)}$, где $V = V_1 \parallel V_2 \parallel \ldots \parallel V_{2k-1}$ и $U = V_{2k-1} \parallel V_{2k-2} \parallel \ldots \parallel V_1$. Блоками максимального порядка являются: $\mathbf{B}_1 = \mathbf{P}_{2/1}^{(2)}$, $\mathbf{B}_2 = \mathbf{P}_{4/6}^{(4)}$, $\mathbf{B}_3 = \mathbf{P}_{8/20}^{(8)}$, $\mathbf{B}_4 = \mathbf{P}_{16/56}^{(16)}$, $\mathbf{B}_5 = \mathbf{P}_{32/144}^{(32)}$ и т. д.

Использование данной схемы в криптографических приложениях ограничено скоростными параметрами: $t(\mathbf{B}_k) = \tau(2k-1)$, а также необходимостью формировать m-разрядный управляющий вектор, для которого важно обеспечить независимость и равновероятность всех 2^m значений управляющего вектора.

На рис. 3.18, *б* приведена схема симметричного начального блока $\mathbf{P}_{2/2}^{(2)}$, позволяющего построить модернизированную схему Бенеша, рассматриваемую далее,

которая представляет наибольший интерес с точки зрения криптографических приложений.

Рис. 3.18. Рекурсивная схема Бенеша $\mathbf{B}_{k+1}$ (*а*) и схема симметричного начального блока $\mathbf{P}^{(2)}_{2/2}$ (*б*)

Модернизированный вариант схемы Бенеша

Теорема 3.4

Если в схеме блока $\mathbf{P}^{(2m)}_{2n/2m+n} = \mathbf{B}_{k+1}$ (рис. 3.18, *а*) средний слой блоков $\mathbf{P}_{2/1}$ заменить блоками $\mathbf{P}^{(2)}_{2/2}$ (рис. 3.18, *б*), тогда блок $\mathbf{P}_{2n/m'}$ удовлетворяет следующим требованиям:

- является блоком максимального порядка;
- может быть представлен в виде суперпозиции рекурсивно сформированных взаимно обратных блоков равновероятного смещения первого порядка $\mathbf{R}_{k+1(V)} \circ \mathbf{R}^{-1}_{k+1(U)}$;
- является блоком равновероятного смещения.

Утверждение теоремы очевидно, поскольку блок $\mathbf{P}^{(2)}_{2/2}$ (рис. 3.18, *б*) эквивалентен блоку $\mathbf{P}_{2/1}$ с одноразрядным управляющим вектором $V' = v_1 \oplus v_2$, а именно:

$$\mathbf{P}^{(2)}_{2/2(v_1 \| v_2)} \equiv \mathbf{P}_{2/1(v_1)} \circ \mathbf{P}^{-1}_{2/1(v_2)} \equiv \mathbf{P}^{(2)}_{2/1(v_1 \oplus v_2)}.$$

Блок $\mathbf{P}^{(2n)}_{2n/m'}$ состоит из $2(\log_2 n + 1) = 2(k + 1)$ слоев $\mathbf{L}_{2n}$, причем значение $m' = 2n(\log_2 n + 1) = 2^{k+1}(k + 1)$.

Действительно, первые $k+1$ слоев $\mathbf{L}_{2n}$ блока $\mathbf{P}_{2n/m'}^{(2n)}$ реализуют рекурсивную модель блока равновероятного смещения $\mathbf{R}_{k+1(V)}$ (см. рис. 3.11, *а*), а остальные $k+1$ слоев — рекурсивную модель блока равновероятного смещения $\mathbf{R}_{k+1(U)}^{-1}$ (см. рис. 3.11, *б*).

Для данной конструкции блоками максимального порядка являются: $\mathbf{P}_{4/8}^{(4)}$, $\mathbf{P}_{8/24}^{(8)}$, $\mathbf{P}_{16/64}^{(16)}$, $\mathbf{P}_{32/160}^{(32)}$ и т. д. Как и для основной модели Бенеша, структура блока $(\mathbf{P}_{n/m}^{(n)})^{-1}$ совпадает со структурой блока $\mathbf{P}_{n/m}^{(n)}$, т. е. $(\mathbf{P}_{n/m(V)}^{(n)})^{-1} = \mathbf{P}_{n/m(U)}^{(n)}$, где $V = V_1 \parallel V_2 \parallel \ldots \parallel V_{2k}$ и $U = V_{2k} \parallel V_{2k-1} \parallel \ldots \parallel V_1$.

Практическое использование модернизированной схемы Бенеша, а именно суперпозиции блоков $(\mathbf{P}_{n/m(V)}^{(n)})^{-1}$ и $\mathbf{P}_{n/m(U)}^{(n)}$, рассматривается в *гл. 5*.

Схема Ваксмана

На основе схемы Бенеша Ваксман (Waksman) в [133] предложил схему реализации блоков максимального порядка с минимальной длиной управляющего кода (рис. 3.19). Как и в схеме Бенеша, используется рекурсивный метод построения блоков $\mathbf{P}_{2n/m'}^{(2n)}$, при котором, начиная с $n = 4$, в последнем слое $\mathbf{L}_{2n}$ каждого блока $\mathbf{P}_{2n/m'}^{(2n)}$ отсутствует один блок $\mathbf{P}_{2/1}$.

Рис. 3.19. Рекурсивная схема Ваксмана

Например, для $n = 8$ в четвертом активном слое отсутствуют два блока $\mathbf{P}_{2/1}$, поскольку второй, третий и четвертый слои представляют два параллельно расположенных блока $\mathbf{P}_{4/5}^{(4)}$, а в пятом активном слое — один.

При заданном значении n ($n = 2^k$) блок $\mathbf{P}_{n/m}^{(n)}$ состоит из $2\log_2 n - 1$ слоев, а значение $m = n\log_2 n - n + 1$. Соответственно для блока $\mathbf{P}_{2n/m'}^{(2n)}$ значение $m' = 2n\log_2 n + 1$. Блоками максимального порядка являются: $\mathbf{P}_{2/1}^{(2)}$, $\mathbf{P}_{4/5}^{(4)}$, $\mathbf{P}_{8/17}^{(8)}$, $\mathbf{P}_{16/49}^{(16)}$, $\mathbf{P}_{32/129}^{(32)}$ и т. д.

В общем случае схема Ваксмана является обобщением трехуровневой сети Клоса с произвольными параметрами (r, s, r), в которой отсутствует один блок размера $r \times r$ в третьем уровне, и каждый блок меньшего размера также является схемой Ваксмана.

Для использования в криптографических приложениях схема Ваксмана малопригодна ввиду ее асимметричности.

Возвращаясь к схеме, представленной на рис. 3.18, *а*, необходимо отметить, что свойства блока $\mathbf{P}_{2n/m'}$ определяются свойствами блоков $\mathbf{P}_{n/m}$:

- если блок $\mathbf{P}_{n/m}$ является блоком равновероятного смещения, то и блок $\mathbf{P}_{2n/m'}$ является блоком равновероятного смещения;
- если блок $\mathbf{P}_{n/m}$ является блоком h-го порядка ($h < \frac{n}{2}$), то блок $\mathbf{P}_{2n/m'}$ является, по крайней мере, блоком $2h$-го порядка;
- блок $\mathbf{P}_{2n/m'}$ является трехуровневой сетью Клоса с параметрами (2, n, 2).

Использование рекурсивного метода для построения блоков $\mathbf{P}_{2n/m'}$, по крайней мере, $2h$-го порядка $\mathbf{P}_{2n/m'}$, на основе двух блоков $\mathbf{P}_{n/m}^{(h)}$, рассматривается в *разд. 3.3.3*.

3.3.3. Рекурсивная схема реализации блоков порядка *h*

Рассмотрим рекурсивную модель Бенеша (рис. 3.18, *а*), в которой вместо блоков максимального порядка $\mathbf{P}_{n/m}^{(n)}$ используются блоки h-го порядка $\mathbf{P}_{n/m}^{(h)}$ ($1 \le h \le n$). На рис. 3.20 схематично представлены блоки $\mathbf{P}_{2n/2m+2n}^{(2h)}$ и $\mathbf{P}_{4n/4m+8n}^{(4h)}$, основу которых составляют блоки $\mathbf{P}_{n/m}^{(h)}$. Коммутация между слоями также указана схематично. Так, например, две наклонные линии, примыкающие "сверху" к активному слою $\mathbf{L}_{2n}$, указывают коммутацию "левых" и "правых" разрядов *входа* блоков $\mathbf{P}_{2/1}$ данного активного слоя, а примыкающие "снизу" — коммутацию разрядов *выхода* блоков $\mathbf{P}_{2/1}$ активного слоя $\mathbf{L}_{2n}$.

Пусть $h = 1$. Очевидно, что в схеме на рис. 3.20, *а* первый слой $\mathbf{L}_{2n}$ и два блока $\mathbf{P}_{n/m}^{(n)}$ образуют рекурсивную модель блока первого порядка $\mathbf{R}_{2n/2m+n}$. Но тогда для любого входного разряда блока $\mathbf{P}_{2n/2m+2n}^{(2h)}$ и любого выходного разряда суще-

ствует, по крайней мере, два варианта коммутации. Действительно, каждый блок $\mathbf{P}_{2/1}$ первого слоя $\mathbf{L}_{2n}$ имеет ровно одну коммутацию с "левым" и "правым" блоками $\mathbf{P}_{n/m}^{(h)}$. Аналогично, каждый блок $\mathbf{P}_{2/1}$ последнего слоя $\mathbf{L}_{2n}$ также имеет ровно одну коммутацию с "левым" и "правым" блоками $\mathbf{P}_{n/m}^{(h)}$. Так как блоки $\mathbf{P}_{n/m}^{(h)}$ являются блоками первого порядка, следовательно, между каждым разрядом входа и каждым разрядом выхода в блоке $\mathbf{P}_{n/m}^{(h)}$ существует, по крайней мере, одна коммутация. Но тогда коммутация между любым входным и любым выходным разрядом блока $\mathbf{P}_{2n/2m+2n}^{(2h)}$ возможна как через "левый" блок $\mathbf{P}_{n/m}^{(h)}$, так и через "правый" блок $\mathbf{P}_{n/m}^{(h)}$.

В справедливости утверждения, что блок $\mathbf{P}_{2n/2m+2n}^{(2h)}$ действительно является блоком второго порядка, легко убедиться, рассмотрев четыре варианта взаимного расположения двух входных и двух выходных разрядов, а именно: принадлежат входные (соответственно, выходные) разряды одному блоку $\mathbf{P}_{2/1}$ или нет. Если, например, входные разряды принадлежат одному и тому же блоку $\mathbf{P}_{2/1}$ первого слоя $\mathbf{L}_{2n}$, тогда, выбирая "левую" коммутацию для одного из входных разрядов, для второго входного разряда "правая" коммутация устанавливается автоматически, поскольку блок $\mathbf{P}_{2/1}$ управляется одним разрядом. Если входные разряды принадлежат разным блокам $\mathbf{P}_{2/1}$ первого слоя $\mathbf{L}_{2n}$, тогда, выбирая "левую" коммутацию для одного из входных разрядов, "правая" коммутация для второго входного разряда выбирается независимо от первого входного разряда.

Аналогично рассматриваются два варианта и для выходных разрядов.

Доказательство для $h \geq 2$, с использованием метода математической индукции, принципиально не отличается от случая $h = 2$, однако требуется точность в формулировке предположения справедливости индуктивного утверждения.

Рис. 3.20. Структура блоков $\mathbf{P}_{2n/2m+2n}^{(2h)}$ (*а*) и $\mathbf{P}_{4n/4m+8n}^{(4h)}$ (*б*)

Рис. 3.21. Структура блоков $\mathbf{P}^{(2)}_{32/96}$ (*а*), $\mathbf{P}^{(4)}_{32/112}$ (*б*), $\mathbf{P}^{(8)}_{32/128}$ (*в*)

На рис. 3.21 приведены примеры блоков различного порядка для $n = 32$.

Замечание

Необходимо отметить, что в рекурсивной модели блока $\mathbf{P}^{(2h)}_{2n/2m+2n}$ (см. рис. 3.20, *а*) коммутация, например, "левых" разрядов выхода блоков $\mathbf{P}_{2/1}$ первого активного слоя $\mathbf{L}_{2n}$ с левым блоком $\mathbf{P}_{n/m}$ строго соответствует нумерации блоков $\mathbf{P}_{2/1}$ первого активного слоя $\mathbf{L}_{2n}$. Однако соблюдение такой нумерации вовсе необязательно, что позволяет формировать различные модификации модели блока $\mathbf{P}^{(2h)}_{2n/2m+2n}$, а данную модель рассматривать как базовую.

Существуют и другие варианты синтеза блоков заданного порядка.

Ввиду того, что сложность реализации блока $\mathbf{P}_{n/m}$ существенно зависит от значения параметра h, рекомендуется использовать управляемые блоки $\mathbf{P}_{n/m}$ с мини-

мально необходимым значением параметра h, удовлетворяющего требованиям решаемой задачи.

Например, в статье [18] рассматривалась и была решена задача построения блока *четвертого* порядка $\mathbf{P}_{n/m}^{(4)}$ (рис. 3.22), удовлетворяющего следующим требованиям:

- выполняется условие равновероятного смещения;
- время выполнения операции перестановки блоком $\mathbf{P}_{n/m}^{(4)}$ не зависит от длины n и является постоянной величиной;
- обеспечена уникальность реализуемых модификаций для каждого значения управляющего вектора;
- соотношение параметров $m : n$ не превышает значение 2, т. е. разрядность m управляющего вектора не превышает длин преобразуемого блока данных более чем в два раза.

Рис. 3.22. Блок управляемых перестановок $\mathbf{P}_{n/m}^{(4)}$

На рис. 3.22 параметры n, m и s имеют следующие значения:

$$n = 2^k, \qquad m = n + 7(k-2) + 2, \qquad s = 2^{k-2},$$

а обозначения $\mathbf{X}_{ij}$ соответствуют фиксированным коммутаторам и предназначены для обеспечения уникальности всех модификаций. Говоря о постоянной величине времени выполнения операции блоком $\mathbf{P}_{n/m}^{(4)}$, необходимо отметить, что в статье [18] не учитывалось время формирования управления дешифраторами в блоках циклического сдвига $\mathbf{T}_{4s/k-1}$, $\mathbf{T}_{2s/k-21}$, $\mathbf{T}_{s/k-2}$ (см. слои α, β и δ на рис. 3.22).

Следует сказать, что если невозможно обеспечить формирование управляющего вектора заблаговременно, необходимо вместо блока $\mathbf{P}_{n/m}^{(4)}$ использовать блок $(\mathbf{P}_{n/m}^{(4)})^{-1}$, поскольку в последнем случае первоначально выполняется слой ε, в котором дешифраторы не используются, а дополнительная задержка, обусловленная работой дешифраторов, не превысит значение $k-4$. Схема практически реализуема для $k = 4 \div 8$ ($n = 2^k$), причем для значений $k \geq 4$ существуют варианты, обеспечивающие уникальность реализуемых перестановок для каждого значения управляющего вектора.

3.3.4. Реализация полноцикловых перестановок

В некоторых криптографических приложениях возникает задача построения управляемых блоков полноцикловых перестановок с повышенным быстродействием. Удачным решением представляется следующий вариант формирования полноцикловых перестановок длины n из специальной фиксированной перестановки и перестановок, состоящих из g циклов длины h, где $n = gh$ [17].

Пусть π_1 — произвольная перестановка. Тогда ее можно представить в виде произведения циклов:

$$\pi_1 = (z_{11}, z_{12}, \ldots, z_{1h_1})\,(z_{21}, z_{22}, \ldots, z_{2h_2})\,\ldots\,(z_{g1}, z_{g2}, \ldots, z_{gh_g}), \tag{3.24}$$

где $1 \leq g \leq n$, $1 \leq h_j \leq n$, $h_1 + h_2 + \ldots + h_g = n$, причем все z_{ij} различны, за исключением $z_{i1} = z_{ih_i}$ для $h_i = 1$.

Лемма 3.1

Произведение перестановок $\pi = \pi_1 \circ \pi_2$, где $\pi_2 = (z_{11}, z_{21}, ..., z_{g1})$, является полноцикловой перестановкой.

Доказательство получается простым перемножением, причем перестановка π имеет вид:

$$\pi = (z_{11}, z_{12}, \ldots, z_{1h_1}, z_{21}, z_{22}, \ldots, z_{2h_2}, \ldots, z_{g1}, z_{g2}, \ldots, z_{gh_g}). \tag{3.25}$$

Теорема 3.5

Суперпозиция перестановок π_1 и π_2 является полноцикловой перестановкой, если перестановка π_2 состоит только из одного цикла длины больше 1, причем в данном

цикле присутствует ровно по одному элементу из каждого цикла перестановки π_1 (включая циклы длины 1). Порядок следования элементов в перестановке π_2 не меняет полноцикловую структуру перестановки π.

Доказательство

Утверждение теоремы достаточно очевидно, если учесть, что в перестановке π_1, во-первых, циклы можно выписывать, начиная с любого элемента (например, с представителя данного цикла в π_2), а во-вторых, циклы можно менять местами (например, выписать их в соответствии с порядком следования представителей в π_2). Для такого представления перестановки π_1 выполнены условия леммы, что и доказывает справедливость теоремы.

Но тогда, согласно данной теореме, для построения управляемых блоков полноцикловых перестановок длины n достаточно использовать, например, матричные блоки полноцикловых перестановок меньшей длины (например, длины h) и простое переплетение проводников, реализующее фиксированную перестановку. Использование таких блоков полноцикловых перестановок позволяет увеличить скорость выполнения управляемой операции, что имеет приоритетное значение, например, *в управляемых сумматорах*, в которых блоки полноцикловых перестановок применяются как для управления битами переноса, так и в качестве функции усложнения выходного значения сумматора. Более подробно использование блоков полноцикловых перестановок в управляемых сумматорах рассматривается в работе [17].

Второй вариант построения блоков полноцикловых перестановок с использованием универсальной схемы (см. рис. 3.17) предложен в работах [3, 28]. Рассматривается конструкция, реализующая все полноцикловые перестановки, причем каждая модификация уникальна. Однако, как отмечалось выше, универсальная схема и ее разновидности малопригодны для криптографических приложений.

3.3.5. Синтез блоков с заданной цикловой структурой

Представляет интерес вопрос синтеза блоков управляемых перестановок, которые реализуют перестановки с заданной цикловой структурой. В работе [99] предложен способ построения псевдослучайных подстановок с заданной цикловой структурой, основанный на хорошо известной из теории групп теореме, которая утверждает, что для произвольных подстановок σ и δ подстановки δ и $\sigma \circ \delta \circ \sigma^{-1}$ имеют одинаковую цикловую структуру. При этом σ-подстановка формируется блоком управляемых подстановок $\mathbf{S}_{n/m}$ (рис. 3.23, *а*).

В работе [99] не рассматриваются практические варианты реализации блоков $\mathbf{S}_{n/m}$, что не позволяет оценить свойства и эффективность данной конструкции.

Напомним, что в современных блочных шифрах, использующих блоки управляемых подстановок, как правило, размерность блока не превышает значение 8.

Поэтому для увеличения общей размерности одновременно обрабатываемого блока данных используют несколько параллельно расположенных блоков $\mathbf{S}_{n'/m'}$. То есть $\mathbf{S}'_{n/m} = \mathbf{S}_{n'/m'} \parallel \mathbf{S}_{n'/m'} \parallel ... \parallel \mathbf{S}_{n'/m'}$ ($n = n' + n' + ... + n'$, $m = m' + m' + ... + m'$), где $n' \le 8$. В частности, в DES используется 8 разных блоков $\mathbf{S}_{n'/m'} = \mathbf{S}_{4/2}$, которые и образуют блок управляемых подстановок $\mathbf{S}_{32/16}$ ($n = 32$, $m = 16$). Проблема построения фиксированной подстановки δ общего вида для $n = 32$ остается открытой, поскольку такая подстановка $\delta \in S_{2^{32}}$. Однако данная конструкция может иметь практическое применение, если в качестве подстановки δ использовать, например, подстановку с коммутацией разрядов $\mathbf{P}_\pi$.

Рис. 3.23. Управляемые блоки $\mathbf{S}'_{n/m}$ и $\mathbf{I}_{n/m}$ с заданной цикловой структурой

Кроме того, если вместо блока управляемых подстановок $\mathbf{S}_{n/m}$ использовать блок управляемых перестановок $\mathbf{P}_{n/m}$, то получим блок управляемых перестановок $\mathbf{P}'_{n/m} = \mathbf{P}_{n/m} \circ \mathbf{P}_\pi \circ \mathbf{P}^{-1}_{n/m}$ с заданной цикловой структурой.

Такая структура имеет общий характер, а количество потенциально реализуемых модификаций определяется свойствами блока $\mathbf{P}_{n/m}$. В частности, количество модификаций блока $\mathbf{P}'_{n/m}$ может быть меньше значения 2^m даже в том случае, когда в блоке $\mathbf{P}_{n/m}$ все 2^m модификаций уникальны. Данный способ синтеза построения специальных классов управляемых перестановок с заданной цикловой структурой является универсальным.

В частности, он легко позволяет построить блоки управляемых перестановок, которые реализуют только полноцикловые перестановки, причем получаемая схема является более экономичной и быстродействующей по сравнению с универсальной схемой построения блоков управляемых полноцикловых перестановок [3].

Инволюции

Определение 3.20

Инволюцией называется преобразование $\mathbf{I}$, для которого $(\mathbf{I})^2$ — тождественное преобразование.

Интерес к данному преобразованию при построении шифров заключается в удобстве их использования при зашифровании и расшифровании, поскольку $\mathbf{I} = \mathbf{I}^{-1}$.

В частности, перестановка π является *инволюцией*, если ее представление в виде произведения непересекающихся циклов не содержит циклов длины больше, чем два.

Определение 3.21

Подстановочное преобразование $\mathbf{I} = \mathbf{P}_\pi$ называется *перестановочной инволюцией*, если инволюцией является перестановка π.

Если в схеме (рис. 3.23, *б*) перестановка π является инволюцией, тогда $\mathbf{I} = \mathbf{P}_\pi$ — перестановочная инволюция, а блок управляемых перестановок $\mathbf{I}_{n/m} = \mathbf{P}_{n/m} \circ \mathbf{I} \circ \mathbf{P}_{n/m}^{-1}$ реализует некоторый класс перестановочных инволюций. Существенным конструктивным недостатком предложенного в [99] варианта построения управляемых инволюций является двукратное увеличение времени задержки по сравнению с временем задержки блока $\mathbf{P}_{n/m}$.

Для того чтобы обойти этот недостаток, можно использовать конструктивную схему (рис. 3.24) синтеза блока управляемых инволюций $\mathbf{I}_{2n/m}$ с $2n$-битовым входом и m-битовым управляющим вектором [15]. Эта схема в некотором смысле также является универсальной, поскольку можно использовать произвольный блок $\mathbf{P}_{n/m}$. Блоки $\mathbf{P}_{n/m(V)}$ и $\mathbf{P}_{n/m(V)}^{-1}$ функционируют параллельно, а не последовательно, что обеспечивает более высокое быстродействие по сравнению со схемой, приведенной на рис. 3.23, *б*. Преобразование двоичного вектора $X = L \,||\, R$ длины $2n$ для схемы, представленной на рис. 3.24, осуществляется следующим образом: n-битовые векторы L и R поступают на вход блоков $\mathbf{P}_{n/m}$ и $\mathbf{P}_{n/m}^{-1}$, соответственно, после чего преобразованные значения меняются местами.

Формула преобразования имеет следующий вид:

$$\mathbf{I}_{2n/m(V)}(L \,||\, R) = \mathbf{P}_{n/m(V)}^{-1}(R) \,||\, \mathbf{P}_{n/m(V)}(L), \tag{3.26}$$

т. к. $\quad Y = L' \,||\, R' = \mathbf{I}_{2n/m(V)}(X) = \mathbf{I}_{2n/m(V)}(L \,||\, R) = \mathbf{P}_{n/m(V)}^{-1}(R) \,||\, \mathbf{P}_{n/m(V)}(L)$.

Очевидно, что $\mathbf{I}_{2n/m}$ действительно инволюция, поскольку

$$X' = \mathbf{I}_{2n/m}(Y) = \mathbf{I}_{2n/m}(L' \,||\, R') = \mathbf{P}_{n/m}^{-1}(R') \,||\, \mathbf{P}_{n/m}(L') =$$

$$= \mathbf{P}_{n/m}^{-1}(\mathbf{P}_{n/m}(L)) \,||\, \mathbf{P}_{n/m}(\mathbf{P}_{n/m}^{-1}(R)) = (L \,||\, R) = X.$$

Рис. 3.24. Блок управляемых инволюций $\mathbf{I}_{2n/m}$ с повышенным быстродействием

Таким образом, схема на рис. 3.24 реализует блок управляемых перестановочных инволюций $\mathbf{I}_{2n/m}$ при произвольном блоке $\mathbf{P}_{n/m}$. Мощность множества реализуемых инволюций равна мощности множества различных перестановок, реализуемых блоком $\mathbf{P}_{n/m}$. В частности, если все модификации блока $\mathbf{P}_{n/m}$ уникальны, то тогда реализуются ровно 2^m уникальных инволюций.

Следует отметить, что схема (см. рис. 3.24) реализует только малую часть реальных инволюций: не более $n!$ из возможных $2n!! = (2n-1)(2n-3)\times\ldots\times3\times1$ инволюций, не содержащих единичных циклов.

В заключение отметим, что в схеме (рис. 3.24) для блоков $\mathbf{P}_{n/m}$ и $\mathbf{P}^{-1}_{n/m}$ можно использовать разные управляющие векторы V и U, а именно $\mathbf{P}_{n/m(V)}$ и $\mathbf{P}^{-1}_{n/m(U)}$. В этом случае реализуется блок управляемых перестановок $\mathbf{P}_{2n/2m(\mathbf{G}(V\|U))}$, для которого функция $\mathbf{G}(V\,\|\,U)$ определяет взаимное расположение разрядов векторов V и U в суммарном управляющем векторе с учетом принятых соглашений.

Формула преобразования для блока управляющих перестановок $\mathbf{P}_{2n/2m(\mathbf{G}(V\|U))}$ имеет следующий вид:

$$L'\,\|\,R' = \mathbf{P}_{2n/2m(\mathbf{G}(V\|U))}(L\,\|\,R) = \mathbf{P}^{-1}_{n/m(U)}(R)\,\|\,\mathbf{P}_{n/m(V)}(L)\,.$$

Покажем универсальность схемы для выполнения обратного преобразования.

Обратное преобразование может быть выражено следующей формулой:

$$\mathbf{P}^{-1}_{2n/2m(\mathbf{G}(V\|U))} = \mathbf{P}_{2n/2m(\mathbf{G}(U\|V))}\,.$$

То есть для выполнения обратного преобразования достаточно векторы V и U поменять местами.

Действительно,

$$L''\,\|\,R'' = \mathbf{P}_{2n/2m(\mathbf{G}(U\|V))}(L'\,\|\,R') = \mathbf{P}^{-1}_{n/m(V)}(R')\,\|\,\mathbf{P}_{n/m(U)}(L') =$$

$$= \mathbf{P}^{-1}_{n/m(V)}(\mathbf{P}_{n/m(V)}(L))\,\|\,\mathbf{P}_{n/m(U)}(\mathbf{P}^{-1}_{n/m(U)}(R)) = L\,\|\,R.$$

3.4. Линейный криптоанализ управляемых перестановок

Метод линейного криптоанализа блочных шифров был предложен в 1993 г. М. Мацуи [86] для вскрытия ключа алгоритма DES. И хотя данный метод в качестве основной меры использует линейную корреляционную зависимость между значениями векторов входа и выхода, т. е. частный случай корреляционной зависимости двух функций, предложенный метод получил самостоятельное название, прежде всего, благодаря четкой формализации, а также достигнутому практическому результату по анализу DES. Так, с использованием метода линейного криптоанализа достаточно проанализировать порядка 2^{43} пар исходных и преобразованных блоков двоичных данных, что является нижней оценкой для алгоритма DES. Дальнейшее развитие метод получил в более поздних работах [60, 70, 87, 101], из числа которых, прежде всего, необходимо отметить статью [101], написанную К. Ниберг (K. Nyberg) в 1994 г. В статье [101], в частности, доказана фундаментальная теорема линейной аппроксимации блочных шифров, а также для DES-подобных шифров показано, каким образом с использованием нелинейных функций достичь доказуемой стойкости шифра к методу линейного криптоанализа. Необходимо отметить, что в большинстве статей, посвященных линейному криптоанализу, рассматривается случай независимых раундовых ключей, хотя данное условие выполняется далеко не во всех шифрах. Случай зависимых ключей или входных данных является гораздо более сложной задачей, и исследования в этой области далеко не завершены.

Линейный криптоанализ блоков управляемых перестановок как криптографических примитивов рассмотрим с использованием базового метода, поскольку он позволяет понять внутренние механизмы данного метода как при формировании специальных характеристик, так и при их согласовании или вычислении.

3.4.1. Основные понятия линейного криптоанализа

Линейный криптоанализ [86] исследует статистически устойчивые линейные зависимости между некоторыми фиксированными подмножествами битов ключа, а также исходных и преобразованных блоков двоичных данных для *предсказания* значений некоторых битов ключа по совокупности имеющихся исходных и соответствующих им преобразованных блоков двоичных данных. Фактически он является комбинированным методом [7], сочетающим в себе поиск статистических линейных зависимостей, анализ имеющихся исходных и преобразованных блоков данных, а также использующим методы согласования и перебора.

Пусть $X \in \mathrm{GF}(2)^n$. Тогда каждому подмножеству $\mathbf{M} = [i_1, i_2, \ldots, i_s]$ номеров разрядов входного двоичного вектора X $(0 \le s \le n)$ можно поставить в соответствие бинарный вектор $a = (a_1, a_2, \ldots, a_n)$, в котором $a_j = 1$, если $j \in \mathbf{M}$, и $a_j = 0$ — в про-

тивном случае. При необходимости для множества **M** можно использовать уточняющий индекс $\mathbf{M}_a$ или обозначение $[a] = [i_1, i_2, \ldots, i_s]$.

В базовом методе линейного криптоанализа применительно к векторной булевой функции $Y = \mathbf{F}(X,K)$, где $X, Y \in \mathrm{GF}(2)^n$, $V \in \mathrm{GF}(2)^m$, исследуются линейные соотношения между подмножествами значений для заданных разрядов входных и выходных векторов на предмет наличия статистически устойчивых линейных зависимостей следующего вида:

$$a \bullet X \oplus b \bullet \mathbf{F}(X,K) \oplus c \bullet K = \varepsilon, \tag{3.27}$$

где $\varepsilon \in \mathrm{GF}(2)$ — константа, a, b ($b \neq e_0$), c — фиксированные вектора, именуемые *масками*, причем a всегда соответствует входному (исходному) вектору X, b — выходному (преобразованному) значению Y функции $\mathbf{F}(X,K)$, а c — ключу K.

Поскольку уравнение (3.27) справедливо не для всех значений X и K, говорят о вероятностном характере линейной зависимости. Если при этом выполнено условие независимого случайного и равновероятного выбора значений X и K и вероятность

$$P = \Pr_{X,K}\{a \bullet X \oplus b \bullet \mathbf{F}(X,K) \oplus c \bullet K = \varepsilon\} \neq \frac{1}{2}, \tag{3.28}$$

тогда говорят о наличии статистически устойчивой линейной зависимости.

Если $P > \frac{1}{2}$, то говорят, что выполняется уравнение (3.27). Если $P < \frac{1}{2}$, тогда говорят, что уравнение (3.27) выполняется для $\varepsilon' = 1 - \varepsilon$.

Поскольку определяющим является значение $|P - \frac{1}{2}|$, в уравнении (3.27) полагают $\varepsilon = 0$, а значение

$$p = p(a,b,c) = \Pr_{X,K}\{a \bullet X \oplus b \bullet \mathbf{F}(X,K) \oplus c \bullet K = 0\} - \frac{1}{2}, \tag{3.29}$$

называют *смещением*. Конкатенацию $(a, b, c) = a \parallel b \parallel c \in \mathrm{GF}(2)^{2n+m}$ будем называть *объединенной маской*. Очевидно, что $-\frac{1}{2} \leq p \leq \frac{1}{2}$.

В случае если условие равновероятного использования ключей не выполняется, тогда необходимо пользоваться формулой полной вероятности, а именно:

$$\begin{aligned} p = p(a,b,c) &= \left(\sum_k \Pr_X\{a \bullet X \oplus b \bullet \mathbf{F}(X,k) \oplus c \bullet k = 0\} \Pr_K\{K = k\}\right) - \frac{1}{2} = \\ &= \sum_k p_k(a,b,c) \Pr_K\{K = k\}, \end{aligned} \tag{3.30}$$

где

$$p_k = p_k(a,b,c) = \Pr_X\{a \bullet X \oplus b \bullet \mathbf{F}(X,k) \oplus c \bullet k = 0\} - \frac{1}{2} \qquad (3.31)$$

смещение для объединенной маски (a, b, c) и фиксированного ключа k, а $\Pr_K\{K = k\}$ — вероятность использования ключа k.

Хотя в большинстве статей предполагается, что все ключи равновероятны, на практике это условие реализуется не всегда. Более того, используется не все множество ключей. Например, вместо ключа K может рассматриваться некоторый вектор V, зависящий от ключа K и части входных данных. Формула (3.30) учитывает данные особенности.

При исследовании отдельных функций для смещений часто используют уточняющие индексы, например, $p^{(\mathbf{F})}$ или $p^{(\mathbf{F}_k)}$.

На практике в линейном криптоанализе задача определения некоторых битов ключа решается методом от частного к общему, а именно:

- выявление и анализ для отдельной операции статистически устойчивых линейных зависимостей, при этом смещения в совокупности с соответствующими им масками входных и выходных векторов являются характеристиками, которые будем называть *линейными характеристиками*;
- согласование *линейных характеристик* для всей последовательности преобразований внутри одного раунда и построение на их основе *однораундовой характеристики*;
- построение результирующей характеристики путем согласования раундовых характеристик, что позволяет получить вероятностную оценку четности нескольких битов ключа по совокупности имеющихся пар входных и соответствующих им выходных векторов.

Однораундовой характеристикой [43] называется четверка $(a, b, c, ½ + p)$, где a, b, c — маски входных, выходных векторов и ключа, а p — смещение, соответствующее уравнению (3.29). Характеристики, для которых абсолютное значение смещения равно ½, называют *наилучшими.* Для согласования двух характеристик $(a^{(1)}, b^{(1)}, c^{(1)}, ½ + p_1)$ и $(a^{(2)}, b^{(2)}, c^{(2)}, ½ + p_2)$ *с независимыми ключами*, в соответствии с теорией необходимо и достаточно, чтобы выполнялось условие: $a^{(2)} = b^{(1)}$. Причем результирующая характеристика будет иметь вид: $(a^{(1)}, b^{(2)}, c^{(1)} \oplus c^{(2)}, ½ + p')$. Для DES с независимыми ключами $p' = 2p_1p_2$.

Поэтому при построении раундовой функции необходимо стремиться к тому, чтобы максимальное абсолютное значение смещения такой функции было меньше некоторого заданного значения $0 < \delta < ½$. Для случая зависимых ключей вычисление смещения является достаточно сложной задачей.

Анализ отдельной операции предполагает построение таблицы линейных зависимостей между некоторыми подмножествами входных и выходных битов и ее использование для формирования таблицы линейных характеристик. Структура линейных характеристик, а также правила согласования, в целом, аналогичны раундовым характеристикам.

Рассмотрим более подробно, как формируется таблица линейных характеристик для отдельной операции. Пусть $\mathbf{F}$: $\mathbf{GF}(2)^r \to \mathbf{GF}(2)^n$ — векторная булева функция, заданная на множестве двоичных векторов X длины r ($r \geq n$). *Таблица линейных зависимостей* для функции $\mathbf{F}$ состоит из 2^r строк и 2^n столбцов, причем элементами таблицы являются смещения p линейной характеристики (a, b, $½ + p$), а координатами — значения векторов a и b. Фактически, для таблицы линейных зависимостей значения вычисляются в соответствии с формулой

$$p = p(a,b) = \Pr_X\{a \bullet X \oplus b \bullet \mathbf{F}(X) = 0\} - \frac{1}{2}. \tag{3.32}$$

Определение аналитической зависимости между конфигурацией масок и значением смещения для больших значений r и n является достаточно сложной задачей и в общем виде решается не всегда. Однако для небольших значений r и n данная задача может быть решена с использованием компьютеров. Второй проблемой является наглядное представление и анализ таблицы линейных зависимостей при больших длинах векторов входа и выхода, включая и случай нескольких входов. Поэтому предлагается использовать форму представления данных в виде таблицы, в каждой строке которой приведены параметры конкретной линейной характеристики, причем таблица упорядочена по убыванию абсолютного значения смещения и содержит только линейные характеристики с ненулевым смещением. При этом значения масок удобно представлять в шестнадцатеричном виде с указанием младшего разряда. На практике в таблицах линейных характеристик наряду со значением p целесообразно использовать параметр $p' = |2p|$. Примером таблицы линейных характеристик является табл. 3.2.

Рассмотрим некоторые общие свойства линейных характеристик, связанные с формой представления входных и выходных данных.

Композиция и декомпозиция входных и выходных векторов

Представим вектор X и соответствующую ему маску a в виде конкатенации векторов меньшей длины: $X = X^{(1)} \parallel X^{(2)} \parallel \ldots \parallel X^{(s)}$ и $a = a^{(1)} \parallel a^{(2)} \parallel \ldots \parallel a^{(s)}$.

Тогда функцию $\mathbf{F}$ можно рассматривать как некоторую функцию $\mathbf{F}'$ от s переменных $\mathbf{F}'(X^{(1)}, X^{(2)}, \ldots, X^{(s)})$ с характеристикой ($a^{(1)}$, $a^{(2)}$, …, $a^{(s)}$, b, $½ + p$). При этом уравнение (3.32) преобразуется к следующему виду:

$$p = \Pr_{X_1, X_2, \ldots, X_s}\{a^{(1)} \bullet X^{(1)} \oplus a^{(2)} \bullet X^{(2)} \oplus \ldots \oplus a^{(s)} \bullet X^{(s)} \oplus b \bullet \mathbf{F}'(X^{(1)}, X^{(2)}, \ldots, X^{(s)}) = 0\} - \frac{1}{2}. \tag{3.33}$$

Отсюда, если функция **F** зависит от двух переменных X и K, то, полагая $s = 2$ и $X^{(2)} = K$, в линейной характеристике маской ключа K является $a^{(2)} = c$. Такое представление одинаково справедливо как для фиксированного ключа k, так и в случае, когда ключ выбирается из некоторого множества ключей $\{K\}$. При этом линейная характеристика будет иметь следующий вид: $(a, c, b, ½ + p)$. В частности, можно рассматривать характеристики с нулевой маской ключа ($c = e_0$).

Справедлив и обратный переход от s независимых переменных к одной переменной. То есть, если $\mathbf{F}'$ — функция от s независимых переменных $\mathbf{F}'(X^{(1)}, X^{(2)}, \ldots, X^{(s)})$, для которой построена линейная характеристика $(a^{(1)}, a^{(2)}, \ldots, a^{(s)}, b, ½ + p)$, то, рассматривая конкатенацию векторов $X^{(1)}, X^{(2)}, \ldots, X^{(s)}$ в качестве новой переменной X: $X = X^{(1)} \| X^{(2)} \| \ldots \| X^{(s)}$, функцию $\mathbf{F}'$ можно представить как функцию от одной переменной большей размерности, причем размерность новой переменной X равна сумме размерностей всех переменных X_i, а вместо s масок $a^{(1)}, a^{(2)}, \ldots, a^{(s)}$ необходимо рассматривать их конкатенацию $a = a^{(1)} \| a^{(2)} \| \ldots \| a^{(s)}$. Требование независимости переменных в данном случае является существенным. Аналогичным образом рассматриваются линейные характеристики, в которых представление одной переменной в виде объединения нескольких рассматривается для вектора выхода Y.

Согласование линейных характеристик

Согласование линейных характеристик для двух преобразований, выполняемых последовательно одно за другим, проследим на примере, в котором в качестве первого преобразования $\mathbf{F}_1(X, K)$ рассматривается операция $X \oplus K$, а в качестве второго — некоторое преобразование $\mathbf{F}_2(X)$ с наилучшей линейной характеристикой $(a', b', ½ + p')$. В принятой выше терминологии для преобразования $\mathbf{F}_1$ линейная характеристика имеет вид: $(a, c, b, ½ + p^{\oplus})$, причем, как известно, $p^{\oplus} = ½$, тогда и только тогда, когда $a = c = b$, и $p^{\oplus} = 0$ в остальных случаях. То есть, для операции $\oplus$ все линейные характеристики с совпадающими масками являются наилучшими. Но тогда наилучшая суммарная линейная характеристика $(a'', c'', b'', ½ + p'')$ с учетом правила согласования будет иметь вид $(a'' = a, c'' = a, b'' = b', ½ + p'')$, поскольку $a' = b = a$ и $p'' = 2p^{\oplus}p' = p'$.

Покажем теперь, как вычисляется суммарная линейная характеристика для двух выполняемых последовательно преобразований, зависящих от одного и того же ключа. Пусть $(a^{(1)}, c^{(1)}, b_j, ½ + p_j)$ и $(a^{(2)} = b_j, c^{(2)}, b^{(2)}, ½ + p_j')$ — пара линейных характеристик, соответствующая маске b_j.

В случае независимых ключей суммарная линейная характеристика имеет вид: $(a^{(1)}, c^{(1)} \oplus c^{(2)}, b^{(2)}, ½ + p')$, где $p' = 2\sum_{b_j} p_j p_j'$.

Если маски $c^{(1)}$ и $c^{(2)}$ принадлежат одной и той же переменной K, тогда суммарная линейная характеристика будет иметь вид $(a^{(1)}, c, b^{(2)}, ½ + p')$, где $p' = 2\sum_{b_j,\, c^{(1)} | c^{(1)} \oplus c^{(2)} = c} p_j p_j'$.

Справедливость такого представления базируется на том факте, что разряды масок, принадлежащие одной и той же переменной и учитываемые в уравнении (3.32) четное число раз независимо от конкретных значений переменной в данных разрядах, не оказывают никакого влияния на вычисление смещения по формуле (3.32). В данном примере для преобразований, в которых полностью или частично совпадают входные параметры, показан принцип формирования суммарных линейных характеристик. Измененный порядок расположения масок по сравнению с раундовыми характеристиками, предложенными в [43], не является принципиальным, а обусловлен лишь методикой изложения.

В более поздних работах, посвященных изучению стойкости к линейному криптоанализу отдельных функций, часто вместо смещения p рассматривается его аналог $|p|$ [101] или $(2p)^2$ [87]. Правомерно изучение параметра $p' = |2p|$, для которого справедливо: $\forall p_i$: $0 \le p'' \le p'_i \le 1$.

Некоторые результаты этих работ, а также параметр p', мы рассмотрим далее. Здесь лишь отметим, что в качестве оценки стойкости функции к линейному криптоанализу рассматривают максимальное значение $|p|$ по всем характеристикам, в которых маска выхода не равна e_0. Очевидно, что если $|p_0|$ — максимальное значение для $|p|$, то и значения p_0', p_0'' также максимальные, причем соответствуют той же характеристике, что и $|p_0|$.

3.4.2. Линейные характеристики блоков управляемых перестановок

Прежде чем переходить к непосредственному вычислению смещений для блоков управляемых перестановок $\mathbf{P}(X,V)$, введем дополнительные обозначения:

- $\mathbf{V}_{(a\to b)} \overset{def}{=} \{v \in \mathbf{V} : \mathbf{P}(a,v) = b\}$ — множество векторов v, для которых выполняется условие $b = \mathbf{P}(a, v)$;
- $\boldsymbol{\lambda}_V(a,b) \overset{def}{=} \Pr_V\{\mathbf{V}_{(a\to b)}\}$ — вероятность события $V \in \mathbf{V}_{(a\to b)}$;
- $\mathbf{V}_{(a\to b;\, c)} \overset{def}{=} \{v \in \mathbf{V} : \mathbf{P}(a,v) = b,\ c \bullet v = 0\}$ — множество векторов v, для которых выполняется одновременно два условия: $b = \mathbf{P}(a, v)$ и $c \bullet v = 0$;
- $\boldsymbol{\lambda}_V(a,b,c) \overset{def}{=} \Pr_V\{\mathbf{V}_{(a\to b;\, c)}\}$ — вероятность одновременного выполнения двух условий: $b = \mathbf{P}(a,V)$ и $c \bullet V = 0$.

В общем случае

$$\boldsymbol{\lambda}_V(a,b) \overset{def}{=} \Pr_V\{\mathbf{V}_{(a\to b)}\} = \sum_v \theta_v(a,b) \Pr_V\{V = v\}, \tag{3.34}$$

где

$$\theta_v(a,b) \stackrel{def}{=} \begin{cases} 1, & \text{если} \quad \mathbf{P}(a,v) = b \\ 0, & \text{если} \quad \mathbf{P}(a,v) \neq b \end{cases}. \tag{3.35}$$

Соответственно

$$\boldsymbol{\lambda}_V(a,b,c) = \sum_v \theta_v(a,b,c)\,\mathrm{Pr}_V\{V = v\}, \tag{3.36}$$

где

$$\theta_v(a,b,c) \stackrel{def}{=} \begin{cases} 1, & \text{если} \quad \mathbf{P}(a,v) = b \ \text{ и } \ c \bullet v = 0 \\ 0, & \text{в} \quad \text{противном случае} \end{cases}. \tag{3.37}$$

Для равномерно распределенной случайной величины $V \in \mathbf{V} = \mathrm{GF}(2)^m$ формула (3.34) имеет вид:

$$\boldsymbol{\lambda}_V(a,b) \stackrel{def}{=} \mathrm{Pr}_V\{\mathbf{V}_{(a\to b)}\} = \frac{\#\mathbf{V}_{(a\to b)}}{\#\mathbf{V}} = \frac{\#\mathbf{V}_{(a\to b)}}{2^m}. \tag{3.38}$$

Следующие утверждения, являющиеся следствием общих свойств перестановок, достаточно очевидны и приведены без доказательств.

$$\mathbf{P}_v(a) = b \Rightarrow \mathbf{P}_v(\bar{a}) = \bar{b} \quad \text{и} \quad \mathbf{V}_{(\bar{a}\to\bar{b})} = \mathbf{V}_{(a\to b)}. \tag{3.39}$$

$$\mathbf{P}_v(a^{(1)} \otimes a^{(2)}) = \mathbf{P}_v(a^{(1)}) \otimes \mathbf{P}_V(a^{(2)}). \tag{3.40}$$

$$\begin{gathered} \mathbf{P}_v(a) = b \Rightarrow \mathbf{WH}(a) = \mathbf{WH}(b) \\ \text{и } \mathbf{WH}(a) \neq \mathbf{WH}(b) \Rightarrow \mathbf{P}_v(a) \neq b \ \forall V \Rightarrow \lambda_V(a, b) = 0. \end{gathered} \tag{3.41}$$

$$\begin{gathered} \mathbf{WH}(a^{(1)} \oplus a^{(2)}) = \mathbf{WH}(a^{(1)}) + \mathbf{WH}(a^{(2)}) - \mathbf{WH}(a^{(1)} \otimes a^{(2)}) \Rightarrow a^{(1)} \otimes a^{(2)} = e_0 \Rightarrow \\ \Rightarrow \mathbf{WH}(a^{(1)} \oplus a^{(2)}) = \mathbf{WH}(a^{(1)}) + \mathbf{WH}(a^{(2)}). \end{gathered} \tag{3.42}$$

$$\mathbf{WH}(a) = \mathbf{WH}(b) \Rightarrow \lambda_v(a, b) = \lambda_v(a, b). \tag{3.43}$$

$$\lambda_V(a, b) = \lambda_V(a, b, e_0) \quad \text{и} \quad V_{(a\to b)} = V_{(a\to b;\, e_0)}. \tag{3.44}$$

$$\forall c \ \ \lambda_V(a, b, c) \leq \lambda_V(a, b). \tag{3.45}$$

Отметим, что часть из приведенных утверждений рассматривалась в предыдущих разделах.

Далее будем говорить, что X — *равномерно распределенная случайная величина* ($X \in \mathbf{X} = \mathrm{GF}(2)^n$), если

$$\Pr_X\{X = x\} = \frac{1}{\#\mathbf{X}} = \frac{1}{2^n}. \tag{3.46}$$

Лемма 3.2

Пусть X — независящая от V равномерно распределенная случайная величина, и $\mathbf{WH}(a) > 0$. Тогда

$$\Pr_X\{a \bullet X \oplus c \bullet V = 0\} = \frac{1}{2}. \tag{3.47}$$

Доказательство

Так как $\mathbf{WH}(a) > 0$, следовательно, существует разряд j: $a_j = 1$, $j \in \{1, 2, \ldots, n\}$, для которого булева функция $a \bullet X \oplus c \bullet V$ линейна по x_j, поскольку в выражении $a \bullet X \;\; a_j \cdot x_j = x_j$, а $c \bullet V$ вообще не зависит от X. Но тогда $a \bullet X \oplus c \bullet V$ — сбалансированная булева функция и

$$\Pr_X\{a \bullet X \oplus c \bullet V = 0\} = \frac{1}{\#\mathbf{X}} \sum_x \left(a \bullet X \oplus c \bullet V \oplus 1\right) = \frac{1}{\#\mathbf{X}} 2^{n-1} = \frac{1}{2^n} 2^{n-1} = \frac{1}{2}.$$

Что и требовалось доказать.

Лемма 3.3

Пусть $\mathbf{P}(X, V)$ — блок управляемых перестановок. Тогда $\forall X, b \in \mathbf{X}$ и $\forall V \in \mathbf{V}$ справедливо равенство:

$$B \bullet \mathbf{P}(X, V) = \mathbf{P}^{-1}(b, V) \bullet X. \tag{3.48}$$

Доказательство

Покажем, что формула (3.48) верна при любых фиксированных значениях x и v.

Пусть $X = x$, $V = v$ и b': $b' = \mathbf{P}_v^{-1}(b) = \mathbf{P}^{-1}(b, v)$. Тогда последовательно применяя замену b на $\mathbf{P}(b', v)$, формулы (3.15) и (3.13), получим:

$$b \bullet \mathbf{P}(x, v) = \mathbf{WH}_{\oplus}(b \otimes \mathbf{P}(x, v)) = \mathbf{WH}_{\oplus}(\mathbf{P}(b', v) \otimes \mathbf{P}(x, v)) =$$
$$= \mathbf{WH}_{\oplus}(\mathbf{P}(b' \otimes x), v)) = \mathbf{WH}_{\oplus}(b' \otimes x) = b' \bullet x = \mathbf{P}^{-1}(b, v) \bullet x.$$

Что и требовалось доказать.

Следствие 3.2

Пусть $\mathbf{F}(X, V)$ — блок управляемых перестановок. Тогда $\forall a, b, X \in \mathbf{X}$ и $\forall v \in \mathbf{V}$ справедливо равенство:

$$b \bullet \mathbf{P}(X, v) \oplus a \bullet X = (\mathbf{P}^{-1}(b, v) \oplus a) \bullet X. \tag{3.49}$$

Лемма 3.4

Пусть X — независящая от V равномерно распределенная случайная величина, и $\mathbf{P}(a,v) \neq b > 0$. Тогда для векторной булевой функции $\mathbf{P}(X, V)$ смещение $\boldsymbol{p}_v(a, b, c) = 0$.

Доказательство

С учетом формулы (3.31) и леммы 3.3

$$p_v(a,b,c) = \Pr_X\{a \bullet X \oplus b \bullet \mathbf{P}(X,v) \oplus c \bullet v = 0\} - \frac{1}{2} =$$

$$= \Pr_X\{(\mathbf{P}^{-1}(b,v) \oplus a) \bullet X \oplus c \bullet v = 0\} - \frac{1}{2}\ .$$

Так как $\mathbf{P}^{-1}(b, v) \neq a$, следовательно, $\mathbf{P}^{-1}(b, v) \oplus a \neq e_0$.

Но тогда $\mathbf{WH}(\mathbf{P}^{-1}(b, v) \oplus a) > 0$ и с учетом леммы 3.2

$$\Pr_X\{(\mathbf{P}^{-1}(b, v) \oplus a) \bullet X \oplus c \bullet v = 0\} = \frac{1}{2}.$$

Поэтому $\quad p_v(a,b,c) = \frac{1}{2} - \frac{1}{2} = 0.$

Лемма доказана.

Следствие 3.3

Пусть X — независящая от V равномерно распределенная случайная величина, и $\mathbf{WH}(a) \neq \mathbf{WH}(b) > 0$. Тогда для векторной булевой функции $\mathbf{P}(X, V)$ смещение $\boldsymbol{p}(a, b, c) = 0$.

Доказательство

С учетом формулы (3.30) и леммы 3.4

$$p(a,b,c) = \sum_v p_v(a,b,c) \Pr_V\{V = v\} = 0 \sum_v \Pr_V\{V = v\} = 0.$$

Следствие 3.4

Пусть X — независящая от V равномерно распределенная случайная величина. Если для векторной булевой функции $\mathbf{P}(X, V)$ смещение $\boldsymbol{p}(a, b, c) \neq 0$, тогда $\mathbf{WH}(a) = \mathbf{WH}(b)$.

Лемма 3.5

Пусть X — независящая от v равномерно распределенная случайная величина. Если при этом $b = \mathbf{P}(a, v)$, тогда

$$p_v(a,b,c) = \theta_v(a,b,c) - \frac{1}{2}. \tag{3.50}$$

Доказательство

Условие $b = \mathbf{P}(a, v)$ равносильно $\mathbf{P}^{-1}(b, v) = a$. Поэтому с учетом формулы (3.31) и леммы 3.3

$$p_v(a,b,c)+\frac{1}{2}=\Pr_X\{a\bullet X\oplus b\bullet \mathbf{P}(X,v)\oplus c\bullet v=0\}=$$

$$=\Pr_X\{(\mathbf{P}^{-1}(b,v)\oplus a)\bullet X\oplus c\bullet v=0\}=\Pr_X\{(a\oplus a)\bullet X\oplus c\bullet v=0\}=$$

$$=\Pr_X\{c\bullet v=0\}=\begin{cases}1, & \text{если } c\bullet v=0\\ 0, & \text{если } c\bullet v\neq 0\end{cases}=\theta_v(a,b,c).$$

Лемма доказана.

Теорема 3.6

О смещениях управляемых перестановок

Пусть X — независящая от V равномерно распределенная случайная величина. Тогда

$$p(a,b,c)=\boldsymbol{\lambda}_V(a,b,c)-\frac{1}{2}\boldsymbol{\lambda}_V(a,b) \tag{3.51}$$

Доказательство

$$p(a,b,c)=\sum_v p_v(a,b,c)\Pr_V\{V=v\}=$$

$$=\sum_{v:\mathbf{P}(a,v)=b} p_v(a,b,c)\Pr_V\{V=v\}+\sum_{v:\mathbf{P}(a,v)\neq b} p_v(a,b,c)\Pr_V\{V=v\}$$

Второе слагаемое равно нулю, т. к. с учетом леммы 3.4 $p_v(a, b, c) = 0$, если $\mathbf{P}(a, v) \neq b$.

Но тогда

$$p(a,b,c)=\sum_{v:\mathbf{P}(a,v)=b} p_v(a,b,c)\Pr_V\{V=v\}=$$

$$=\sum_{v:\mathbf{P}(a,v)=b}(\theta_v(a,b,c)-\frac{1}{2})\Pr_V\{V=v\}=$$

$$=\sum_v \theta_v(a,b)(\theta_v(a,b,c)-\frac{1}{2})\Pr_V\{V=v\}=$$

$$=\sum_v(\theta_v(a,b,c)-\theta_v(a,b)\frac{1}{2})\Pr_V\{V=v\}=$$

$$= \sum_{v} \theta_v(a,b,c)\{V = v\} - \frac{1}{2}\sum_{v} \theta_v(a,b)\,\mathrm{Pr}_V\{V = v\} =$$

$$= \boldsymbol{\lambda}_V(a,b,c) - \frac{1}{2}\boldsymbol{\lambda}_V(a,b).$$

Что и требовалось доказать.

Следствие 3.5

$\forall a, b \;\; 0 \le 2p(a, b) = \mathrm{Pr}_V\{\mathbf{V}_{(a\to b)}\} \le 1.$

Действительно,

$$2p(a, b) = 2p(a, b, e_0) = 2\lambda_V(a, b, e_0) + \lambda_V(a, b) =$$
$$= 2\lambda_V(a, b) - \lambda_V(a, b) = \lambda_V(a, b) = \mathrm{Pr}_V\{\mathbf{V}_{(a\to b)}\}.$$

Следствие 3.6

$\forall a, b$ справедливо равенство $p(\overline{a},\overline{b}) = p(a,b)$.

Доказательство очевидно, т. к. $\boldsymbol{\lambda}_V(\overline{a},\overline{b}) = \boldsymbol{\lambda}_V(a,b)$.

Следствие 3.7

$$\forall c \;\; |p(a, b, c)| \le p(a, b). \tag{3.52}$$

Действительно,

$$0 \le \lambda_V(a, b, c) \le \lambda_V(a, b) \;\Rightarrow$$
$$\Rightarrow |2\lambda_V(a, b, c) - \lambda_V(a, b)| \le \lambda_V(a, b, c) \;\Rightarrow\; |p(a, b, c)| \le p(a, b).$$

Следствие 3.8

Свойство полноты

$$\forall a, b \qquad \sum_{b:\, \mathbf{WH}(b)=\mathbf{WH}(a)} 2p(a,b) = 1 \quad \text{и} \quad \sum_{a:\, \mathbf{WH}(a)=\mathbf{WH}(b)} 2p(a,b) = 1. \tag{3.53}$$

Действительно, $\forall a, v \;\; \exists! b: b = \mathbf{P}(a,v) \;\Rightarrow\; \sum_{b} \boldsymbol{\lambda}_V(a,b) = 1$,

причем

$$\sum_{b} \boldsymbol{\lambda}_V(a,b) = \sum_{b:\, \mathbf{WH}(b)\neq\mathbf{WH}(a)} \boldsymbol{\lambda}_V(a,b) + \sum_{b:\, \mathbf{WH}(b)=\mathbf{WH}(a)} \boldsymbol{\lambda}_V(a,b) = \sum_{b:\, \mathbf{WH}(b)=\mathbf{WH}(a)} \boldsymbol{\lambda}_V(a,b).$$

Следовательно,

$$\sum_{b:\mathbf{WH}(b)=\mathbf{WH}(a)} 2p_V(a,b) = \sum_{b:\mathbf{WH}(b)=\mathbf{WH}(a)} \boldsymbol{\lambda}_V(a,b) = 1.$$

Подобным образом доказывается и вторая формула.

Аналогичные результаты в общем виде приведены в работе [87] для параметра $(2p)^2$.

Следствие 3.9

Если $a = b = e_0$ или $a = b = d_0$, тогда $2p(a, b) = 1$.

Следствие 3.10

Пусть $\mathbf{WH}(a) = 1$. Если $\forall b$: $\mathbf{WH}(b) = 1$ вероятность $\Pr_V\{\mathbf{V}_{(a\to b)}\}$ = const, тогда

$$2p(a,b) = \Pr_V\{\mathbf{V}_{(a\to b)}\} = \frac{1}{n}. \tag{3.54}$$

Действительно, поскольку $\Pr_V\{\mathbf{V}_{(a\to b)}\}$ = const $\forall b$: $\mathbf{WH}(b) = 1$, причем существует ровно n таких векторов. Следовательно, $\Pr_V\{\mathbf{V}_{(a\to b)}\} = \frac{1}{n}$. Но тогда $2p(a,b) = \frac{1}{n}$. Аналогичный результат был, например, получен в [74] для операции циклического сдвига.

Важный вывод

Для блока управляемых перестановок абсолютное значение смещения линейной характеристики с ненулевой маской ключа не превышает значения смещения линейной характеристики с нулевой маской ключа, которое равно половине вероятности события, что маска входа преобразуется в маску выхода.

Заметим, что в следствиях теоремы 3.6 используется удвоенное значение смещения. Поэтому, как уже отмечалось ранее, в линейных характеристиках целесообразно дополнительно использовать параметр $p' = |2p|$, который при нулевой маске ключа равен вероятности $\Pr_V\{\mathbf{V}_{(a\to b)}\}$, т. е. $p' = |2p| = 2\,p(a, b) = \Pr_V\{\mathbf{V}_{(a\to b)}\}$.

Основной практический результат теоремы 3.6 и ее следствий заключается в следующем:

- для блоков управляемых перестановок достаточно анализировать линейные характеристики только с нулевой маской ключа;
- вычисление смещения $p(a, b)$ равносильно вычислению вероятности $\Pr_V\{\mathbf{V}_{(a\to b)}\}$;
- достаточно анализировать смещения только для $\mathbf{WH}(a) \le n/2$;
- необходимо разрабатывать блоки управляемых перестановок, для которых выполняется условие: $\forall a, b \notin \{e_0, d_0\}$ $\Pr_V\{\mathbf{V}_{(a\to b)}\} \le 1/n$.

Рассмотрим последний пункт более подробно.

Пусть $\xi(t) = \max\limits_{a,b|w_H(a)=t} \Pr_V\{\mathbf{V}_{(a\to b)}\}$ — функция максимума значения $\Pr_V\{\mathbf{V}_{(a\to b)}\}$ для заданного веса $\mathbf{WH} = t$. Очевидно, что $\xi(n-t) = \xi(t)$, поэтому достаточно рассматривать данную функцию только для $t = 1, 2, \ldots, n/2$. Для $\mathbf{WH}(a) = 1$ с учетом следствия 3.8 $\xi(t) \geq 1/n$, т. к. число возможных значений векторов b, таких что $\mathbf{WH}(b) = 1$, не превышает n. Соответственно $\xi(t) = 1/n$ в случае, когда блок является блоком равновероятного смещения [18], т. е. блоком, в котором $\forall a, b \in \mathrm{GF}(2)^n$: $\mathbf{WH}(a) = \mathbf{WH}(b) = 1$ и $\Pr_V\{\mathbf{V}_{(a\to b)}\} = \mathrm{const}$. С увеличением значения t число всевозможных векторов b с заданным весом t равно $C_n^t \overset{def}{=} \binom{n}{t} > n$.

Следовательно, существуют объективные условия построения блоков управляемых перестановок, для которых функция $\xi(t)$ для значений $t = 1, 2, \ldots, n/2$ будет монотонно убывающей.

В частности, данному условию для $t = 1, 2, \ldots, h$ удовлетворяют рассматриваемые в *разд. 3.3* блоки h-го порядка.

3.4.3. Способы и практические рекомендации вычисления смещений для линейных характеристик блоков управляемых перестановок

В данном разделе рассматриваются два способа вычисления смещений для блоков управляемых перестановок, а также практические рекомендации по разбиению множеств входных и выходных масок на классы эквивалентности и вычисление смещений только для представителей каждого класса.

Вместо слов "маска входа a" и "маска выхода b" будем использовать термины "вектор a" и "вектор b" или просто a и b.

На рис. 3.25 представлен блок $\mathbf{P}_{4/4}$ с параметрами $n = 4$, $m = 4$, реализующий 16 уникальных перестановок, который мы будем использовать в качестве поясняющего примера. Блок состоит из четырех блоков $\mathbf{P}_{2/1}$, каждый из которых при нулевом значении управляющего сигнала реализует тождественное преобразование.

Способ 1. Пусть $\mathbf{T}$ — табличное представление для $\mathbf{P}(X, V)$, в котором каждая строка соответствует конкретному значению управляющего вектора V и содержит:

- двоичное представление вектора $V = (v_1, v_2, \ldots, v_m)$, его индекс $/V/$ ($/V/ \geq 0$) и вероятность использования $P_v = \Pr_V\{V = v\}$;
- конкретный вид реализуемой перестановки, заданный в n первых столбцах, причем в столбце с номером i ($i = 1, 2, \ldots, n$) указан номер разряда выхода j, т. е. $E_j \equiv \mathbf{P}(E_i, v)\ \forall E_i \in \mathrm{GF}(2)^n$.

Рис. 3.25. Блок $\mathbf{P}_{4/4}$

Тогда вычисление $p' = \Pr_V\{\mathbf{V}_{(a\to b)}\}$ выполняется следующим образом:

- из n первых столбцов выбираются только такие номера i ($1 \le i \le n$), для которых в векторе $a = (a_1, a_2, \ldots, a_n)$ значения $a_i = 1$;
- выбираются строки, для которых в отмеченных столбцах, заданных вектором a, *содержатся все номера разрядов*, соответствующих ненулевым значениям b_j вектора b, т. е. для данных строк $\theta_V(a, b) = 1$;
- для выбранных строк вычисляется сумма вероятностей P_v и, тем самым, вычисляется вероятность $\Pr_V\{\mathbf{V}_{(a\to b)}\} = \lambda_V(a, b)$.

Пусть **T** — табличное представление для $\mathbf{P}_{4/4}$ (табл. 3.1) и $a = (0,1,0,1)$, $b = (1,0,1,0)$. Тогда $[a] = [2,4]$, а $[b] = [1,3]$.

По табл. 3.1 находим, что только для $V = (0,1,1,0)$ и $V = (1,0,0,1)$ во втором и четвертом столбцах присутствуют номера 1 и 3. Следовательно,

$$p' = \Pr_V\{\mathbf{V}_{(a\to b)}\} = \lambda_V(a, b) = P_6 + P_9.$$

Для вычисления смещения с ненулевой маской управляющего вектора $V = v$, согласно теореме 3.6, необходимо вычислить значение $\lambda_V(a, b, c)$. Для этого в уже отмеченных при вычислении значения $\lambda_V(a, b)$ строках необходимо выбрать те из них, для которых в соответствии с маской c четность суммы выделенных битов вектора v равна 1 (т. е. $c \bullet v \oplus 1 = 0$), и вычислить сумму вероятностей P_v только для этих строк.

Полученная сумма и является значением $\lambda_V(a, b, c)$.

С учетом формулы (3.51) теоремы 3.6 для вычисления смещения:

$$p_V(a,b,c) = \boldsymbol{\lambda}_V(a,b,c) - \frac{1}{2}\boldsymbol{\lambda}_V(a,b)$$

абсолютная величина удвоенного смещения примет следующий вид:

$$p' = |2\lambda_V(a, b, c) - \lambda_V(a, b)|.$$

Таблица 3.1. *Таблица линейных зависимостей для блока* $\boldsymbol{P}_{4/4}$

1	2	3	4	$\theta_V(a, b) = 1$	$\lvert V\rvert$	v_1	v_2	v_3	v_4	P_v	p
1	3	2	4		0	0	0	0	0	P_0	
1	4	2	3		1	0	0	0	1	P_1	
2	3	1	4		2	0	0	1	0	P_2	
2	4	1	3		3	0	0	1	1	P_3	
1	3	4	2		4	0	1	0	0	P_4	
1	4	3	2		5	0	1	0	1	P_5	
2	3	4	1	*	6	0	1	1	0	P_6	*
2	4	3	1		7	0	1	1	1	P_7	
3	1	2	4		8	1	0	0	0	P_8	
4	1	2	3	*	9	1	0	0	1	P_9	*
3	2	1	4		10	1	0	1	0	P_{10}	
4	2	1	3		11	1	0	1	1	P_{11}	
3	1	4	2		12	1	1	0	0	P_{12}	
4	1	3	2		13	1	1	0	1	P_{13}	
3	2	4	1		14	1	1	1	0	P_{14}	
4	2	3	1		15	1	1	1	1	P_{15}	

Пусть в нашем примере $c = (c_1, c_2, c_3, c_4,) = (1,1,1,0)$.

Тогда для векторов V_6 и V_9 достаточно проверить выполнение условия $c \bullet v \oplus 1 = 0$, т. е. проверить нечетность суммы битов первого, второго и третьего разрядов.

Так как $c \bullet v \oplus 1 = 0$ только для V_9, следовательно, $\lambda_V(a, b, c) = P_9$, а

$$p(a,b,c) = \boldsymbol{\lambda}_V(a,b,c) - \frac{1}{2}\boldsymbol{\lambda}_V(a,b) = P_9 - \frac{1}{2}(P_6 + P_9) = \frac{1}{2}(P_9 - P_6)\,.$$

Таким образом, при наличии таблицы **T** вычисление любой характеристики выполняется за один просмотр таблицы, содержащей 2^m строк.

Однако при больших значениях m такую таблицу построить затруднительно или невозможно, поэтому, как отмечалось выше, достаточно использовать таблицу линейных характеристик, содержащую только линейные характеристики с ненулевым смещением, упорядоченную по убыванию параметра p'.

В табл. 3.2 приведен фрагмент такой упорядоченной таблицы линейных характеристик для $\mathbf{P}_{4/4}$, по которой можно проверить справедливость теоремы 3.6 и ее следствий.

Таблица 3.2. *Фрагмент таблицы линейных характеристик для блока* $\boldsymbol{P}_{4/4}$

A	*B*	*C*	*P*	*2p*	*A*	*B*	*C*	*P*	*2p*	*A*	*B*	*C*	*P*	*2p*	*A*	*B*	*C*	*P*	*2p*
0	0	0	0.500	1.000	9	3	0	0.125	0.250	9	9	0	0.062	0.125	1	4	2	–0.125	0.250
15	15	0	0.500	1.000	9	12	0	0.125	0.250	9	10	0	0.062	0.125	1	8	2	0.125	0.250
1	1	0	0.125	0.250	10	3	0	0.125	0.250	10	5	0	0.062	0.125	2	4	2	–0.125	0.250
1	2	0	0.125	0.250	10	12	0	0.125	0.250	10	6	0	0.062	0.125	2	8	2	0.125	0.250
1	4	0	0.125	0.250	11	7	0	0.125	0.250	10	9	0	0.062	0.125	3	5	2	–0.125	0.250
1	8	0	0.125	0.250	11	11	0	0.125	0.250	10	10	0	0.062	0.125	3	6	2	–0.125	0.250
2	1	0	0.125	0.250	11	13	0	0.125	0.250	1	1	1	–0.125	0.250	3	9	2	0.125	0.250
2	2	0	0.125	0.250	11	14	0	0.125	0.250	1	2	1	0.125	0.250	3	10	2	0.125	0.250
2	4	0	0.125	0.250	12	5	0	0.125	0.250	2	1	1	–0.125	0.250	4	4	2	0.125	0.250
2	8	0	0.125	0.250	12	6	0	0.125	0.250	2	2	1	0.125	0.250	4	8	2	–0.125	0.250
3	5	0	0.125	0.250	12	9	0	0.125	0.250	3	5	1	–0.125	0.250	7	7	2	–0.125	0.250
3	6	0	0.125	0.250	12	10	0	0.125	0.250	3	6	1	0.125	0.250	7	11	2	0.125	0.250
3	9	0	0.125	0.250	13	7	0	0.125	0.250	3	9	1	–0.125	0.250	8	4	2	0.125	0.250
3	10	0	0.125	0.250	13	11	0	0.125	0.250	3	10	1	0.125	0.250	8	8	2	–0.125	0.250
4	1	0	0.125	0.250	13	13	0	0.125	0.250	4	1	1	0.125	0.250	11	7	2	–0.125	0.250
4	2	0	0.125	0.250	13	14	0	0.125	0.250	4	2	1	–0.125	0.250	11	11	2	0.125	0.250
4	4	0	0.125	0.250	14	7	0	0.125	0.250	7	13	1	–0.125	0.250	12	5	2	0.125	0.250
4	8	0	0.125	0.250	14	11	0	0.125	0.250	7	14	1	0.125	0.250	12	6	2	0.125	0.250
5	3	0	0.125	0.250	14	13	0	0.125	0.250	8	1	1	0.125	0.250	12	9	2	–0.125	0.250
5	12	0	0.125	0.250	14	14	0	0.125	0.250	8	2	1	–0.125	0.250	12	10	2	–0.125	0.250
6	3	0	0.125	0.250	5	5	0	0.062	0.125	11	13	1	–0.125	0.250	13	7	2	0.125	0.250
6	12	0	0.125	0.250	5	6	0	0.062	0.125	11	14	1	0.125	0.250	13	11	2	–0.125	0.250
7	7	0	0.125	0.250	5	9	0	0.062	0.125	12	5	1	0.125	0.250	14	7	2	0.125	0.250
7	11	0	0.125	0.250	5	10	0	0.062	0.125	12	6	1	–0.125	0.250	14	11	2	–0.125	0.250
7	13	0	0.125	0.250	6	5	0	0.062	0.125	12	9	1	0.125	0.250	3	5	3	0.125	0.250
7	14	0	0.125	0.250	6	6	0	0.062	0.125	12	10	1	–0.125	0.250	3	6	3	–0.125	0.250
8	1	0	0.125	0.250	6	9	0	0.062	0.125	13	13	1	0.125	0.250	3	9	3	–0.125	0.250
8	2	0	0.125	0.250	6	10	0	0.062	0.125	13	14	1	–0.125	0.250	3	10	3	0.125	0.250
8	4	0	0.125	0.250	9	5	0	0.062	0.125	14	13	1	0.125	0.250	12	5	3	0.125	0.250
8	8	0	0.125	0.250	9	6	0	0.062	0.125	14	14	1	–0.125	0.250	12	6	3	–0.125	0.250

Способ 2. Данный способ применим в тех случаях, когда блок управляемых перестановок имеет каскадную структуру, т. е. может быть представлен в виде суперпозиции двух или более последовательно применяемых блоков управляемых перестановок, причем, если

$$\mathbf{P}_V = \mathbf{P}^{<1>}_{V^{(1)}} \circ \mathbf{P}^{<2>}_{V^{(2)}} \circ \ldots \circ \mathbf{P}^{<s>}_{V^{(s)}},$$

где $V = V^{(1)} \| V^{(2)} \| \ldots \| V^{(s)}$ и, соответственно,

$$Y = \mathbf{P}^{<s>}_{V^{(s)}}(\ldots\mathbf{P}^{<2>}_{V^{(2)}}(\mathbf{P}^{<1>}_{V^{(1)}}(X))\ldots),$$

тогда

$$p' = 2p(a, b) = \Pr_V\{\mathbf{V}_{(a\to b)}\} =$$

$$= \sum_{b^{(1)},\ldots,b^{(s-1)}:\, w_H(b^{(s-1)})=\ldots=w_H(b^{(1)})=w_H(a)} \Pr_{V^{(1)}}(a \to b^{(1)}) \times \Pr_{V^{(2)}}(b^{(1)} \to b^{(2)}) \times \ldots \times P_{V^{(s-1)}}(b^{(s-1)} \to b) = \qquad (3.55)$$

$$= \sum_{b^{(1)},\ldots,b^{(s-1)}:\, w_H(b^{(s-1)})=\ldots=w_H(b^{(1)})=w_H(a)} p'(a, b^{(1)}) \times p'(b^{(1)}, b^{(2)}) \times \ldots \times p'(b^{(s-1)}, b).$$

Формула (3.55) является *следствием* доказанной выше теоремы 3.6.

Например, для $s = 3$ вычисление вероятности $\Pr_V\{\mathbf{V}_{(a\to b)}\}$ выполняется только по таким маскам $b^{(1)}$ и $b^{(2)}$, для которых одновременно $p'(a, b^{(1)}) \neq 0$, $p'(b^{(1)}, b^{(2)}) \neq 0$ и $p'(b^{(2)}, b) \neq 0$. В частности, блок $\mathbf{P}_{4/4}$ имеет каскадную структуру (см. рис. 3.25), причем $\mathbf{P}^{<2>}_{V(2)}$ является фиксированной перестановкой:

$$\mathbf{P}_\pi = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 3 & 2 & 4 \end{pmatrix}.$$

Пусть $a = (1,0,1,0)$ и $b = (0,0,1,1)$.

Тогда очевидно, что только для $b^{(1)} = (0,1,0,1)$ и $b^{(2)} = (0,0,1,1)$ рассматриваемые смещения одновременно отличны от нуля, причем $p'(a, b^{(1)}) = 1/4$, $p'(b^{(1)}, b^{(2)}) = 1$ и $p'(b^{(2)}, b) = 1$. Следовательно, $p' = \Pr_V\{\mathbf{V}_{(a\to b)}\} = 1/4$.

Разбиение множества объединенных масок на классы эквивалентности

Кроме предложенных выше способов необходимо учитывать особенности конкретной реализации блоков управляемых перестановок. Так как равенство является отношением эквивалентности, следовательно, все множество линейных характеристик можно представить в виде объединения непересекающихся смежных классов пар векторов a и b относительно значений $p' = \Pr_V\{\mathbf{V}_{(a\to b)}\}$ и $\mathbf{WH} = \mathbf{WH}(a) = \mathbf{WH}(b)$.

Класс эквивалентности, которому принадлежит пара (a, b), обозначим через $[(a, b)]_{p', \mathbf{WH}}$. При таком разбиении достаточно вычислить смещение только для одного представителя каждого класса. Существуют различные приемы формирования таких классов.

Определение 3.22

Разряды выхода i и j *эквиваленты* относительно a, если $\forall b$: $\mathbf{WH}(b) = \mathbf{WH}(a) - 1$, $b_i = b_j = 0$ и $\mathrm{Pr}_V\{\mathbf{V}_{(a\to b')}\} = \mathrm{Pr}_V\{\mathbf{V}_{(a\to b'')}\}$, где $b' = b \oplus E_i$, $b'' = b \oplus E_j$. Обозначение: $(i, j)_a$.

Очевидно, что для фиксированного a все множество номеров разрядов выхода $\mathbf{M}^{(\mathrm{B})} = [1, 2, \ldots, n]$ можно представить в виде объединения непересекающихся подмножеств эквивалентности:

$$\mathbf{M}^{(\mathrm{B})} = (i_1^{(1)}, i_2^{(1)}, ..., i_{s_1}^{(1)})_a \cup (i_1^{(2)}, i_2^{(2)}, ..., i_{s_2}^{(2)})_a \cup ... \cup (i_1^{(t)}, i_2^{(t)}, ..., i_{s_t}^{(t)})_a,$$

где $t \geq 1$ и $s_1 + \ldots + s_t = n$.

Пусть $\mu(\mathbf{WH}(a), t, s_1, \ldots, s_t)$ — число вариантов распределения $\mathbf{WH}(a)$ "дробинок" в t "ящиков", при условии, что в "ящик" с номером j $(1 \leq j \leq t)$ помещается не более s_j "дробинок". Тогда для фиксированного a существует не более $\mu(\mathbf{WH}(a), t, s_1, \ldots, s_t)$ смежных классов $[(a, b)]_{p', a}$, причем достаточно специальным образом отобрать всего $\mu_{\mathrm{B}}(\mathbf{WH}(a)) = \mu(\mathbf{WH}(a), t, s_1, \ldots, s_t)$ векторов b, соответствующих конкретным вариантам распределения "дробинок", и вычислить для них $\mathrm{Pr}_V\{\mathbf{V}_{(a\to b')}\}$. Если $(\mu_1, \mu_2, \ldots, \mu_t)$ — один из вариантов распределения "дробинок" для фиксированного вектора a, где $0 \leq \mu_j \leq s_j$ и $\sum_j \mu_j = \mathbf{WH}(a)$, то достаточно рассмотреть $[\mathbf{B}] = \bigcup_{j:\, \mu_j \geq 1} [i_1^{(j)}, i_2^{(j)}, ..., i_{\mu_j}^{(j)}]$. Пусть, например, $\mathbf{P}_{4/4}$ и $a = (1,0,1,0)$. В этом случае $\mathbf{M}^{(\mathrm{B})} = [1,2]_a \cup [3,4]_a$, причем, если $(0,2)$ — один из вариантов распределения "дробинок", то тогда $[\mathbf{B}] = [3,4]$, т. е. $b = (0,0,1,1)$.

Аналогично вводится эквивалентность для разрядов входа относительно вектора b и смежные классы $[(a, b)]_{p', b}$.

Определение 3.23

Разряды выхода i и j *эквиваленты* относительно $\mathbf{WH}(a)$, если они эквиваленты $\forall a'$: $\mathbf{WH}(a') = \mathbf{WH}(a)$. Обозначение: $(i, j)_{\mathbf{WH}(a)}$.

Следует отметить, что если $(i_1, i_2, \ldots, i_{s_1})_{\mathbf{WH}(a)}$ — подмножество эквивалентности, то оно не обязательно является подмножеством эквивалентности для значения $\mathbf{WH} \neq \mathbf{WH}(a)$. Так, в $\mathbf{P}_{4/4}$ для $\mathbf{WH}(a) = 1$ $\mathbf{M}^{(\mathrm{B})} = (1,2,3,4)_1$, но для $\mathbf{WH}(a) = 2$ $\mathbf{M}^{(\mathrm{B})} = (1,2)_2 \cup (3,4)_2$.

Пусть

$$\mathbf{M}^{(\mathrm{B})} = (i_1^{(1)}, i_2^{(1)}, ..., i_{s_1}^{(1)})_{w_H(a)} \cup (i_1^{(2)}, i_2^{(2)}, ..., i_{s_2}^{(2)})_{w_H(a)} \cup ... \cup (i_1^{(t)}, i_2^{(t)}, ..., i_{s_t}^{(t)})_{w_H(a)},$$

$$\mu_{\mathrm{B}}(\mathbf{WH}) = \mu(\mathbf{WH}(a), t, s_1, s_2, \ldots, s_t).$$

Соответственно:

$$\mathbf{M}^{(\mathrm{A})} = (i_1^{(1)}, i_2^{(1)}, ..., i_{s_1'}^{(1)})_{w_H(b)} \cup (i_1^{(2)}, i_2^{(2)}, ..., i_{s_2'}^{(2)})_{w_H(b)} \cup ... \cup (i_1^{(t')}, i_2^{(t')}, ..., i_{s_{t'}'}^{(t')})_{w_H(b)},$$

$$\mu_{\mathrm{A}}(\mathbf{WH}) = \mu(\mathbf{WH}(b), t', s'_1, s'_2, \ldots, s'_{t'}).$$

Так как для $\Pr_V\{\mathbf{V}_{(a\to b')}\} > 0$ всегда выполняется условие $\mathbf{WH}(b) = \mathbf{WH}(a)$, следовательно, для каждого значения **WH** существует не более $\mu_{\mathrm{A}}(\mathbf{WH})\cdot\mu_{\mathrm{B}}(\mathbf{WH})$ классов эквивалентности $[(a, b)]_{p',\mathbf{WH}}$. Но тогда общее число классов эквивалентности равно сумме по **WH** от единицы до $n/2$.

Кроме того, при построении подмножеств эквивалентности можно использовать наличие различных видов симметрии в схеме построения блока управляемых перестановок. Например, блоки $\mathbf{P}_{2/1}$, $\mathbf{P}_{4/4}$, $\mathbf{P}_{16/32}$, $\mathbf{P}_{32/80}$ [28] являются блоками зеркальной симметрии (инвариантность относительно изменения порядка нумерации разрядов на противоположный), но только первые три из них имеют симметрию "входа/выхода", то есть симметрию, при которой вход и выход можно поменять местами.

Таким образом, полученные результаты позволяют сформулировать следующие выводы:

- использование блоков управляемых перестановок в качестве базового примитива с учетом стойкости к линейному криптоанализу представляется перспективным для построения стойких скоростных алгоритмов блочного шифрования данных;
- предложенные способы вычисления линейных характеристик можно использовать для оптимизации разрабатываемых блоков управляемых перестановок;
- при конструировании схем алгоритмов блочного шифрования данных целесообразно выбирать конструкции, исключающие или ограничивающие для блоков управляемых перестановок возможность использования объединенной маски ($a = d_0$, $b = d_0$, c).

3.5. Формирование управляющих векторов

3.5.1. Процедуры формирования управляющих векторов для блоков $\mathbf{P}_{n/m}$

При построении алгоритмов блочного шифрования данных на основе управляемых операций различают следующие варианты формирования управляющих векторов:

- по секретному ключу;
- по одному из подблоков входных данных;
- по секретному ключу и одному из подблоков входных данных.

В *первом* случае в течение всего периода использования одного и того же секретного ключа все модификации управляемых операций являются фиксированными. Данный вариант применяется, например, в блочном шифре ICE [82].

Во *втором* случае для каждой управляемой операции с изменением блока данных, участвующего в формировании управляющего вектора, изменяется и значение управляющего вектора, что влечет изменение модификации, участвующей в преобразовании. Однако определенным недостатком данного варианта является тот факт, что в некоторых случаях выбором модификаций можно управлять (например, в случае использования криптографической атаки на основе специально подобранных входных данных). Данный вариант формирования управляющего вектора применяется, например, в алгоритмах RC5 [115] и RC6 [114].

В *третьем* случае выбор используемой модификации зависит как от значения секретного ключа, так и от значения входного блока данных. Поэтому рассматриваемый вариант представляет наибольший интерес.

Классическим примером использования данного варианта формирования управляющих векторов является алгоритм DES, в котором 16-разрядный управляющий двоичный вектор V является результатом операции поразрядного сложения 16 битов ключа и 16 битов входного блока преобразуемых данных. Напомним, что сформированный 16-разрядный управляющий вектор, представленный в виде объединения восьми двоичных векторов длины два, управляет объединенным блоком подстановок $\mathbf{S}_{32/16} = \mathbf{S}_{4/2} \| \mathbf{S}_{4/2} \| \ldots \| \mathbf{S}_{4/2}$, состоящим из восьми блоков $\mathbf{S}_{4/2}$, причем каждый двоичный вектор длины два "выбирает" одну их четырех заданных подстановок длины четыре.

Необходимо отметить, что в алгоритме DES, как на входе S-блока, так и в управлении этим **S**-блоком, используются 32 и, соответственно, 16 разрядов одного и того же 32-разрядного блока данных. Такая конструкция стала возможной благодаря наличию в раундовой функции алгоритма DES *процедуры расширения входного вектора*, преобразующей 32-разрядный вектор в 48-разрядный.

Использование 48-разрядного раундового ключа для поразрядного сложения с вновь сформированным 48-разрядным двоичным вектором является тем условием достаточности, которое обеспечивает выбор не только всех возможных модификаций (всего 2^{16}), но и всех возможных значений для блока входных данных (всего 2^{32}).

Действительно, в данном случае 48-разрядный двоичный вектор объединенных входных данных можно рассматривать в качестве равномерно распределенной случайной величины, для которой выбор **S**-блоков, с учетом необходимых алгебраических и вероятностно-статистических свойств, сводится к решению некоторой математической задачи. Что и было сделано разработчиками фирмы IBM, причем требования, которым должны удовлетворять формируемые **S**-блоки, довольно продолжительное время оставались секретными. В настоящее время они известны.

Поскольку из 48 разрядов сформированного вектора 32 использовались в качестве входных данных для блока подстановок $\mathbf{S}_{32/16}$, имеет смысл говорить *о процедуре расширения управляющего вектора V*, для которой введем обозначение $\mathbf{E}(X, K_V)$. Первый параметр соответствует блоку входных данных, а второй — раундовому ключу K_V.

Построение той или иной процедуры $\mathbf{E}(X, K_V)$ существенным образом зависит от схемы одного раунда, от базовых требований, предъявляемых как ко всему раунду, так и к конкретному блоку управляемой операции, а также от конструкции управляемой операции.

В случае с DES такая процедура после перенумерации разрядов имеет достаточно простой вид, а именно:

$$v_j = l_j \oplus k_j \ (1 \le j \le 16),$$

где все l_j и k_j независимы друг от друга.

В случае использования в алгоритме блочного шифрования одного или нескольких блоков управляемых перестановок все гораздо сложнее. Главная проблема заключается в том, что разрядность блока входных данных X в каждом раунде, например, для 64-разрядного блочного шифра равна 64, в то время как длина управляющего вектора, например, для одного блока $\mathbf{P}_{32/80}$ равна 80. То есть в уравнениях типа $v_s = l_j \oplus k_i$ некоторые l_j и k_i обязательно будут повторяться.

Рассмотрим блок управляемых перестановок $\mathbf{P}_{n/m(V)}$ и процедуру расширения управляющего вектора $\mathbf{E}(L,K_V)$.

В записи $\mathbf{E}(L,K_V)$, вектор L — это половина входного блока данных, то есть $X = L \parallel R$, $\omega(L) = \omega(R) = n/2$, где ω — размерность. Так как необходимым условием реализуемости всех модификаций блоком $\mathbf{P}_{n/m(V)}$ является независимость и равномерная распределенность всех разрядов m-разрядного управляющего вектора V, следовательно, для обеспечения выполнения данного условия необходимо, чтобы число независимых переменных входного вектора $(L \parallel K_V)$ было не меньше m. Данное условие выполнимо, если $\omega(K_V) = m - \omega(L) \ge m - n/2$, поскольку $\omega(L) = n/2$.

При формировании процедуры $\mathbf{E}(L,K_V)$, как правило, необходимо обеспечить выполнение следующих требований:

- максимально высокое быстродействие при низкой схемотехнической сложности реализации;
- каждый разряд v_i вектора V должен с вероятностью ½ принимать значение 0 или 1 на всем множестве входных значений $\{L \parallel K_V\}$, т. е. булева функция $v_i = \mathbf{E}_i(L, K_V)$ должна быть сбалансированной;
- каждый разряд v_i вектора V должен существенно зависеть от вектора L, т. е. должны существовать L и j $(1 \le j \le n/2)$ такие, что $\mathbf{E}_i(L, K_V) \ne \mathbf{E}_i(L \oplus E_j, K_V)$;

- должна быть обеспечена равномерность участия всех разрядов входных данных L в управлении блоком управляемых перестановок;
- для произвольно выбранной коммутации одного разряда входа с одним разрядом выхода в управлении должны участвовать разные разряды вектора L и разные разряды ключа K_V, что гарантирует возможность реализации всех вариантов коммутации.

Данным требованиям удовлетворяют простейшие булевы функции $v_i = l_j$ и $v_i = l_j \oplus (K_V)_s$ $(1 \le i, j, s \le n)$, для первой из которых $\tau = 0$, а для второй $\tau = \tau_{\oplus}$. Именно данные два класса функций обычно используются при формировании процедуры $\mathbf{E}(L, K_V)$.

Необходимо отметить, что если вектор V_1 ($V_1 \in \mathrm{GF}(2)^{n/2}$ и $V = V_1 \parallel V_2 \parallel \ldots \parallel V_s$, где $s = 2m/n$) сформирован с использованием только булевых функций $v_i = l_j$ (например, $V = L$), то тогда процедура $\mathbf{E}(L, K_V)$ не создает дополнительную временную задержку для блока $\mathbf{P}_{n/m}$, что важно при реализации скоростного алгоритма.

Не менее важная особенность, которую необходимо учитывать при формировании процедуры $\mathbf{E}(L, K_V)$, — конструкция схемы всего раунда.

Так, например, в усовершенствованной схеме патента РФ 2140714 последовательно применяются два взаимно обратных блока управляемых перестановок: $\mathbf{P}_{n/m(V)}$ и $\mathbf{P}^{-1}_{n/m(U)}$. Если при этом используется одна и та же процедура расширения управляющего вектора, тогда обязательно ключи K_V и K_U должны быть различны. В противном случае, $V = \mathbf{E}(L,K_V) = \mathbf{E}(L,K_U) = U$, и часть раундового преобразования, выполняемого над правым подблоком данных, а именно: $\mathbf{P}^{-1}_{n/m(U)}(\mathbf{P}_{n/m(V)}(R))$, превращается в тождественное преобразование. То есть $\mathbf{P}^{-1}_{n/m(U)}(\mathbf{P}_{n/m(V)}(R)) = R$.

Кроме того, в конструкции $\mathbf{P}^{-1}_{n/m(U)}(\mathbf{P}_{n/m(V)}(R))$ существует еще одна особенность, и без ее учета могут быть нарушены требования, которым должна удовлетворять конструкция $\mathbf{P}^{-1}_{n/m(U)}(\mathbf{P}_{n/m(V)}(R))$.

Действительно, если при формировании вектора V_s, управляющего последним слоем блока $\mathbf{P}_{n/m}$, использовать булевы функции, зависящие только от блока входных данных, то есть $w_i = \mathrm{f}_i(L)$ и $V_s = W$, то для конструкции $\mathbf{P}^{-1}_{n/m(U)}(\mathbf{P}_{n/m(V)}(R))$ это равносильно использованию **P**-блоков с $s - 1$ слоем, поскольку, в соответствии с формулами (3.19) и (3.21)

$$\mathbf{P}_{n/m(V)} \circ \mathbf{P}^{-1}_{n/m(U)} = \mathbf{P}_{\pi^{(0)}} \circ \mathrm{L}_{n(V_1)} \circ \mathbf{P}_{\pi^{(1)}} \circ \mathrm{L}_{n(V_2)} \circ \ldots \circ \mathbf{P}_{\pi^{(s-1)}} \circ \mathrm{L}_{n(V_s)} \circ \mathbf{P}_{\pi^{(s)}} \circ$$

$$\circ (\mathbf{P}_{\pi^{(s)}})^{-1} \circ \mathrm{L}_{n(U_s)} \circ (\mathbf{P}_{\pi^{(s-1)}})^{-1} \circ \mathrm{L}_{n(U_{s-1})} \circ (\mathbf{P}_{\pi^{(s-2)}})^{-1} \circ \ldots \circ (\mathbf{P}_{\pi^{(1)}})^{-1} \circ \mathrm{L}_{n(U_1)} \circ (\mathbf{P}_{\pi^{(0)}})^{-1} =$$

$$= \mathbf{P}_{\pi^{(0)}} \circ \mathrm{L}_{n(V_1)} \circ \mathbf{P}_{\pi^{(1)}} \circ \mathrm{L}_{n(V_2)} \circ \ldots \circ \mathbf{P}_{\pi^{(s-1)}} \circ \mathrm{L}_{n(W)} \circ \mathbf{P}_{\pi^{(s)}} \circ$$

$$\circ (\mathbf{P}_{\pi^{(s)}})^{-1} \circ \mathrm{L}_{n(W)} \circ (\mathbf{P}_{\pi^{(s-1)}})^{-1} \circ \mathrm{L}_{n(U_{s-1})} \circ (\mathbf{P}_{\pi^{(s-2)}})^{-1} \circ \ldots \circ (\mathbf{P}_{\pi^{(1)}})^{-1} \circ \mathrm{L}_{n(U_1)} \circ (\mathbf{P}_{\pi^{(0)}})^{-1} =$$

$$= \mathbf{P}_{\pi^{(0)}} \circ \mathrm{L}_{n(V_1)} \circ \mathbf{P}_{\pi^{(1)}} \circ \ldots \circ \mathbf{P}_{\pi^{(s-2)}} \circ \mathrm{L}_{n(V_{s-1})} \circ \mathrm{L}_{n(U_{s-1})} \circ (\mathbf{P}_{\pi^{(s-2)}})^{-1} \circ \ldots \circ (\mathbf{P}_{\pi^{(1)}})^{-1} \circ \mathrm{L}_{n(U_1)} \circ (\mathbf{P}_{\pi^{(0)}})^{-1}.$$

Очевидно, что такая реализация процедуры расширения управляющего вектора не может быть признана удачной, поскольку сокращает число потенциально реализуемых модификаций. Конкретные модели процедур расширения управляющего вектора для заданных алгоритмов блочного преобразования данных (SPECTR–H64 и SPECTR–128) рассматриваются в *гл. 5*.

3.6. Основные результаты

К числу основных результатов исследования управляемых перестановочных операций, рассмотренных в данной главе, можно отнести следующие.

- Приведено математическое описание блоков управляемых перестановок, реализующих нелинейные подстановочные преобразования частного вида, сохраняющие вес Хэмминга. В ходе рассмотрения различных подходов к конструированию таких блоков (формальный, экономичный, универсальный, каскадный, рекурсивный) показано, что наилучшими показателями скорости при невысокой схемотехнической сложности обладают так называемые перестановочные сети (**P**-сети).
- Показана простота и эффективность реализации обратного преобразования в случае использования перестановочных сетей.
- Проанализированы конкретные модели блоков управляемых перестановок, обладающие заданными алгебраическими и вероятностно-статистическими свойствами (блоки равновероятного смещения, блоки h-го порядка, блоки максимального порядка, блоки полноцикловых перестановок, блоки с заданной цикловой структурой, уникальность, обратимость). С учетом приоритета заданных требований предложены варианты их схемотехнической реализации.
- Показано, что при одинаковой скорости и комбинационной сложности реализации блок равновероятного смещения и 1-го порядка, построенный с использованием рекурсивной модели, обладает более качественными перемешивающими свойствами по сравнению с операцией управляемого циклического сдвига, что обусловлено существенно большим числом реализуемых уникальных модификаций первой модели (2^n вместо n).
- На основе двух взаимно обратных блоков управляемых перестановок первого порядка рекурсивного типа предложен метод синтеза блоков максимального порядка, реализующих любые перестановки.

- Для блоков управляемых перестановок для линейных характеристик в методе линейного криптоанализа доказана теорема о вычислении смещений, выражающих корреляционную зависимость между входными и выходными параметрами исследуемого преобразования.
- Разработаны практические рекомендации эффективного вычисления смещений для заданных линейных характеристик, которые могут быть использованы при оптимизации разрабатываемых блоков управляемых перестановок.
- Рассмотрены основные принципы формирования управляющих векторов для блоков управляемых перестановок.

ГЛАВА 4

Управляемые подстановочные операции

4.1. Характеристика управляемых подстановочных преобразований

В соответствии с теоретическими основами криптографии [120] качество алгоритмов блочного шифрования непосредственно зависит от показателей "рассеивания" и "перемешивания" двоичной информации, которые обеспечиваются чередованием *подстановочных и перестановочных преобразований*. В данной главе рассматриваются теоретические и некоторые практические аспекты использования подстановочных преобразований. В современных блочных шифрах такие преобразования, как правило, связаны с применением криптографических примитивов двух типов:

- специальных нелинейных **S**-блоков, задаваемых в табличном виде;
- стандартных арифметических или алгебраических операций, имеющих реализацию в компьютерных командах.

Применение в блочных шифрах примитивов первого типа связано с разбиением преобразуемого блока данных из n битов на подблоки по k битов и формированием для этих подблоков небольших подстановок (**S**-блоков). Таблицы, определяющие такие подстановки, имеют размер, пропорциональный величине 2^k. Недостатки табличного представления подстановочных преобразований заключаются в сложности их аппаратной реализации и в быстром увеличении размеров таблиц с ростом значения k. Это обстоятельство делает проблематичным применение больших **S**-блоков как при аппаратной, так и при программной реализации блочных шифров. В связи с этим размеры **S**-блоков ограничиваются в основном 4—8 битами. Это существенно ухудшает ряд криптографических свойств преобразований, осуществляемых над всем блоком данных размера n, а также уменьшает общее многообразие (число *модификаций*) таких преобразований.

Подстановочные примитивы второго типа достаточно эффективно реализуются как стандартные операции в программном виде. Однако эти операции либо осуществляют линейные преобразования (операция XOR), либо имеют низкую нелинейность (операция сложения по модулю 2^n), либо связаны с большой сложностью аппаратной реализации (операции возведения в степень, умножения по модулю 2^n или простому модулю и др.).

При оценке качества любых криптографических примитивов следует учитывать сочетание следующих трех основных параметров, определяющих стойкость шифров к различным методам криптоанализа [120]:

- степени нелинейности;
- степени распространения изменений (ошибок);
- уровня корреляционной иммунности.

Подробно эти свойства и механизмы их влияния на криптографическую стойкость блочных шифров рассматриваются в следующих разделах данной главы.

Касаясь в общих чертах перечисленных свойств для подстановочных примитивов, следует иметь в виду, что упомянутые выше **S**-блоки предназначены обеспечить заданную в пределах каждого **S**-блока степень нелинейности и степень распространения ошибок. Вместе с тем, достаточно проблематично достижение высоких показателей одновременно по всем трем параметрам в отношении всего преобразуемого блока данных с использованием серии **S**-блоков малого размера.

С другой стороны, эффективно реализуемые программно и несложные при аппаратной реализации математические операции (XOR и сложение по модулю 2^n) обладают высокой корреляционной иммунностью для всего преобразуемого блока данных, но имеют невысокую степень нелинейности и распространения изменений. Следует отметить, что эти и другие математические операции вообще не рассматривались как криптографические примитивы блочных шифров, а использовались в криптографии лишь по причине их готовой реализации в виде команд процессора ЭВМ.

Таким образом, становится очевидной перспективность создания специальных криптографических примитивов, сочетающих и оптимизирующих положительные свойства используемых в блочных шифрах подстановочных преобразований.

Другим важным направлением совершенствования криптографических примитивов является повышение их степени неопределенности за счет увеличения размеров блока преобразуемых данных и введения дополнительных параметров, определяющих результаты преобразований и увеличивающих количество их модификаций. В частности, указанная идея может быть реализована в виде специальных операций, дополнительно управляемых некоторыми псевдослучайными векторами, зависящими от преобразуемых данных и ключей шифра. Как уже упоминалось в предыдущих главах, такие криптографические примитивы получили названия *управляемых преобразований*.

Сказанное можно пояснить на примерах существующих блочных шифров, где принципы действия управляемых преобразований нашли свое упрощенное применение, но еще не получили необходимого развития. Так, каждый из восьми **S**-блоков в DES, осуществляющих *сюръективное (см. разд. 4.2)* отображение $GF(2)^6 \to GF(2)^4$, можно рассматривать как управляемое подстановочное преобразование вида $GF(2)^2 \times GF(2)^4 \to GF(2)^4$, содержащее четыре бита преобразуемых данных и два управляющих бита [107]. Например, блок $\mathbf{S}_3$ осуществляет следующую комбинированную подстановку, состоящую из одной верхней строки, определяющей в числовом виде исходные битовые четырех-граммы, и из четырех нижних строк, задающих результирующие битовые четырех-граммы:

$$
\begin{array}{c}
 \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3}
\end{array}
\begin{array}{c}
\begin{array}{cccccccccccccccc}
\mathbf{0} & \mathbf{1} & \mathbf{2} & \mathbf{3} & \mathbf{4} & \mathbf{5} & \mathbf{6} & \mathbf{7} & \mathbf{8} & \mathbf{9} & \mathbf{10} & \mathbf{11} & \mathbf{12} & \mathbf{13} & \mathbf{14} & \mathbf{15}
\end{array} \\
\begin{bmatrix}
15 & 1 & 8 & 14 & 6 & 11 & 3 & 4 & 9 & 7 & 2 & 13 & 12 & 0 & 5 & 10 \\
3 & 13 & 4 & 7 & 15 & 2 & 8 & 14 & 12 & 0 & 1 & 10 & 6 & 9 & 11 & 5 \\
0 & 14 & 7 & 11 & 10 & 4 & 13 & 1 & 5 & 8 & 12 & 6 & 9 & 3 & 2 & 15 \\
13 & 8 & 10 & 1 & 3 & 15 & 4 & 2 & 11 & 6 & 7 & 12 & 0 & 5 & 14 & 9
\end{bmatrix}
\end{array}
$$

В указанной таблице четыре бита данных фиксируют номер столбца подстановки, а два бита управления — номер нижней строки подстановки, определяя тем самым переход битовой четырех-граммы в преобразовании. В результате для восьми **S**-блоков имеется $4^8 = 65\,536$ различных вариантов (модификаций) преобразования всего 32-битового блока данных ($GF(2)^{32} \to GF(2)^{32}$).

Аналитически функции преобразования для каждого **S**-блока в DES имеют достаточно сложную запись. Так, для блока $\mathbf{S}_3$ булева функция, определяющая первую из четырех компонентов выхода, выглядит следующим образом:

$$
\begin{gathered}
y_1 = x_1x_2x_3x_5x_6 \oplus x_1x_2x_3x_4x_5 \oplus x_1x_3x_4x_6 \oplus x_1x_2x_4x_6 \oplus x_1x_2x_3x_6 \oplus x_1x_3x_4x_5 \oplus x_1x_2x_4x_5 \oplus \\
\oplus x_1x_2x_3x_5 \oplus x_3x_4x_6 \oplus x_2x_5x_6 \oplus x_2x_4x_6 \oplus x_1x_4x_6 \oplus x_1x_2x_3 \oplus x_1x_3x_6 \oplus x_3x_4x_5 \oplus x_1x_3x_4 \oplus \\
\oplus x_4x_6 \oplus x_3x_6 \oplus x_4x_5 \oplus x_2x_5 \oplus x_3x_4 \oplus x_1x_3 \oplus x_3.
\end{gathered}
$$

В блочном шифре LOKI91 [118] используются уже управляемые отображения в четырех **S**-блоках вида $GF(2)^4 \times GF(2)^8 \to GF(2)^8$, в каждом из которых задается случайный выбор одной из шестнадцати подстановок на восьми битах исходных данных. Несмотря на увеличенные размеры **S**-блоков, в этом шифре так же, как и в DES, имеется всего $16^4 = 65\,536$ различных модификаций преобразования всего 32-битового блока данных.

Из приведенных примеров следует, что в ряде случаев сюръективные отображения можно рассматривать как *управляемые биективные преобразования (см. разд. 4.2)*, определяемые управляющими двоичными векторами, которые задают различные модификации биективных преобразований.

В качестве другого примера шифра, содержащего управляемое преобразование, ранее приводился пример блочного шифра RC5 [115], для которого в раундовой функции используется операция управляемого циклического сдвига (УЦС) на 0—31 позицию в зависимости от значения пяти управляющих битов.

В настоящее время при конструировании современных блочных шифров, в частности, представленных на конкурс AES, управляемые преобразования по-прежнему находят широкое применение. Так, например, в блочном шифре MARS [49] в раундовой функции используется два управляемых циклических сдвига на 0—31 позицию в зависимости от значения пятизначного битового вектора. Управляемый циклический сдвиг использует еще один кандидат AES — блочный шифр RC6 [114], являющийся улучшенным вариантом алгоритма RC5.

Таким образом, практика создания современных блочных шифров показывает, что использование управляемых преобразований является перспективным направлением проектирования новых криптографических примитивов. За последнее десятилетие в открытой литературе по криптографии появилось достаточно большое количество публикаций, в которых применение управляемых преобразований при конструировании блочных шифров получило серьезное теоретическое обоснование. Однако анализ применяемых на практике управляемых преобразований, в том числе и в приведенных выше примерах, показал, что способы их построения и использования обладают существенными недостатками, среди которых можно выделить следующие:

- небольшие размеры управляемых **S**-блоков позволяют во многих случаях применять методы вскрытия ключевых элементов по частям в пределах действия отдельных **S**-блоков;
- наличие незначительного количества различных модификаций управляемых преобразований создает условия для успешного применения вероятностно-алгебраических методов криптоанализа;
- криптографические свойства существующих управляемых преобразований не оптимальны и не обеспечивают высокой эффективности преобразования данных;
- в ряде случаев достаточно большая математическая сложность управляемых преобразований создает препятствия для высокоскоростной аппаратной и программной реализации блочных алгоритмов.

В связи с перечисленными недостатками управляемых преобразований возникает необходимость дальнейшего совершенствования этих криптографических примитивов на базе принципиально новых способов их реализации. В последующих разделах рассматривается один из нестандартных подходов к решению этой проблемы. Предлагаемый метод связан с созданием специальных криптографически ориентированных управляемых подстановочных преобразований и обеспечивает эффективное их использование в блочных шифрах. Основными направлениями

улучшения криптографических свойств новых примитивов и повышения качества базирующихся на них блочных алгоритмов являются:

- увеличение до 32—128 битов размеров блоков данных, над которыми осуществляются одновременные управляемые преобразования;
- значительное увеличение до 2^n ($n = 32$—128) и более количества возможных модификаций управляемых преобразований за счет совершенствования механизмов управления;
- построение различных моделей управляемых преобразований, обладающих заданными криптографическими свойствами и небольшой сложностью;
- определение новых принципов построения криптосхем на основе управляемых преобразований, обеспечивающих высокую стойкость к современным методам криптоанализа и большое быстродействие при аппаратной и, в ряде случаев, программной реализации.

Предлагаемый метод решения указанных задач связан с дальнейшим развитием общей идеи управляемых преобразований в виде обобщенного понятия *управляемых подстановочных операций* (УПО). Основная особенность новых криптографических примитивов состоит в том, что, сохраняя во многом свойства подстановки или иначе выполняя в определенном смысле функции **S**-блоков, УПО имеют признаки классических математических операций, осуществляющих преобразование одного или двух потоков информации в один выходной поток в зависимости от управляющего вектора. Особенно эффективно применение УПО в сочетании с блоками управляемых перестановок *(см. гл. 3)* и управляемых циклических сдвигов (УЦС), выполняющих функции перестановочных преобразований (**P**-блоков). Такое сочетание криптографических примитивов позволяет качественно изменить свойства подстановочно-перестановочных преобразований в блочных шифрах и в полной мере реализовать принципы эффективного "рассеивания" и "перемешивания" преобразуемых данных.

По своей природе предлагаемые в последующих разделах УПО представляют развитие двух алгебраических операций над двоичными векторами: побитового сложения по модулю 2 (XOR) и сложения по модулю 2^n. Теоретической основой построения новых операций является видоизменение булевых функций, определяющих значения каждого бита на выходе преобразования с целью оптимизации параметров нелинейности, корреляционной иммунности и распространения ошибок всего преобразования в целом. Такие функции, получившие названия *образующих булевых функций*, зависят от одного или двух операндов, управляемых дополнительным вектором, зависящим от преобразуемых данных и ключа.

Следует отметить, что по своему назначению образующие булевы функции во многом похожи на нелинейные функции усложнения преобразований в поточных шифрах (нелинейные фильтры). Однако, несмотря на определенное подобие механизма действия УПО в блочных шифрах процессу поточного шифрования, принципиальное отличие первого от второго заключается в применении обра-

зующих булевых функций для преобразования не отдельных знаков текста, а целых блоков данных. Это обстоятельство определяет необходимость учитывать при создании УПО особые свойства блочных преобразований, такие как обратимость (биективность), общая нелинейность и ряд других криптографических параметров, присущих только блочным шифрам *(подробную информацию см. в разд. 4.3)*. Эти особенности непосредственно влияют на необходимые качества образующих УПО булевых функций.

Прототипом одной из частных моделей УПО являются криптографические преобразования битовых данных на основе так называемых *клеточных автоматов* (cellular automata) [137]. Эти примитивы используют в качестве базовых составляющих самые простейшие образующие булевы функции. Клеточные автоматы ранее широко использовались для моделирования "хаоса" в физических процессах, а с начала 1990-х годов стали эффективно применяться при проектировании различных шифров и хэш-функций. В данной главе идеи клеточных автоматов получили обобщение и дальнейшее развитие в частных разновидностях последовательной модели УПО в виде векторных булевых функций, обладающих рядом необходимых криптографических свойств и невысокой сложностью реализации.

Для УПО разработан целый комплекс моделей преобразований, главными из которых являются последовательная биективная модель (развитие операции XOR) и последовательная рекурсивная модель (развитие операции сложения по модулю 2^n). Термин "управляемость" предлагаемых подстановочных операций фактически связан с псевдослучайным выбором вариантов унарных или бинарных операций, принадлежащих различным алгебраическим структурам, осуществляемым в зависимости от дополнительных параметров (ключа и/или преобразуемых данных). Простейшим примером УПО может служить управляемый сумматор по модулю 2^n, в котором значения битов переноса зависят еще и от битов некоторого двоичного управляющего вектора.

В отличие от табличных подстановок различные типы УПО легко реализуются в виде несложных электронных схем с показателями задержки, не превышающими времени выполнения нескольких операций XOR. Исследования алгебраических и вероятностно-статистических свойств УПО и других управляемых операций, а также способов их применения показали перспективность этих криптографических примитивов для построения высокоскоростных и криптографически стойких блочных шифров. Примерами таких шифров служат блочные алгоритмы SPECTR-H64 (128), COBRA-128, CIKS-128 и ряд других *(подробную информацию см. в гл. 5)*.

В последующих разделах данной главы для конкретизации управляемых подстановочных операций как новых криптографических примитивов рассматриваются следующие вопросы:

- формализуется понятие УПО на основе аппарата булевых функций;
- проводится моделирование и описание классов УПО;

- исследуются криптографические и схемотехнические свойства различных моделей УПО;
- определяются механизмы влияния УПО на стойкость блочных шифров к современным методам криптоанализа;
- оценивается степень сложности реализации УПО различных моделей;
- определяются основные принципы использования УПО в сочетании с другими управляемыми преобразованиями для синтеза эффективных аппаратно и программно ориентированных блочных шифров.

В заключение вступительного раздела необходимо отметить, что предлагаемые новые криптографические примитивы не рассматриваются как идеальные конструкции, являющиеся полной альтернативой другим методам криптографических преобразований. Предлагается всего лишь один из возможных альтернативных способов построения криптографических примитивов, дающий определенный эффект при создании блочных шифров. Авторы не исключают, что ряд теоретических положений относительно предлагаемых моделей преобразований может быть усовершенствован и развит в математическом и прикладном плане. Главной задачей главы является иллюстрация далеко не исчерпанных, как может показаться на первый взгляд, возможностей в разработке новых методов криптографического преобразования информации. Хотелось бы, чтобы для заинтересованного и критически мыслящего читателя излагаемый в главе материал послужил основой дальнейших размышлений в области проектирования и практической реализации блочных шифров нового поколения.

4.2. Математические свойства криптографических преобразований

Криптографические преобразования информации по своей структуре и последовательности реализации в блочных шифрах можно представить в виде совокупности элементарных или иначе базовых криптографических примитивов. За последние десятилетия в современной открытой литературе по теории блочных шифров сформировался определенный математический аппарат, на основе которого проводится анализ свойств различных криптографических примитивов и шифров в целом.

Подстановочные и перестановочные преобразования, обеспечивающие принципы "рассеивания" и "перемешивания" битовых данных и составляющие основу любого блочного шифра, нашли свое отражение в виде строгих математических структур, а их свойства и характеристики описываются на языке математических формул. Ниже приводится частичный обзор результатов исследований, проведенных за последние время в открытых источниках в области изучения матема-

тических свойств криптографических преобразований. Значительная часть определяемых далее терминов и приводимых результатов используется в последующих разделах данной главы при моделировании и оценке свойств новых криптографических примитивов — *управляемых подстановочных операций*.

4.2.1. Представление криптографических преобразований в виде отображений и подстановок

В соответствии с общей теорией [22] каждый базовый криптографический примитив, выполняющий то или иное преобразование на различных этапах блочного шифрования, можно представить в виде отображения $GF(2)^n \to GF(2)^m$ некоторого векторного пространства $GF(2)^n$ n-мерных двоичных векторов $X = (x_1, x_2, \ldots, x_n)$ в другое векторное пространство $GF(2)^m$ m-мерных двоичных векторов $Y = (y_1, y_2, \ldots, y_m)$, где $\forall i \in \{1, \ldots, n\}$ и $\forall j \in \{1, ..., m\}$, $x_i \in GF(2)$, $y_j \in GF(2)$. Данное отображение может определяться в функциональном виде $Y = F(X)$, устанавливающем соответствие $X \to Y$, где $X \in GF(2)^n$, $Y \in GF(2)^m$.

Среди криптографических преобразований часто встречаются отображения вида $GF(2)^{kn} \to GF(2)^n$, которые в ряде случаев удобно записывать с использованием прямого произведения k одинаковых множеств: $GF(2)^n \times \ldots \times GF(2)^n \to GF(2)^n$.

При анализе криптографических преобразований необходимо различать отображения трех типов в соответствии со следующими определениями.

Определение 4.1

Отображение $Y = F(X)$: $GF(2)^n \to GF(2)^m$ называется *вложением* или *инъективным отображением (инъекцией)*, если для любых $X_1, X_2 \in GF(2)^n$ равенство $F(X_1) = F(X_2)$ влечет за собой равенство $X_1 = X_2$.

Определение 4.2

Отображение $Y = F(X)$: $GF(2)^n \to GF(2)^m$ $(n \geq m)$ называется *наложением* или *сюръективным отображением (сюръекцией)*, если для всякого $Y \in GF(2)^m$ найдется такой элемент $X \in GF(2)^n$, что $F(X) = Y$.

Определение 4.3

Отображение, являющееся одновременно инъекцией и сюръекцией, называется *взаимно однозначным* или *биективным отображением (биекцией)*.

Легко показать, что всего существует 2^{m2^n} различных сюръективных отображений $GF(2)^n \to GF(2)^m$. Каждая биекция $GF(2)^n \to GF(2)^n$ является *подстановкой*, справедливо и обратное — любая подстановка на множестве $GF(2)^n$ образует биективное отображение. Всего имеется $2^n!$ различных подстановок или биекций.

4.2.2. Булевы функции в криптографических преобразованиях

Эффективным инструментом анализа криптографических преобразований служит аппарат булевых функций, в соответствии с которым каждое отображение, в том числе и подстановку $GF(2)^n \to GF(2)^m$, можно представить в виде *векторной булевой функции*, компонентами которой являются обычные *булевы функции* с областью определения $GF(2)^n$ и областью значения GF(2):

$$Y = F(X) = \{y_1, y_2, \ldots, y_m\} = \{f_1(X), f_2(X), \ldots, f_m(X)\},$$

где $\forall i \in \{1, ..., m\}$, $y_i = f_i(X)$, $X \in GF(2)^n$ и $y_i \in GF(2)$. Функции $f_i(X)$ будем называть *образующими булевыми функциями* преобразования $Y = F(X)$.

В некоторых случаях для записи аргументов и значений булевых функций используется взаимно однозначное соответствие между векторами из $GF(2)^n$, расположенными в лексикографическом порядке, и множеством целых чисел $\{0, \ldots, 2^n-1\}$. Указанное соответствие для удобства можно определить вектором $\boldsymbol{\beta_i} \in GF(2)^n$, чье целочисленное представление равно $i \in \{0, ..., 2^n-1\}$, т. е. $i = \beta_1^{(i)} 2^0 + \beta_2^{(i)} 2^1 + ... + \beta_n^{(i)} 2^{n-1}$.

Для описания свойств криптографических преобразований, а также определения их стойкости к известным методам криптоанализа большое значение имеют представленные далее характеристики образующих булевых функций.

Определение 4.4

Последовательность значений булевой функции $f(X)$, состоящая из элементов $\{0, 1\}$ и имеющая вид $\mathbf{S}(f) = \{f(\boldsymbol{\beta}_0), f(\boldsymbol{\beta}_1), ..., f(\boldsymbol{\beta}_{2^n-1})\}$, где $\boldsymbol{\beta}_j$ приобретают все различные значения из $GF(2)^n$, называется *таблицей истинности* булевой функции $f(X)$ [122].

В криптографических преобразованиях важными и определяемыми далее факторами являются сбалансированность образующих булевых функций и регулярность всего преобразования в целом. Указанные свойства позволяют избежать прямых статистических атак на элементы блочного шифра и во многих случаях должны рассматриваться как необходимые условия при проектировании базовых криптографических примитивов блочных алгоритмов.

Относительно понятия сбалансированности и регулярности имеют место следующие определения и утверждения [122, 125].

Определение 4.5

Булева функция $f(X)$ ($X \in GF(2)^n$) называется *сбалансированной*, если ее таблица истинности содержит одинаковое количество "0" и "1"

$$\#\{X \mid f(X) = 0\} = \#\{X \mid f(X) = 1\} = 2^{n-1}.$$

Утверждение 4.1

Если одна из булевых функций $f_1(X)$ или $f_2(W)$ ($X \in \mathrm{GF}(2)^{n_1}$, $W \in \mathrm{GF}(2)^{n_2}$) является сбалансированной булевой функцией, то функция $f(X, W) = f_1(X) \oplus f_2(W)$ также будет сбалансированной булевой функцией.

Определение 4.6

Отображение $Y = F(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$) при $n \geq m$ называется *регулярным*, если Y ровно 2^{n-m} раза принимает все 2^m различных значений из $\mathrm{GF}(2)^m$, в то время как X проходит все 2^n значений из $\mathrm{GF}(2)^n$.

Очевидно, что каждое регулярное отображение $Y = F(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$) при $m = n$ является биективным отображением.

Утверждение 4.2

Отображение $Y = F(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$) при $n \geq m$ регулярно тогда и только тогда, когда любая линейная комбинация элементов вектора $F(X) = \{f_1(X), f_2(X), \ldots, f_m(X)\}$ является сбалансированной булевой функцией, т. е. $\forall \boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m) \in \mathrm{GF}(2)^m$ сумма $\sum_{i=1}^{m} \alpha_i f_i(X)$ — сбалансированная булева функция.

Иногда для удобства описания свойств криптографических примитивов вместо произвольной булевой функции $f(X)$ рассматривается *сопряженная* с ней функция $f^*(X) = 1 - 2f(X) = (-1)^{f(X)}$, принимающая значения не на множестве $\{0, 1\}$, а на множестве двух действительных чисел $\{1, -1\}$. Аналогом таблицы истинности для этой функции является *характеристическая последовательность*, формируемая в соответствии со следующим определением:

Определение 4.7

Последовательность, состоящая из элементов $\{1, -1\}$ и имеющая вид:

$$\mathbf{S}^*(f) = \{(-1)^{f(\beta_0)}, (-1)^{f(\beta_1)}, \ldots, (-1)^{f(\beta_{2^n-1})}\},$$

где β_j принимают все различные значения из $\mathrm{GF}(2)^n$, называется *характеристической последовательностью* действительной функции $f^*(X) = (-1)^{f(X)}$.

Булевы функции могут быть записаны по-разному. Из многообразия возможных типов таких записей в дальнейшем будут рассматриваться только *алгебраические нормальные формы* (АНФ) или, иначе, *полиномы Жегалкина*, имеющие следующий вид: $f_{АНФ}(X) = \gamma_0 \oplus \bigoplus_{1 \leq i \leq n} \gamma_i x_i \oplus \bigoplus_{1 \leq i \leq j \leq n} \gamma_{ij} x_i x_j \oplus \gamma_{12\ldots n} x_1 x_2 \ldots x_n$, где $X \in \mathrm{GF}(2)^n$ и все $\gamma \in \mathrm{GF}(2)$ (в указанной записи используются только булевы операции побитового умножения "·" ("∧") и побитового сложения "⊕" ("XOR")).

Степень максимального (крайне правого) элемента АНФ называется *алгебраической степенью* булевой функции $f(X)$ и обозначается как $\mathbf{deg}(f)$.

Существует эффективный метод получения АНФ произвольной булевой функции $f(X)$ на основе специальной матрицы линейного преобразования $\mathbf{A}_n$ и таблицы истинности этой функции [112].

Если расположить в лексикографическом порядке все возможные 2^n коэффициента из $f_{АНФ}(X)$ в виде последовательности $\gamma(f) = \{1, \gamma_1, \gamma_2, \gamma_1\gamma_2, \gamma_3, \gamma_1\gamma_3, \ldots, \gamma_1\gamma_2\gamma_3\ldots\gamma_n\}$ и разместить ее в столбец $[\gamma(f)]$, а также расположить в столбец 2^n значений таблицы истинности $[\mathbf{S}(f)]$, то коэффициенты АНФ можно получить из соотношения:

$[\gamma(f)] = \mathbf{A}_n[\mathbf{S}(f)]$, где $\mathbf{A}_n$ формируется следующей итерацией:

$$\mathbf{A}_n = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \otimes \mathbf{A}_{n-1} = \begin{bmatrix} \mathbf{A}_{n-1} & 0 \\ \mathbf{A}_{n-1} & \mathbf{A}_{n-1} \end{bmatrix}, \ \mathbf{A} = [1]$$

и "$\otimes$" обозначает тензорное (Кронекеровское) произведение матриц, при этом само преобразование является *инволюцией*, т. е. $\mathbf{A}_n^2 = \mathbf{I}_n$.

4.2.3. Роль дискретного преобразования Фурье в описании свойств криптографических преобразований

В вопросах дифференциального [44], линейного [86] и других видов криптоанализа [89] шифров фундаментальную роль играет понятие *преобразования Уолша-Адамара* [112], являющееся разновидностью *дискретного преобразования Фурье*. Для определения этого преобразования введем некоторые новые понятия.

Определение 4.8

Аффинными функциями $\{f_{aff}(X)\}$ ($X \in \mathrm{GF}(2)^n$) называются булевы функции вида $f_{aff}(X) = \boldsymbol{\alpha}\bullet X \oplus c = \alpha_1 x_1 \oplus \ldots \oplus \alpha_n x_n \oplus c$, где $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_n)$, $\alpha_i, c \in \mathrm{GF}(2)$ и "$\bullet$" — знак скалярного произведения. При $c = 0$ аффинные булевы функции образуют подмножество *линейных функций* $\{f_{lin}(X)\}$, каждая из которых является скалярным произведением вида $f_{lin}(X) = \boldsymbol{\alpha}\bullet X$.

Определение 4.9

Преобразованием Уолша-Адамара (ПУА) действительной функции $f(X) \in \mathrm{GF}(2)$ по вектору $\boldsymbol{\alpha}$ ($X, \boldsymbol{\alpha} \in \mathrm{GF}(2)^n$) называется линейное преобразование, принимающее значение в области действительных чисел и имеющее следующий вид:

$$\mathbf{U}_\alpha(f) = \sum_{X \in GF(2)^n} f(X)(-1)^{\alpha\bullet X}.$$

Для сопряженной функции $f^*(X)$ ПУА приобретает вид:

$$\mathbf{U}_\alpha^*(f) = \sum_{X \in GF(2)^n} f^*(X)(-1)^{\alpha\bullet X} = \sum_{X \in GF(2)^n} (-1)^{f(X)}(-1)^{\alpha\bullet X} = \sum_{X \in GF(2)^n} (-1)^{(f(X)\oplus\alpha\bullet X)}.$$

Следует заметить, что для ПУА функции $f(X)$ элемент $\boldsymbol{\alpha}\bullet X$ рассматривается как функция, принимающая действительные значения 0 и 1, а для ПУА сопряженной функции $f^*(X)$ указанные действительные значения принимает функция $f(X) \oplus \boldsymbol{\alpha}\bullet X$.

Из определений 4.4 и 4.7 следует, что ПУА выражается через скалярное произведение таблиц истинности и характеристических последовательностей соответствующих булевых функций: $\mathbf{U}_\alpha(f) = \mathbf{S}(f)\bullet\mathbf{S}^*(\boldsymbol{\alpha}\bullet X)$ и $\mathbf{U}^*_\alpha(f) = \mathbf{S}^*(f)\bullet\mathbf{S}^*(\boldsymbol{\alpha}\bullet X)$.

Указанные преобразования связаны следующими соотношениями [67]:

$$\mathbf{U}^*_\alpha(f) = -2\mathbf{U}_\alpha(f) + 2^n\delta(\alpha) \text{ и } \mathbf{U}_\alpha(f) = (-1/2)\,\mathbf{U}^*_\alpha(f) + 2^{n-1}\delta(\alpha),$$

где $\delta(\boldsymbol{\alpha}) = \begin{cases} 1 & \text{при } \boldsymbol{\alpha} = (0, \ldots, 0) \\ 0 & \text{при } \boldsymbol{\alpha} \neq (0, \ldots, 0) \end{cases}$.

Из определений 4.5 и 4.9 следует, что значение ПУА в точке $\boldsymbol{\alpha} = \mathbf{0} = (0, 0, \ldots, 0)$ определяет степень отклонения булевой функции $f(X)$ от сбалансированности, при этом само условие сбалансированности выражается следующим образом:

$$\mathbf{U}_0(f) = 2^{n-1},\ \mathbf{U}^*_0(f) = 0.$$

Для *обратного преобразования Уолша-Адамара* справедливы, соответственно, следующие формулы:

$$f(X) = \frac{1}{2^n}\sum_{\alpha\in GF(X)^n}\mathbf{U}_\alpha(f)(-1)^{\alpha\bullet X} \text{ и } f^*(X) = \frac{1}{2^n}\sum_{\alpha\in GF(X)^n}\mathbf{U}^*_\alpha(f)(-1)^{\alpha\bullet X}.$$

Образующими элементами для линейных преобразований, осуществляемых ПУА, являются *матрицы Уолша-Адамара*, которые составляют специальный класс *матриц Адамара*, формируемый в соответствии со следующими ниже правилами:

Определение 4.10

Матрица $\boldsymbol{H}$ с элементами из множества $\{-1, 1\}$ называется *матрицей Адамара*, если $\mathbf{H}\mathbf{H}^{\mathrm{T}} = n\mathbf{I}_n$, где $\mathbf{H}^{\mathrm{T}}$ — транспонированная матрица $\mathbf{H}$, $\mathbf{I}_n$ — единичная матрица.

Если 2^n значений характеристической последовательности $\mathbf{S}^*(f)$ функции $f^*(X)$ и 2^n значений действительной функции $\mathbf{U}^*_\alpha(f)$ записать в виде столбцов $[\mathbf{S}^*(f)]$ и $[\mathbf{U}^*_\alpha(f)]$, то ПУА задается матрицей Уолша-Адамара $\mathbf{H}_n$ как $[\mathbf{U}^*_\alpha(f)] = \mathbf{H}_n[\mathbf{S}^*(f)]$, где $\mathbf{H}_n$ формируется следующей итерацией:

$$\mathbf{H}_n = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \otimes \mathbf{H}_{n\text{-}1} = \begin{bmatrix} \mathbf{H}_{n-1} & \mathbf{H}_{n-1} \\ \mathbf{H}_{n-1} & -\mathbf{H}_{n-1} \end{bmatrix},\ \mathbf{H}_0 = [1].$$

Основными свойствами матриц Уолша-Адамара являются:

- $\forall s, t \mid s + t = n\ \mathbf{H}_n = \mathbf{H}_s \otimes \mathbf{H}_t$;
- $\mathbf{H}_n^2 = 2^n\mathbf{I}_n$;

- каждая i-я строка матрицы $\mathbf{H}_n$ — суть характеристическая последовательность $\mathbf{S}^*(\alpha_i \bullet X)$ линейной функции $\alpha_i \bullet X$, где целочисленное представление вектора $\alpha_i \in GF(2)^n$ равно i. С другой стороны, каждая характеристическая последовательность $\mathbf{S}^*(f_{lin})$ линейной функции от n переменных — суть строка матрицы $\mathbf{H}_n$.

Аппарат ПУА служит фундаментальным средством оценки криптографических свойств преобразований, непосредственно влияющих на качество блочных шифров. Как отмечалось ранее, среди таких свойств особое место занимают *степень нелинейности*, *уровень корреляционной иммунности* и *степень распространения изменений (ошибок)*, которые рассматриваются в последующих разделах.

4.2.4. Характеристики нелинейности булевых функций

При оценке эффективности криптографических примитивов с точки зрения их стойкости к линейному криптоанализу в качестве одной из основных характеристик нелинейности функций можно рассматривать удаленность образующих булевых функций от множества аффинных или линейных функций. Для детализации понятия нелинейности булевых функций приведем следующие определения и утверждения [112].

Определение 4.11

Расстоянием Хэмминга $\mathbf{DH}(f, g)$ между двумя булевыми функциями $f(X)$ и $g(X)$ называется количество значений, по которым эти функции отличаются, что соответствует условию: $\mathbf{DH}(f, g) = \mathbf{WH}(\mathbf{S}(f) \oplus \mathbf{S}(g))$.

Утверждение 4.3

$$\mathbf{DH}(f,g) = 2^{n-1} - \frac{1}{2} \sum_{X \in GF(X)^n} f^*(X) g^*(X) = 2^{n-1} - \frac{1}{2} \mathbf{S}^*(f) \bullet \mathbf{S}^*(g).$$

Определение 4.12

Корреляцией $\mathbf{C}(f, g)$ между булевыми функциями $f(X)$ и $g(X)$ называется значение выражения:

$$\mathbf{C}(f,g) = \frac{1}{2^n} \sum_{X \in GF(X)^n} f^*(X) g^*(X) = \frac{1}{2^n} \mathbf{S}^*(f) \bullet \mathbf{S}^*(g) =$$

$$= \frac{1}{2^n} \sum_{\substack{X \in GF(X)^n \\ f(X) = g(X)}} f^*(X) g^*(X) + \frac{1}{2^n} \sum_{\substack{X \in GF(X)^n \\ f(X) \neq g(X)}} f^*(X) g^*(X).$$

Из определения 4.12 следует, что $\mathbf{C}(f, g) = \mathbf{Pr}_X[f(X) = g(X)] - \mathbf{Pr}_X[f(X) \neq g(X)]$, и между функциями корреляции и расстоянием Хэмминга имеет место соотношение:

$$\mathbf{C}(f, g) = (1/2^n)[2^n - \mathbf{DH}(f, g)] - (1/2^n)[2^n - \mathbf{DH}(f, \overline{g})] =$$
$$= (1/2^n)[\mathbf{DH}(f, \overline{g}) - \mathbf{DH}(f, g)] = 1 - (1/2^{n-1})\mathbf{DH}(f, g).$$

Утверждение 4.4

Для произвольной булевой функции $f(X)$ справедливо *равенство Парсеваля* [59]:

$$\sum_{\forall f_{lin}} \mathbf{C}(f, f_{lin})^2 = 1 \cdot$$

Из равенства Парсеваля вытекает принципиально важное следствие:

Следствие 4.1

Для любой булевой функции $f(X)$ всегда существует корреляция с какой-то линейной булевой функцией, т. е. $\exists f_{lin} \mid \mathbf{C}(f, f_{lin}) \neq 0$.

Определение 4.13

Нелинейностью булевой функции $f(X)$ $(\mathrm{GF}(2)^n \to \mathrm{GF}(2))$ называется значение следующего выражения [81, 106]: $\mathbf{N}(f) = \min\limits_{\forall f_{aff}} \mathbf{DH}(f, f_{aff})$ ·

В терминах линейных булевых функций значение $\mathbf{N}(f)$ выражается следующим образом:

$$\mathbf{N}(f) = \min_{\forall f_{lin}} \min\{\mathbf{DH}(f, f_{lin}), 2^n - \mathbf{DH}(f, f_{lin})\} =$$
$$= 2^{n-1} - \frac{1}{2}\max_{\forall f_{lin}} \left|\mathbf{DH}(f, f_{lin}) - \mathbf{DH}(f, \overline{f_{lin}})\right| = 2^{n-1} - 2^{n-1}\max_{\forall f_{lin}} \left|\mathbf{C}(f, f_{lin})\right|,$$

где

$$\mathbf{C}(f, f_{lin}) = \mathbf{Pr}_X[f(X) = f_{lin}(X)] - \mathbf{Pr}_X[f(X) \neq f_{lin}(X)] = 2\mathbf{Pr}_X[(f(X) = \alpha \bullet X)] - 1 =$$
$$= \frac{1}{2^n}\mathbf{S}^*(f) \bullet \mathbf{S}^*(\alpha \bullet X) = \frac{1}{2^n}\mathbf{U}^*_\alpha(f).$$

Функция $\mathbf{C}(f, f_{lin})$ играет определяющую роль в вопросах линейного криптоанализа, где используется в качестве *линейной характеристики* булевых функций $f(X)$ по линейной булевой функции $f_{lin}(X) = \alpha \bullet X$.

Утверждение 4.5

В соответствии с утверждением 4.3 значение нелинейности булевой функции $f(X)$ можно определить из следующего равенства:

$$\mathbf{N}(f) = 2^{n-1} - \frac{1}{2}\max_{\forall f_{lin}} \left|\mathbf{S}^*(f) \bullet \mathbf{S}^*(f_{lin})\right| = 2^{n-1} - \frac{1}{2}\max_{\forall \alpha \in GF(2)^n} \left|\mathbf{U}^*_\alpha(f)\right| \cdot$$

Данное соотношение является достаточно эффективным инструментом определения нелинейности булевых функций при малых n.

Утверждение 4.6

Если $f(X, Y) = f_1(X) \oplus f_2(Y)$ ($X \in \mathrm{GF}(2)^{n_1}$, $Y \in \mathrm{GF}(2)^{n_2}$) и d_1, d_2 являются значениями нелинейности, соответственно, функций $f_1(X)$ и $f_2(Y)$, то $\mathbf{N}(f(X, Y)) \geq d_1 2^{n_2} + d_2 2^{n_1} - 2d_1 d_2$. Дополнительно имеют место неравенства [125]: $\mathbf{N}(f(X, Y)) \geq d_1 2^{n_2}$ и $\mathbf{N}(f(X, Y)) \geq d_2 2^{n_1}$.

Определение 4.14

Линейностью булевой функции $f(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)$) называется значение следующего выражения [106]: $\mathbf{L}(f) = \max\limits_{\forall f_{lin}} \left|\mathbf{C}(f, f_{lin})\right|$.

Из определений 4.13 и 4.14 следует, что линейность и нелинейность булевой функции $f(X)$ связаны следующими соотношениями: $\mathbf{N}(f) = 2^{n-1}(1 - \mathbf{L}(f))$ и $\mathbf{L}(f) = 1 - 2^{1-n}\mathbf{N}(f)$.

Из равенства Парсеваля следует, что $2^{-n/2} \leq \mathbf{L}(f) \leq 1$ и $0 \leq \mathbf{N}(f) \leq 2^{n-1} - 2^{(n/2)-1}$.

Определение 4.15

Общей нелинейностью и, соответственно, *общей линейностью векторной булевой функции* $F(X)$ называются значения следующих выражений [106]:

$$\mathbf{N}(F) = \min_{\forall b \in GF(2)^m,\ b \neq 0} \mathbf{N}(b \bullet F); \quad \mathbf{L}(F) = \max_{\forall b \in GF(2)^m,\ b \neq 0} \mathbf{L}(b \bullet F).$$

Из определений 4.14 и 4.15 следует, что $2^{-n/2} \leq \mathbf{L}(F) \leq 1$ и $0 \leq \mathbf{N}(F) \leq 2^{n-1} - 2^{(n/2)-1}$.

Определение 4.16

Средней нелинейностью (линейностью) векторной булевой функции $F(\mathrm{X})$ называются, соответственно, значения следующих выражений:

$$\mathbf{N}'(F) = \frac{1}{2^m - 1} \sum_{\forall b \in GF(2)^m,\ b \neq 0} \mathbf{N}(b \bullet F); \quad \mathbf{L}'(F) = \frac{1}{2^m - 1} \sum_{\forall b \in GF(2)^m,\ b \neq 0} \mathbf{L}(b \bullet F).$$

В блочных шифрах одним из основных способов обеспечения нелинейности преобразований является применение **S**-блоков, которые в общем случае можно определить как некоторые нелинейные отображения вида $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$. Поскольку при больших размерах блоков исходных данных (n) достаточно сложно выполнить такое преобразование целиком, то данные разбиваются на части и далее преобразуются по серии **S**-блоков меньшего размера (s). Такое комбинированное преобразование обозначается в виде $\mathbf{S}_{k \times s}$: $\mathrm{GF}(2)^s \times \ldots \times \mathrm{GF}(2)^s \to \mathrm{GF}(2)^m$, где k — количество **S**-блоков и $n = k \times s$ — размер всего преобразования.

4.2.5. Автокорреляция булевых функций

Важную роль при построении криптографических примитивов, обеспечивающих высокую стойкость к дифференциальному криптоанализу, играет функция автокорреляции, относительно которой в соответствии с [59, 112] имеется ряд следующих определений и утверждений.

Определение 4.17

Автокорреляцией булевой функции $f(X)$ относительно вектора $\boldsymbol{\alpha} \in \mathrm{GF}(2)^n$ называется функция вида:

$$\mathbf{r}_\alpha(f) = \frac{1}{2^n} \sum_{\forall X \in GF(2)^n} f^*(X) f^*(X \oplus \alpha) = \mathbf{S}^*(f) \bullet \mathbf{S}^*(f(X \oplus \alpha)) \cdot$$

Определение 4.18

Автокорреляцией векторной булевой функции $F(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$) относительно вектора $\boldsymbol{\alpha} \in \mathrm{GF}(2)^n$ по вектору $\boldsymbol{\omega} \in \mathrm{GF}(2)^m$ называется функция следующего вида:

$$\mathbf{R}_{\alpha,\omega}(F) = \frac{1}{2^n} \sum_{\forall X \in GF(2)^n} (-1)^{\omega \bullet F(X) \oplus \omega \bullet F(X \oplus \alpha)} .$$

Таким образом, автокорреляцию векторной булевой функции $F(X)$ можно охарактеризовать матрицей автокорреляции $\mathbf{R} = \| \mathbf{R}_{i,j}(F) \|$, где i и j — целочисленные значения упорядоченных в лексикографическом порядке двоичных векторов $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{2^n}$.

Утверждение 4.7

Для булевой функции $f(X)$ ($X \in \mathrm{GF}(2)^n$) имеет место соотношение [112]:

$$\forall \boldsymbol{\omega} \in \mathrm{GF}(2)^n \ \mathbf{U}^*_\omega(\mathbf{r}_\alpha(f)) \sum_{\alpha \in GF(2)^n} \mathbf{r}_\alpha(f)(-1)^{\alpha \bullet \omega} = \frac{1}{2^n}(\mathbf{U}^*_\omega(f))^2 ,$$

при этом функция $(\mathbf{U}^*_\omega(f))^2$ называется *энергетическим спектром булевой функции* $f(X)$.

Если использовать ОПУА *(см. разд. 4.2.3)*, то из утверждения 4.7 следует, что

$$\forall \boldsymbol{\alpha} \in \mathrm{GF}(2)^n \ \mathbf{r}_\alpha(f) = \frac{1}{2^{2n}} \sum_{\alpha \in GF(2)^n} \mathbf{U}^*_\omega(f)(-1)^{\alpha \bullet \omega} ,$$

т. е. автокорреляция пропорциональна ПУА энергетического спектра булевой функции $f(X)$.

Утверждение 4.8

Для булевой функции $f(X)$ ($X \in \mathrm{GF}(2)^n$) справедливы следующие соотношения [112]:

$$\mathbf{Pr}_X[f^*(X) \neq f^*(X \oplus \boldsymbol{\alpha})] = \frac{1}{2} - \frac{1}{2}\mathbf{r}_{\boldsymbol{\alpha}}(f);$$

$$\sum_{\alpha \in GF(2)^n, \alpha \neq 0} \mathbf{Pr}_X[f^*(X) \neq f^*(X \oplus \boldsymbol{\alpha})] = 2^{n-1} - \frac{(U_0^*(f))^2}{2^{n+1}}.$$

Если рассматривать вектор $\boldsymbol{\alpha}$ как некоторое изменение или ошибку в исходном векторе X, то первая из указанных в утверждении 4.8 вероятностей определяет, каким образом такая ошибка статистически влияет на значение булевой функции, т. е. как статистически распространяется ошибка $\boldsymbol{\alpha}$ в криптографическом преобразовании $f(X)$. Чем ближе данная вероятность к значению 1/2, тем меньше информации имеет злоумышленник при попытке определить исходное значение X по известному значению Y. Отсюда следует, что, чем ближе автокорреляция в точке $\boldsymbol{\alpha}$ к нулю, тем эффективнее распространяется ошибка $\boldsymbol{\alpha}$.

Второе выражение утверждения 4.8 определяет общую оценку распространения ошибки по всем значениям $\boldsymbol{\alpha}$. При этом качество распространения ошибки тем лучше, чем ближе к нулю значение энергетического спектра булевой функции $f(X)$ в точке ноль.

Утверждение 4.9

Для векторной булевой функции $F(X)$ ($\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$) справедливо следующее равенство [59]:

$$\mathbf{Pr}_X(F(X) \oplus F(X \oplus \boldsymbol{\alpha}) = \boldsymbol{\beta}) = \frac{1}{2^m} \sum_{\omega \in GF(2)^m} (-1)^{\omega \bullet \beta} \mathbf{R}_{\alpha,\omega}(F).$$

Данное соотношение является фундаментальным в вопросах математической интерпретации дифференциального криптоанализа. Указанная выше вероятность аналитически определяет степень распространения изменения или ошибки в векторной булевой функции $F(X)$.

4.2.6. Критерий строгого лавинного эффекта и критерий распространения изменений

Важными характеристиками качества криптографических примитивов, уточняющими приводимые в *разд. 4.2.5* автокорреляционные свойства преобразований, являются понятия *критерия строгого лавинного эффекта* (Strict Avalanche Criterion, **SAC**) и *критерия распространения изменений* (Propagation Criterion, **PC**) [59, 67, 81, 85, 112]. Основным содержанием указанных критериев является

оценка вероятности изменения значения булевых функций в зависимости от изменения определенной части аргументов этих функций.

Пусть $\Delta f(X,\alpha) = f(X) \oplus f(X \oplus \alpha)$, где X, $\alpha \in \mathrm{GF}(2)^n$ и $f(X) \in \mathrm{GF}(2)$, тогда для булевой функции $f(X)$ можно сформулировать следующее определение [81]:

Определение 4.19

Булева функция $f(X)$ удовлетворяет:

- **SAC**, если $\Delta f(X, \alpha)$ — сбалансированная булева функция $\forall \alpha \mid \mathbf{WH}(\alpha) = 1$;
- **SAC** порядка k (**SAC**(k)), если любая частная функция, полученная из $f(X)$ фиксацией любых ее k переменных, удовлетворяет **SAC**;
- **PC** степени $l \le n$ (**PC**(l)), если $\Delta f(X, \alpha)$ — сбалансированная булева функция $\forall \alpha \mid 1 \le \mathbf{WH}(\alpha) \le l$;
- **PC** степени $l \le n$ и порядка $k \le n - 1$ (**PC**(l, k), если любая частная функция, полученная из $f(X)$ фиксацией любых ее k переменных, удовлетворяет **PC**(l).

Утверждение 4.10

Если булева функция $f(X)$ удовлетворяет **SAC**(k) при $0 \le k < n - 2$, то $\mathbf{deg}(f) \le n - k - 1$.

В терминах автокорреляции булевой функции (см. определение 4.17) базовые понятия указанных выше критериев приобретают следующий вид:

Утверждение 4.11

Булева функция $f(X)$ удовлетворяет критериям:

- **SAC**, если $\mathbf{r}_\alpha(f) = 0 \ \forall \alpha \mid \mathbf{WH}(\alpha) = 1$;
- **PC**(l), если $\mathbf{r}_\alpha(f) = 0 \ \forall \alpha \mid 1 \le \mathbf{WH}(\alpha) \le l$.

Понятие *"распространение изменений"* может определяться не только множеством векторов определенного веса Хэмминга, но и произвольным множеством бинарных векторов.

Определение 4.20

Булева функция $f(X)$ удовлетворяет критерию распространения изменений на множестве бинарных векторов $\{\alpha\} \subset \mathrm{GF}(2)^n$ (**PC**($\{\alpha\}$)), если $\forall \alpha \in \{\alpha\}$ $\mathbf{r}_\alpha(f) = 0$.

Впервые идеи **SAC** были сформулированы Вебстером и Таваресом [134]. В дальнейшем эта концепция была развита Форре, который сформулировал понятие **SAC**(k) [67]. Обобщение оценок нелинейности булевых функций и **SAC** было проведено в работах Пренила и др., где, в частности, были определены и развиты понятия **PC**(l) и **PC**(l, k) [112]. Методы построения булевых функций, удовлетворяющих критериям **SAC**(k) и **PC**(l, k), изложены в работах [67, 81, 85] и в целом ряде других публикаций.

Пример 4.1. Булева функция (квадратичная форма) $f(X) = \sum_{1<i<j<n} x_i x_j$ — удовлетворяет следующим критериям:

- **SAC**(n–2);
- **PC**(l, k) при $l + k \le n - 1$ или ($l + k = 1$ и l — четное);
- **PC**(2, n–2) (это единственная булева функция, удовлетворяющая этому критерию);
- **PC**(n) при четном n , т. е. $f(X)$ — бент-функция *(см. разд. 4.2.8)*;
- **PC**(n–1) при нечетном n и $f(X)$ будет сбалансированной булевой функцией.

В частности, квадратичная форма вида:

$$f(X) = x_1x_2 \oplus x_1x_3 \oplus x_1x_4 \oplus x_1x_5 \oplus x_1x_6 \oplus x_2x_3 \oplus x_2x_4 \oplus x_2x_5 \oplus x_2x_6 \oplus x_3x_4 \oplus x_3x_5 \oplus x_3x_6 \oplus x_4x_5 \oplus x_4x_6 \oplus x_5x_6$$

удовлетворяет следующим критериям: **PC**(6), **PC**(4, 2), **PC**(2, 4) и имеет следующие таблицу истинности, спектр ПУА и автокорреляцию:

$$\mathbf{S}(f) = 0001011101111110011111101110100001111110111010001110100010000001;$$

$$\mathbf{U}^*_\alpha(f) = -8,-8,-8,8,-8,8,8,8,-8,8,8,8,8,8,8,-8,-8,8,8,8,8,8,8,-8,8,8,8,-8,8,-8,-8,-8,-8,$$
$$-8,8,8,8,8,8,8,-8,8,8,8,-8,8,-8,-8,-8,8,8,8,-8,8,-8,-8,-8,8,-8,-8,-8,-8,-8,-8,8,8;$$

$$\mathbf{r}_\alpha(f) = 0000000000000000000000000000000000000000000000000000000000000000.$$

Важной качественной характеристикой криптографических примитивов является *лавинный эффект*, который определяет, каким образом изменяется значение выхода преобразования в зависимости от изменения некоторого количества битов данных или ключа на входе преобразования. Вектора, определяющие эти изменения, называются *лавинными векторами* и имеют вид $\Delta F(X, \alpha) = F(X) \oplus F(X \oplus \alpha)$. Характеристики векторного лавинного эффекта теоретически оцениваются в соответствии со значениями векторной автокорреляции из результатов утверждения 4.9.

4.2.7. Корреляционная иммунность и эластичность булевых функций

Естественным усилением свойства сбалансированности булевой функции $f(X)$ является, в частности, требование сбалансированности всех функций, полученных из исходной функции фиксированием некоторых ее переменных. Указанное требование способствует стойкости криптографических примитивов к прямым статистическим атакам при фиксированных значениях нескольких битов на входе преобразования. Данное свойство, а также ряд других криптографически важных качеств булевых функций связаны с понятиями их корреляционной иммун-

ности и эластичности, формулируемыми в соответствии со следующими положениями [50, 57, 131]:

Определение 4.21

Булева функция $f(X)$ ($X \in \mathrm{GF}(2)^n$) называется *корреляционно-иммунной степени l* (Correlation Immunity of degree l, $\mathbf{CI}(l)$), если $\forall \alpha \mid 1 \le \mathbf{WH}(\alpha) \le l < n \quad \mathbf{U}^*_\alpha(f) = 0$.

Впервые понятие корреляционной иммунности функций было введено Зигенталером [127] и развито Ксяо и Мэсси [135], а также Камеоном и Карле [50]. В частности, показано, что $\forall f(X)$, удовлетворяющей $\mathbf{CI}(l)$, всегда выполняется неравенство $l + \mathbf{deg}(f) \le n$.

Наличие корреляционной иммунности степени l означает, что значения функция $Y = f(X)$ статистически независимы от любого набора из не более чем l компонентов произвольного вектора-аргумента $X \in \mathrm{GF}(2)^n$. Это равносильно условию, что на выход преобразования не "просачивается" информация о векторах, поступающих на вход преобразования и имеющих вес Хэмминга не более l. Примечательным является тот факт, что в соответствии со следствием 4.1, если некоторая функция имеет корреляционную иммунность степени l, то она непременно будет иметь корреляцию с некоторой линейной булевой функцией $\alpha \bullet X$, где вектор $\mathbf{WH}(\alpha) > l$, т. е. $\exists \alpha \mid \mathbf{WH}(\alpha) > l \quad \mathbf{U}^*_\alpha(f) \ne 0$. Отсюда справедливо следствие:

Следствие 4.2

Максимальная степень корреляционной иммунности булевой функции $Y = f(X)$ ($X \in \mathrm{GF}(2)^n$) не превышает n–1.

В отдельных случаях, когда для булевой функции $f(X)$ не удается добиться полной корреляционной иммунности степени l, указанное понятие может рассматриваться в ослабленном виде относительно не всего исходного вектора X, а лишь некоторой его части X^* в соответствии со следующим определением:

Определение 4.22

Булева функция $f(X)$ ($X \in \mathrm{GF}(2)^n$) называется *корреляционно-иммунной степени l относительно множества* X^* (Correlation Immunity of degree l with set X^*, $\mathbf{CI}(l, X^*)$), образованного из части компонентов вектора X, если $\forall \alpha = \alpha_{X \setminus X^*} \cup \alpha_{X^*}$ и $1 \le \mathbf{WH}(\alpha_{X^*}) \le l < n \Rightarrow \mathbf{U}^*_\alpha(f) = 0$, где α_{X^*} — компоненты вектора α, соответствующие множеству X^*, а $\alpha_{X \setminus X^*} = 0$ — остальные компоненты вектора α.

Поскольку сбалансированность булевых функций — одно из важных свойств криптографических преобразований, то дальнейшее развитие понятия корреляционной иммунности задается следующим определением:

Определение 4.23

Булева функция $f(X)$ называется *эластичной степени l и порядка k* (Resilient function) ($\boldsymbol{R}(l, k)$), если любая частная функция, полученная из $f(X)$ фиксированием

k переменных является корреляционно-иммунной степени l и сбалансированной функцией.

Таким образом, сбалансированные корреляционно-иммунные БФ являются частным случаем эластичных булевых функций.

Пример 4.2. Булева функция от шести переменных вида:

$$f(X) = x_1 \oplus x_2 \oplus x_3 \oplus x_2x_5 \oplus x_3x_6 \oplus x_4x_5 \oplus x_4x_6 \oplus x_3x_5x_6 \oplus x_4x_5x_6 \oplus x_2x_5x_6$$

удовлетворяет критерию $\boldsymbol{R}(2, 0)$ и имеет нелинейность $\mathbf{N}(f) = 24$ при максимально возможном значении нелинейности 26 для сбалансированных булевых функций от шести переменных. Таблица истинности, спектр ПУА и значения автокорреляции этой булевой функции имеют следующий вид:

$$\mathbf{S}(f) = 0000011110111100110110100110000111111000010000110010010110011110;$$

$$\mathbf{U}^*_\alpha(f) = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 16, 16, –16, –16, 0, 0, 0, 0, 16, –16, 16, –16, 16, 16, 16, 16, 16, –16, –16, 16;$$

$$\mathbf{r}_\alpha(f) = 00000000000000000000000000000000–10000000000000000000000000000000.$$

Определение 4.24

Векторная булева функция $Y = F(X)$ $(\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m)$ называется (n, m, k)-*эластичной* ($\boldsymbol{R}(n, m, k)$), если любая частная функция, полученная из $F(X)$ фиксированием k переменных, является *регулярной*, т. е. ее значения пробегают все возможные m-граммы из $\mathrm{GF}(2)^m$ равное количество раз в том случае, когда остальные $n–k$ переменных функции пробегают все 2^{n-k} значения из $\mathrm{GF}(2)^{n-k}$ по одному разу.

Из определения 4.6 следует, что свойство регулярности векторных булевых функций является частным случаем свойства эластичности (регулярные — это $(n, m, 0)$-эластичные булевы функции).

Понятие эластичных булевых функций была введено Голдрихом в соавторстве с другими математиками [57]. В дальнейшем теория таких функций была развита в работах Себерри, Чжань и Чжень [123], Саркара и Маитра [119], Таранникова [131] и др.

4.2.8. Роль бент-функций в криптографических преобразованиях

Среди булевых функций особое место занимают функции, обладающие предельно высокими показателями нелинейности и распространения ошибок ($\mathbf{PC}(n)$) и тем самым способствующие повышению стойкости к методам линейного и дифференциального криптоанализа блочных шифров.

Определение 4.25

Булева функция $f(X)$ ($X \in \mathrm{GF}(2)^n$) называется *совершенно нелинейной* или иначе *бент-функцией* (Bent function) [112], если $\forall\ \boldsymbol{\alpha} \in \mathrm{GF}(2)^n$ имеет место равенство: $\mathbf{U}^*_{\boldsymbol{\alpha}}(f) = \pm 2^{n/2}$.

Впервые концепция бент-функций была введена Ротхаусом в 1960-е годы и опубликована в 1976 году [117]. В дальнейшем эти идеи были развиты Диллоном [63] и другими математиками.

Основными свойствами бент-функций $\{f_{bent}(X)\}$ являются следующие[122]:

- любая $f_{bent}(X)$ ($X \in \mathrm{GF}(2)^n$) имеет четное число переменных n и $\mathbf{deg}(f_{bent}) \le n/2$;
- все $f_{bent}(X)$ имеют максимальную нелинейность $\mathbf{N}(f_{bent}(X)) = 2^{n-1} - 2^{(n/2)-1}$ и, соответственно, минимальную линейность $\mathbf{L}(f_{bent}(X)) = 2^{-n/2}$, при этом другие булевы функции не достигают указанных пределов;
- $\{f_{bent}(X)\}$ — единственный класс булевых функций, удовлетворяющих $\mathbf{PC}(n)$, т. е. $\forall \boldsymbol{\alpha} \in \mathrm{GF}(2)^n \mid 1 \le \mathbf{WH}(\boldsymbol{\alpha}) \le n$ $f(X) \oplus f(X \oplus \boldsymbol{\alpha})$ — сбалансированная булева функция;
- $\{f_{bent}(X)\}$ — несбалансированные булевы функции, и каждая из них принимает $2^{n-1} \pm 2^{(n/2)-1}$ раза значение 1 и, соответственно, $2^{n-1} \mp 2^{(n/2)-1}$ раза — значение 0;
- коэффициент корреляции бент-функций с каждой линейной булевой функцией имеет постоянное значение по абсолютной величине $|\mathbf{C}(f_{bent}, f_{lin})| = 2^{-n/2}$ и стремится к 0 с ростом n.

Утверждение 4.12

Если $g_{aff}(X)$ и $f_{bent}(X)$ ($X \in \mathrm{GF}(2)^n$) являются соответственно аффинной и бент-функцией, то $h(X) = g_{aff}(X) \oplus f_{bent}(X)$ будет бент-функцией.

Ротхаусом была сформулирована следующая практически применимая конструкция, позволяющая строить бент-функции любой степени $\mathbf{deg}(f)$ на базе квадратичных форм [117]: $f_{bent}(X) = x_1x_2 \oplus x_3x_4 \oplus \ldots \oplus x_{2k-1}x_{2k} \oplus \phi(x_1, x_3, \ldots, x_{2k-1})$, где ϕ — произвольная булева функция.

Определение 4.26

Векторная булева функция $F(X) = \{f_1(X), \ldots, f_m(X)\}$ называется *векторной бент-функцией*, если $\forall \boldsymbol{\eta} \in \mathrm{GF}(2)^m$ булева функция $T(X) = \boldsymbol{\eta} \bullet F$ является бент-функцией.

Доказано [105], что только векторные бент-функции достигают максимальной нелинейности $\mathbf{N}(f) = 2^{n-1} - 2^{(n/2)-1}$ и, соответственно, минимальной линейности $\mathbf{L}(F) = 2^{-n/2}$ для отображений $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$, при этом такие функции существуют только при $n \ge 2m$ и n четном.

Пример 4.3. Следующие булевы функции являются бент-функциями:

$$f_1(X) = x_1x_3 \oplus x_2x_4,\ \mathbf{N}(f_1) = 6;$$

$$f_2(X) = x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_3x_6,\ \mathbf{N}(f_2) = 28;$$

$$f_3(X) = x_1x_2x_3 \oplus x_2x_4x_5 \oplus x_1x_2 \oplus x_1x_4 \oplus x_2x_6 \oplus x_3x_5 \oplus x_4x_5 \oplus x_7x_8,\ \mathbf{N}(f_3) = 120;$$

$$f_4(X) = x_1x_2x_3 \oplus x_2x_4x_5 \oplus x_3x_4x_6 \oplus x_1x_4 \oplus x_2x_6 \oplus x_3x_4 \oplus$$

$$\oplus\ x_3x_5 \oplus x_3x_6 \oplus x_4x_5 \oplus x_4x_6 \oplus x_7x_8,\ \mathbf{N}(f_4) = 120.$$

Например, таблица истинности, спектр ПУА и автокорреляция функции $f_2(X)$ имеют следующий вид:

$\mathbf{S}(f)$ = 00000000010101010011001101100110000011110101101000111100100101 10;

$\mathbf{U}^*_\alpha(f) =$ 8, 8, 8, 8, 8, 8, 8, –8, 8, –8, 8, –8, 8, –8, 8, 8, 8, 8, 8, –8, –8, 8, 8, –8, 8, 8, 8, –8, –8, 8, 8, –8, –8, –8, 8, 8, 8, 8, –8, –8, –8, 8, 8, –8, 8, –8, –8, 8, –8, –8, 8, 8, –8, –8, –8, –8, 8, –8, 8, –8, –8, 8, –8, 8, 8, 8;

$\mathbf{r}_\alpha(f)$ = 0000000000000000000000000000000000000000000000000000000000000000.

Необходимо отметить, что бент-функции с большим числом переменных еще недостаточно изучены, в частности, не получены и даже не перечислены все такие функции от 10 переменных, однако в настоящее время определены методы построения целых классов бент-функций с произвольно большим количеством переменных и степенью алгебраической нормальной формы. Среди наиболее известных методов можно выделить конструкции Маиорана-Макфарледа [88], Ниберг [105], Карле [51], Диллона [63], Доббертина [64]. Примечательными также являются метод Майера и Штаффенбаха [128] и конструкция Себерри [122], позволяющие строить сбалансированные булевы функции высокой нелинейности и удовлетворяющие критерию $\mathbf{PC}(l)$ высокой степени на основе комбинаций и видоизменений бент-функций. Данные методы использовались для синтеза и оптимизации управляемых подстановочных операций в *разд. 4.3*.

4.2.9. Линейные структуры булевых функций

Альтернативным подходом в оценке нелинейности булевых функций является оценка степени их удаленности от множества функций, имеющих линейные структуры, понятие и свойства которых формулируются в соответствии со следующими определениями и утверждениями:

Определение 4.27

Вектор $\alpha \in \mathrm{GF}(2)^n$ называется *линейной структурой булевой функции* $f(X)$ [124, 128], если функция $\Delta f(X, \alpha) = f(X) \oplus f(X \oplus \alpha)$ является константой, т. е. $\forall X \in \mathrm{GF}(2)^n$ принимает значение 0 или 1.

Утверждение 4.13

Множество линейных структур булевой функции $f(X)$ образует линейное пространство некоторой размерности $l \le n$, которая называется *линейной размерностью булевой функции* $f(X)$ [125].

Утверждение 4.14

Максимальное значение линейной размерности n достигается лишь для аффинных функций.

Утверждение 4.15

При n четном минимальная линейная размерность имеет значение 0 и достигается только для бент-функций [125], при n нечетном минимальная линейная размерность имеет значение 1.

Если обозначить множество булевых функций $f(X)$ ($X \in \mathrm{GF}(2)^n$), имеющих линейные структуры через $\mathbf{LS}(n)$, то *нелинейность* $\mathbf{N}^*(f)$ *булевой функции* $f(X)$ *относительно линейных структур* определяется как: $\mathbf{N}^*(f) = \min\limits_{\forall f_{LS} \in LS(n)} \mathbf{DH}(f, f_{LS})$.

По аналогии с определением общей и средней нелинейности векторных преобразований в смысле расстояния до множества аффинных булевых функций ($\mathbf{N}(F)$ и $\mathbf{N}'(F)$) существуют соответствующие определения и для альтернативного понятия нелинейности векторных булевых функций.

Определение 4.28

Общей и средней нелинейностью векторной булевой функции $F(X)$ относительно линейных структур называются, соответственно, значения следующих выражений [128]:

$$\mathbf{N}^*(F) = \min_{\forall b \in GF(2)^m,\ b \ne 0} \mathbf{N}^*(b \bullet F);\ \mathbf{N}^{*\prime}(F) = \frac{1}{2^m - 1} \sum_{b \in GF(2)^m,\ b \ne 0} \mathbf{N}^*(b \bullet F).$$

Очевидно, что чем дальше функция от множества $\mathbf{LS}(n)$, тем меньше возможность ее приближения линейными функциями. Следует отметить, что альтернативные определения нелинейности $\mathbf{N}(f)$ и $\mathbf{N}^*(f)$ булевых функций тесно связаны. Так, единственными функциями, не имеющими линейных структур, являются бент-функции, для которых $\mathbf{N}^*(f)$ достигает своего максимального значения 2^{n-2}. Именно этот класс булевых функций соответствует максимально возможной нелинейности в смысле расстояния до аффинных булевых функций $\mathbf{N}(f) = 2^{n-1} - 2^{(n/2)-1}$.

В криптографических примитивах в качестве промежуточных преобразований часто используются линейные преобразования исходных векторов. В связи с этим для повышения эффективности анализа таких примитивов следует учитывать следующие свойства преобразований [125]:

Утверждение 4.16

Если $\boldsymbol{B}$ — невырожденная матрица порядка n над полем GF(2), и для некоторой булевой функции $f(X)$ выполняется соотношение $g(X) = f(X\boldsymbol{B})$, то $f(X)$ и $g(X)$ имеют одинаковую алгебраическую степень ($\mathbf{deg}(f) = \mathbf{deg}(g)$) и нелинейность ($\mathbf{N}(f) = \mathbf{N}(g)$, $\mathbf{N}^*(f) = \mathbf{N}^*(g)$), а также — одинаковую линейную размерность.

Утверждение 4.17

Если булева функция имеет вид $f(X) = h_{aff}(X_1) \oplus h_{bent}(X_2)$, где $h_{aff}(X_1)$ ($X_1 \in \mathrm{GF}(2)^l$) и $h_{bent}(X_2)$ ($X_2 \in \mathrm{GF}(2)^{n-l}$, $n - l$ четное) являются, соответственно, аффинной и бент-функциями, то исходная функция имеет линейную размерность l и значение нелинейности $\mathbf{N}(f) = 2^{n-1} - 2^{1/2(n+l)-1}$.

4.2.10. Обобщенные свойства нелинейности криптографических преобразований

В данном разделе приводятся некоторые важные и полезные результаты, которые могут использоваться при синтезе и анализе криптографических примитивов и блочных шифров в целом. В работе [52], в частности, приведена нижняя граница линейности векторных булевых функций $Y = F(X)$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$, имеющая следующий вид: $\mathbf{L}(F) \geq 1/2^n[3\cdot 2^n - 2 - 2(2^n - 1)(2^{n-1} - 1)]^{1/2} = \boldsymbol{Q}(n, m)$, где

- $\boldsymbol{Q}(n, m) < 2^{-n/2}$ при $1 < m < n - 1$;
- $\boldsymbol{Q}(n, m) = 2^{-n/2}$ при $m = n - 1$;
- $\boldsymbol{Q}(n, m) = 2^{-(n-1)/2}$ при $m = n$;
- $\boldsymbol{Q}(n, m) < 2^{-(n-1)/2}$ при $m > n$;
- при $n/2 < m < n$ и $m > n$ минимальное значение линейности не известно.

Как уже отмечалось в *разд. 4.2.8*, только векторные бент-функции при $n \geq 2m$ и n четном достигают минимальной линейности $\mathbf{L}(F) = 2^{-n/2}$ для векторных отображений.

В работах [100, 103] показано, что для $n = m$ и n нечетном существуют преобразования $Y = F(X)$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$, для которых $\mathbf{L}(F) = 2^{-(n-1)/2}$. Такие булевы функции называются *почти бент-функциями* и характеризуются тем, что их корреляция с линейными булевыми функциями принимает только три значения: $\forall \boldsymbol{\eta} \in \mathrm{GF}(2)^m$ $\mathbf{C}(\boldsymbol{\eta}\bullet F, f_{lin}) = \pm 2^{-(n-1)/2}$ или $\mathbf{C}(\boldsymbol{\eta}\bullet F, f_{lin}) = 0$.

Поскольку сбалансированные булевы функции играют особую роль в криптографических преобразованиях, определенный интерес представляют оценки нелинейности таких функций. Для сбалансированных булевых функций, в частности, справедлива следующая оценка сверху [122]:

$$\mathbf{N}(f_{balanced}) \leq \begin{cases} 2^{n-1} - 2^{(1/2)n-1} - 2 & \text{при } n \text{ четном,} \\ [\,[2^{n-1} - 2^{(1/2)n-1}]\,] & \text{при } n \text{ нечетном,} \end{cases}$$

где $[\,[x]\,]$ обозначает максимальное четное целое, не превосходящее x.

Получена также конструктивная оценка снизу [122], имеющая большое практическое значение и доказывающая существование высоко нелинейных сбалансированных булевых функций, в соответствии с которой для любого $n \geq 4$ существует сбалансированные функции, чья нелинейность определяется следующим неравенством:

$$\mathbf{N}(f_{balanced}) \geq \begin{cases} 2^{2^{m-1}} - \frac{1}{2}(2^{2^{m-1}} + 2^{2^{m-2}} + \ldots + 2^{2^2} + 2 \cdot 2^2) & \text{при } n = 2^m, \\ 2^{2^s(2t+1)-1} - \frac{1}{2}(2^{2^{s-1}(2t+1)} + 2^{2^{s-2}(2t+1)} + \ldots + 2^{2t+1} + 2^{t+1}) & \text{при } n = 2^s(2t+1). \end{cases}$$

4.2.11. Общие требования, предъявляемые к криптографическим примитивам

Резюмируя представленные в предыдущих разделах результаты относительно математических свойств подстановочных преобразований, можно сформулировать требования к криптографическим примитивам вида $Y = F(X)$: $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$, играющие принципиально важную роль в различных условиях их применения. К таким требованиям, обеспечивающим обратимость и высокие показатели стойкости блочных шифров к тем или иным методам криптоанализа, относятся:

- биективность (при $n = m$), регулярность или (n, m, k)-эластичность векторного отображения $Y = F(X)$ при $n \geq m$;
- высокая степень корреляционной иммунности ($\mathbf{CI}(l)$, $\mathbf{CI}(l, X^*)$) или свойство $\boldsymbol{R}(l, k)$ образующих булевых функций;
- большая алгебраическая степень $\mathbf{deg}(f)$ и высокая нелинейность $\mathbf{N}(f)$ ($\mathbf{N}^*(f)$) образующих булевых функций;
- высокая общая нелинейность $\mathbf{N}(F)$ ($\mathbf{N}^*(F)$) или в некоторых случаях высокая средняя $\mathbf{N}'(F)$ ($\mathbf{N}^*{}'(F)$) нелинейность векторного отображения в целом;
- низкие автокорреляционные показатели $\mathbf{r}_\alpha(f)$ и $\mathbf{R}_{\alpha,\omega}(F)$ для образующих и векторных булевых функций, обеспечение строгого лавинного критерия $\mathbf{SAC}(k)$ или критерия распространения изменений $\mathbf{PC}(l, k)$ высокой степени l и порядка k.

Поскольку в соответствии с показателями преобразования Уолша-Адамара некоторые из указанных выше требований противоречивы, то возникает важная задача оптимизации образующих булевых функций и векторных отображений в целом. Способы решения этих проблем приводятся в *разд. 4.3.8* на примере синтеза новых криптографических примитивов — управляемых подстановочных операций.

4.3. Модели управляемых подстановочных операций

4.3.1. Определение управляемой подстановочной операции

Неотъемлемой частью проектирования криптографических примитивов является их моделирование как математических объектов. В данном разделе в качестве примера такого проектирования рассматривается синтез новых криптографических примитивов — *управляемых подстановочных операций* (УПО) на основе аппарата булевых функций, представленного в *разд. 4.2*. В этом и последующих разделах данной главы проводится формализация понятия управляемых подстановочных операций, описание их моделей и основных криптографических свойств, на базе которых предполагается построение конкретных реализаций этих операций [13, 20, 72].

Определение 4.29

Управляемой подстановочной операцией называется некоторое отображение $Y = \boldsymbol{G}_V(X, A)$ $(\mathrm{GF}(2)^m \times \mathrm{GF}(2)^n \times \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n)$, осуществляемое с участием двух операндов $X, A \in \mathrm{GF}(2)^n$ $(Y \in \mathrm{GF}(2)^n)$ и зависящее от управляющего вектора $V \in \mathrm{GF}(2)^m$ $(m \geq n)$ (рис. 4.1).

Рис. 4.1. Схема управляемой подстановочной операции
($X = (x_1, x_2, \ldots, x_n)$, $A = (a_1, a_2, \ldots, a_n)$, $V = (v_1, v_2, \ldots, v_m)$, $Y = (y_1, y_2, \ldots, y_n)$)

В соответствии с определением УПО фактически реализует некоторую векторную булеву функцию от трех переменных X, A и V. Однако чтобы подчеркнуть криптографические свойства и особенности механизмов применения таких функций в блочных шифрах, УПО следует рассматривать как некоторую двухместную (или одноместную при $A = 0$) операцию над двумя операндами X и A, зависящую от дополнительного параметра V (управляемую вектором V). При этом в указанной записи сам управляющий вектор V является функцией от эле-

ментов ключа K и/или преобразуемых данных, операнд X соответствует блоку преобразуемых данных, по которому, как правило, обеспечивается биективность, или обратимость преобразования, а операнд A соответствует произвольной преобразуемой информации.

Определение 4.30

Векторная булева функция $Y = \boldsymbol{G}_V(X, A)$, соответствующая конкретному значению управляющего вектора V, называется *модификацией* управляемой подстановочной операции или иначе — *модификацией* $\boldsymbol{G}_V$ и определяет отображение $\mathrm{GF}(2)^n \times \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$.

Общее число различных двоичных отображений с $2n$ входами и n выходами ($\mathrm{GF}(2)^n \times \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$) равно $2^{n2^{2n}}$, следовательно, теоретически максимальное количество возможных модификаций УПО определяется значением $\left|\mathrm{GF}(2)^m\right| = 2^m$, где $m = n2^{2n}$ — размер управляющего вектора V.

Определение 4.31

Две модификации $\boldsymbol{G}_V$ и $\boldsymbol{G}'_V$ $(V' \neq V)$ называются *эквивалентными*, если $\forall X, A$ $\boldsymbol{G}_V(X, A) = \boldsymbol{G}'_V(X, A)$.

Определение 4.32

Модификация $\boldsymbol{G}_V$ называется *уникальной*, если она не имеет эквивалентных модификаций.

Определение 4.33

УПО называется *полной*, если все ее 2^m модификаций уникальны.

Определение 4.34

При $m = n$ управляемая подстановочная операция называется *эффективно полной*, если для любой фиксированной пары

$\{X, A\}$ и $\forall\{V, V' \neq V\} \in \mathrm{GF}(2)^n$ $\boldsymbol{G}_V(X, A) \neq \boldsymbol{G}'_V(X, A)$.

Очевидно, что при $m > n$ УПО не может быть эффективно полной, так как в этом случае отображение $\{V\} \to \{Y\}$ ($\mathrm{GF}(2)^m \to \mathrm{GF}(2)^n$) не будет взаимно однозначным.

Ввиду того, что при фиксированном V УПО определяет отображение $\mathrm{GF}(2)^n \times \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$, то для любой модификации $\boldsymbol{G}_V$ всегда существует множество различных пар $\{\{X, A\}, \{X', A\}\}$, таких, что $\boldsymbol{G}_V(X, A) = \boldsymbol{G}_V(X', A')$. Данная ситуация называется *коллизией по двум операндам* X и A.

Определение 4.35

Для модификации $\boldsymbol{G}_V$ *коллизией по одному операнду*, например по операнду X, называется ситуация, при которой для векторной булевой функции $Y = \boldsymbol{G}_V(X, A)$ при

некотором фиксированном A существуют два значения X и $X' \neq X$, такие, что $\boldsymbol{G_V}(X, A) = \boldsymbol{G_V}(X', A)$.

Определение 4.36

УПО $\boldsymbol{G_V}(X, A)$ называется *обратимой по X*, если для любых фиксированных $\{A, V\}$ существует *однозначное обратное преобразование* $\boldsymbol{G}'_V$ такое, что $\forall X$ из равенства $Y = \boldsymbol{G_V}(X, A)$ следует, что $X = \boldsymbol{G}_V^{-1}(Y, A)$.

В общем виде векторная булева функция $Y = \boldsymbol{G_V}(X, A)$ определяет отображение $\{V\}\times\{X\}\times\{A\} \to \{Y\}$ $(\mathrm{GF}(2)^m\times\mathrm{GF}(2)^n\times\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n)$, которое при фиксировании пар $\{A, V\}$, $\{X, V\}$ и $\{X, A\}$ индуцирует, соответственно, следующие отображения:

$$Y=\sigma_A^{(V)}(X)\colon (\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n),\ Y=\nu_X^{(V)}(A)\colon (\mathrm{GF}(2)^n \to \mathrm{GF}(2)^n),$$

$$Y = \mu_{X,A}(V)\colon (\mathrm{GF}(2)^m \to \mathrm{GF}(2)^n).$$

Очевидно, что векторные булевы функции $\boldsymbol{G_V}$, $\sigma_A^{(V)}(X)$, $\nu_X^{(V)}(A)$ и $\mu_{X,A}$ взаимосвязаны и каждая из них отражает определенные свойства УПО. В дальнейшем по необходимости могут использоваться различные способы представления УПО.

4.3.2. Общая модель управляемой подстановочной операции

Рассмотрим в качестве базового представления УПО один из указанных в предыдущем разделе вариантов отображений (векторных булевых функций):

$$Y = \sigma_A^{(V)}(X) = \{y_1, y_2, ..., y_n\} = \{f_1^{(V)}(X, A), f_2^{(V)}(X, A), ..., f_n^{(V)}(X, A)\}, \tag{4.1}$$

где $y_i = f_1^{(V)}(X, A)$ — некоторые булевы функции, записанные в алгебраической нормальной форме ($X, A, Y\in\mathrm{GF}(2)^n$, $V\in\mathrm{GF}(2)^m$, $m \geq n$).

Определение 4.37

Представление УПО в терминах векторной булевой функции $Y=\sigma_A^{(V)}(X)$ называется *общей моделью УПО* и обозначается $\mathbf{M}_\sigma$, а функции $\{ f_i^{(V)} \}$ называются *образующими функциями* модели $\mathbf{M}_\sigma$.

Если данная модель содержит нетривиальный операнд A, то имеем общую модель бинарной УПО, а в случае, когда модель не содержит операнда A — общую модель унарной УПО. Из определения 4.37 следует, что в широком смысле модель $\mathbf{M}_\sigma$ можно рассматривать как общую модель произвольного криптографического примитива, осуществляющего сюръективное отображение, в том числе и управляемой перестановки.

Определение 4.38

Каждая УПО, соответствующая какой-либо модели, называется *реализацией* этой модели.

Определение 4.39

Две реализации $\boldsymbol{G}_V$ и $\boldsymbol{G}'_V$ какой-либо модели называются *эквивалентными*, если $\forall X, A, V\ \boldsymbol{G}_V(X, A) = \boldsymbol{G}'_V(X, A)$. Реализация некоторой модели считается *уникальной*, если для нее не существует эквивалентных реализаций.

Поскольку модель $\mathbf{M}_\sigma$ имеет слишком общее представление для целого класса отображений, практическое использование ее реализаций при создании криптографических примитивов связано с необходимостью дальнейшей детализации ее структуры, зависящей от конкретной схемы блочного шифра.

В частности, представляется перспективным использование УПО вместо операции "⊕" в традиционной схеме Фейстеля [12, 66] (см. рис. 4.2 и 4.3), состоящей из r раундов (итераций) блочного преобразования с n-битовым входным блоком (исходным текстом) $X \in \mathrm{GF}(2)^n$ и множеством раундовых ключей $K = \{K_1, \ldots, K_r\}$, где $K_i = (k_1^{(i)}, k_2^{(i)}, \ldots, k_{n/2}^{(i)}) \in \mathrm{GF}(2)^{n/2}$. Алгоритм схемы Фейстеля имеет следующий вид:

For $i = 1$ **to** $i = r–1$ **do**

[$X_R(i) = X_L(i–1)$, $X_L(i) = \boldsymbol{E}(X(i–1), K_i) = F(X_L(i–1), K_i) \oplus X_R(i–1)$, $X(i) = X_L(i) \parallel X_R(i)$,

где $i \in \{1, \ldots, r\}$ $K_i \in \mathrm{GF}(2)^{n/2}$ — текущий *раундовый ключ*,
$\boldsymbol{E}(X(i–1), K_i)$ — *раундовое преобразование*, $\boldsymbol{F}(X_L(i–1), K_i)$ — *раундовая функция*,
"‖" — *операция конкатенации* (объединения),

$X(i) \in \mathrm{GF}(2)^n$, $\{X_L(i), X_R(i)\} \in \mathrm{GF}(2)^{n/2}$, $X = X_L \parallel X_R = X_L(0) \parallel X_R(0) = X(0)$ — *исходный блок* данных].

For $i = r$ **do**

[$Y_R = \boldsymbol{F}(X_L(r), K_r) \oplus X_R(r)$, $Y_L = X_L(r)$, $Y = Y_L \parallel Y_R$,

где $Y \in \mathrm{GF}(2)^n$ — *выходной блок* функции преобразования].

Особенностью схемы Фейстеля является тот факт, что, во-первых, обратное преобразование проводится по той же схеме, но при обратном порядке использования раундовых ключей, и, во-вторых, преобразование не нуждается в обратимости раундовой функции $\boldsymbol{F}$.

Использование УПО в указанной выше и подобных итеративных схемах предъявляет ряд требований:

- при любых фиксированных $\forall X$ и A векторная булева функция $Y = \sigma_A^{(V)}(X)$ должна быть обратимой по X, чтобы обеспечить обратимость блочного шифра;

Рис. 4.2. Один раунд преобразований по схеме Фейстеля

Рис. 4.3. Один раунд преобразований по схеме Фейстеля с использованием управляемой подстановочной операции

- в ряде случаев желательно, чтобы одновременно с условием обратимости по X были биективными по A и, соответственно, по V отображения $\nu_X^{(V)}(A)$ и $\mu_{X,A}(V)$ *(см. разд. 4.3.1)*, что создает условия для статистической равномерности выходных характеристик по различным аргументам преобразования *(см. разд. 4.2.1)*;
- каждая модификация УПО $\boldsymbol{G_V}(X, A)$ должна быть по возможности регулярным или (n, m, k)-эластичным отображением $\mathrm{GF}(2)^n \times \mathrm{GF}(2)^n \to \mathrm{GF}(2)^n$, чтобы препятствовать возможности прямых статистических атак на блочные шифры *(см. разд. 4.2.2 и 4.2.7)*;
- механизм управления УПО посредством вектора V должен обеспечивать формирование большого количества уникальных модификаций, желательно при этом, чтобы УПО была полной, а по возможности — эффективно полной *(см. разд. 4.3.1)*;
- в большинстве случаев отображение $\sigma_A^{(V)}(X)$ должно иметь обратное по X преобразование низкой сложности;
- для каждой формируемой модификации УПО $\boldsymbol{G_V}(X, A)$ количество коллизий по двум операндам X и A должно быть по возможности небольшим *(см. разд. 4.3.1)*;

- векторная булева функция $Y=\sigma_A^{(V)}(X)$ должна обладать необходимыми криптографическими свойствами, связанными с высокой нелинейностью и хорошими корреляционными и автокорреляционными показателями *(см. разд. 4.2.4—4.2.6)*;
- векторная булева функция $Y=\sigma_A^{(V)}(X)$ должна иметь относительно небольшую вычислительную сложность и обеспечивать высокое быстродействие при аппаратной и/или программной реализации.

Для дальнейшей детализации общей модели $\mathbf{M}_\sigma$ в соответствии с указанными требованиями рассмотрим свойство обратимости УПО.

В случае, когда реализация модели $\mathbf{M}_\sigma$ обратима по X, она образует подстановку на множестве $\{X\}$ при фиксированных значениях операндов A и V. В терминах модели $\mathbf{M}_\sigma$ по определению 4.36 это означает, что для каждой такой подстановки $\sigma_A^{(V)}$ существует единственная *обратная подстановка* $\sigma_A^{(V)\,-1}$, т. е. если $\forall i \in \{1, \ldots, n\}$ $y_i = f_i^{(V)}(X, A)$, то существуют обратные *образующие булевы функции* $\{ f_i'^{(V)} \}$, такие, что $x_i = f_i'^{(V)}(Y, A)$.

В соответствии с определениями биективного отображения *(см. разд. 4.2.1)* и обратной подстановки справедливо следующее очевидное утверждение:

Теорема 4.1

Если модификация УПО обратима по X, то отображение $\sigma_A^{(V)}(X)$ является биективным, а соответствующая векторная булева функция $Y = \boldsymbol{G_V}(X, A)$ не имеет коллизий по X.

Доказательство

Предположим, что утверждение ложно. Тогда существуют $X, X' \neq X$ и Y такие, что $Y=\sigma_A^{(V)}(X)$ и $Y=\sigma_A^{(V)}(X')$. Так как по условию для $\sigma_A^{(V)}$ существует обратная ей подстановка $\sigma_A^{(V)\,-1}$, то $\sigma_A^{(V)\,-1}(Y)=X$ и $\sigma_A^{(V)\,-1}(Y) = X'$, где $X' \neq X$. Это невозможно ввиду однозначности обратной подстановки $\sigma_A^{(V)\,-1}$, т. е. отображение $Y = \boldsymbol{G_V}(X, A)$ не имеет коллизий по X. Что и требовалось доказать.

Определение 4.40

Обратимая по X модификация УПО называется обратимой по X без изменения функциональной схемы преобразования или иначе — *параметрически обратимой*, если каждая векторная булева функция $\boldsymbol{G_V}$ зависит еще от одного *параметра обратимости* $\boldsymbol{e}$ ($\boldsymbol{G}_{V,e}$), принимающего два значения: 0 — "прямое преобразование" и 1 — "обратное преобразование", для любых фиксированных $\{A, V\}$, $e \in \{0, 1\}$ и $\forall X$ выполняется условие: $Y = \boldsymbol{G_V}(X, A) = \boldsymbol{G}_{V,0}(X, A)$ и $X = \boldsymbol{G_V}^{-1}(Y, A) = \boldsymbol{G}_{V,1}(Y, A)$.

На языке общей модели $\mathbf{M}_\sigma$ это означает, что прямые и обратные подстановки симметричны и связаны условием:

$\sigma_A^{(V)} = \sigma_A^{(V,0)}$ и $\sigma_A^{(V)\,-1} = \sigma_A^{(V,1)}$, т. е. $\forall i \in \{1, \ldots, n\}$ $y_i = f_i^{(V,0)}(X, A)$, $x_i = f_i^{(V,1)}(Y, A)$.

Таким образом, общая модель $\mathbf{M}_\sigma$ параметрически обратимой УПО приобретает следующий вид: $\mathbf{M}_{\sigma,e}$: $\sigma_A^{(V,e)}(X) = \{f_1^{(V,e)}(X, A), f_2^{(V,e)}(X, A), \ldots, f_n^{(V,e)}(X, A)\}$.

Понятие параметрической обратимости имеет большое значение при создании криптографических примитивов, поскольку позволяет проводить прямое и обратное преобразование в блочных шифрах с помощью одной и той же электронной схемы за счет изменения параметра обратимости e.

4.3.3. Последовательная модель управляемой подстановочной операции

В общей модели УПО существует класс биективных по X отображений, в котором возможна реализация параметрически обратных преобразований. Данный класс получил название последовательной модели УПО.

Определение 4.41

Последовательной моделью $\mathbf{S}_\sigma$ называется частная модель общей модели $\mathbf{M}_\sigma$ ($\mathbf{S}_\sigma \subset \mathbf{M}_\sigma$), для которой выполняется следующее условие:

$$\forall i \in \{1, \ldots, n\} \;\; f_i^{(V)}(X, A) = f_i^{(V)}(x_1, \ldots, x_i, A).$$

Таким образом, для модели $\mathbf{S}_\sigma$ имеет место запись:

$$\sigma_A^{(V)}(X) = \{f_1^{(V)}(x_1, A), f_2^{(V)}(x_1, x_2, A), \ldots, f_n^{(V)}(x_1, \ldots, x_n, A)\}. \tag{4.2}$$

В рамках этой модели достаточно хорошо реализуются указанные в *разд. 4.3.2* требования, накладываемые на УПО.

Для уточнения свойств последовательной модели $\mathbf{S}_\sigma$ введем следующие обозначения: $\forall i \le n$ $(x_1, \ldots, x_i) = X^{(i)} \in \mathrm{GF}(2)^i$, $(a_1, \ldots, a_i) = A^{(i)} \in \mathrm{GF}(2)^i$, $(y_1, \ldots, y_i) = Y^{(i)} \in \mathrm{GF}(2)^i$ (в частности, $X = X^{(n)}$, $A = A^{(n)}$ и $Y = Y^{(n)}$).

В указанных обозначениях параметрически обратимая последовательная модель $\mathbf{S}_\sigma$ становится моделью $\mathbf{S}_{\sigma,e}$ следующего вида:

$$\sigma_A^{(V,e)}(X) = \{f_1^{(V,e)}(x_1, A), f_2^{(V,e)}(X^{(2)}, A), \ldots, f_n^{(V,e)}(X^{(n)}, A)\}. \tag{4.3}$$

Теорема 4.2

Для реализаций последовательной модели $\mathbf{S}_\sigma$ *достаточным* условием биективности (обратимости) отображения $\sigma_A^{(V)}$ по X является выполнение следующего соотношения:

$$\forall i \in \{1, \ldots, n\} \;\; f_i^{(V)}(X^{(i)}, A) = x_i \oplus \varphi_i^{(V)}(X^{(i-1)}, A), \tag{4.4}$$

где x_0 — начальное условие. Множество таких реализаций образует *биективную* (*обратимую*) *модель* $\mathbf{S}'_\sigma \subset \mathbf{S}_\sigma$.

Доказательство

Возьмем произвольный Y и построим для него некоторый X такой, что $Y = \sigma_A^{(V)}(X)$:

$$x_1 = y_1 \oplus \varphi_1^{(V)}(x_0, A), \;\; x_2 = y_2 \oplus \varphi_2^{(V)}(x_0, x_1, A), \ldots, \;\; x_n = y_n \oplus \varphi_n^{(V)}(X^{(n-1)}, A).$$

Для полученного X выполняется равенство $Y = \sigma_A^{(V)}(X)$. Предположим, что $\exists X' \neq X$ такой, что $Y = \sigma_A^{(V)}(X')$, тогда $y_1 = x'_1 \oplus \varphi_1^{(V)}(x_0, A)$ и $y_1 = x_1 \oplus \varphi_1^{(V)}(x_0, A)$, следовательно, $x'_1 = x_1$. Далее рассматривая все y_i, где i последовательно изменяется от 2 до n, получаем, что $x'_1 = x_i$ для всех $i \in \{1, \ldots, n\}$, т. е. $X' = X$. Таким образом, полученное выше преобразование является алгоритмом однозначного формирования обратной подстановки $\sigma_A^{(V)^{-1}}$. Что и требовалось доказать.

Теорема 4.2 имеет следствия:

Следствие 4.3

Если $\forall i \in \{1, \ldots, n\}$ выполняется соотношение:

$$f_i^{(V)}(X^{(i)}, A) = x_i \oplus a_i \oplus \varphi_i'^{(V)}(X^{(i-1)}, A^{(i-1)}),$$

то преобразование $Y = \boldsymbol{G}_V(X, A) = \sigma_A^{(V)}(X)$ является обратимым отдельно по X и по A, а следовательно, биективным и не имеющим коллизий отдельно по X и по A.

Доказательство следует из доказательств теорем 4.2 и 4.1 при рассмотрении УПО как подстановки $Y = \nu_X^{(V)}(X)$.

Следствие 4.4

Если механизм управления УПО функционально имеет вид:

$$\forall i \in \{1, \ldots, n\} \;\; f_i^{(V)}(X, A) = \nu_i \oplus \varphi_i''^{(V^{(i-1)})}(X, A),$$

то УПО в соответствии с определением 4.34 является эффективно полной.

Доказательство следует из теорем 4.2 и 4.1 при рассмотрении УПО как подстановки $Y = \mu_{X,A}(V)$.

Следствие 4.5

Если $\forall i \in \{1, \ldots, n\}$ выполняется соотношение:

$$f_i^{(V)}(X^{(i)}, A) = x_i \oplus a_i \oplus \nu_i \oplus \varphi_i'''^{(V^{(i-1)})}(X^{(i-1)}, A^{(i-1)}),$$

то УПО является эффективно полной, а все модификации $\boldsymbol{G}_V$ обратимы отдельно по X и по A, а следовательно, биективны и не имеют коллизий отдельно по X и по A.

Теорема 4.3

Отображение $Y = \boldsymbol{G}_V(X, A) = \sigma_A^{(V)}(X)$, для которого выполняется условие (4.4) теоремы 4.2, является регулярным в соответствии с определением 4.6.

Доказательство

Рассмотрим произвольную линейную комбинацию компонентов вектора:

$$\sigma_A^{(V)}(X) = \{f_1^{(V)}(x_1, A), f_2^{(V)}(X^{(2)}, A), ..., f_n^{(V)}(X^{(n)}, A)\}:$$

$$\boldsymbol{\alpha} \bullet \sigma_A^{(V)}(X) = \bigoplus_{i=1}^{n} \alpha_i f_i^{(V)}(X^{(i)}, A) = \bigoplus_{i=1}^{n} \varphi_n^{(V)}(X^{(n-1)}, A) \oplus \bigoplus_{i=1}^{n} \alpha_i x_i .$$

Пусть k — максимальное значение, при котором $\alpha_k = 1$, тогда $\boldsymbol{\alpha} \bullet \sigma_A^{(V)}(X) = x_k \oplus \chi(X^{(k-1)}, A)$. Поскольку x_k сбалансированная функция, то в соответствии с утверждением 4.1 функция $\boldsymbol{\alpha} \bullet \sigma_A^{(V)}(X)$ будет также сбалансированной. Следовательно, в соответствии с утверждением 4.2 отображение $\sigma_A^{(V)}(X)$ является регулярным. Что и требовалось доказать.

Следствие 4.6

В любой реализации последовательной модели $\mathbf{S}'_\sigma$ обеспечивается равновероятный выход по каждому компоненту векторной булевой функции $Y = \sigma_A^{(V)}(X)$, т. е. $\forall i \in \{1, \ldots, n\}$ $P\{y_i = 1\} = P\{y_i = 0\} = 1/2$.

Теорема 4.4

Для обеспечения параметрической обратимости подстановки $\sigma_A^{(V,e)}(X)$ в последовательной модели УПО $\mathbf{S}'_{\sigma,e}$ *достаточно* в качестве функций преобразования последовательно от $i = 1$ до n использовать следующие функции:

$$\forall i \in \{1, \ldots, n\} \;\; y_i = f_i^{(V,e)}(X^{(i)}, A) = x_i \oplus \varphi_i^{(V)}(W_e^{(i-1)}, A), \tag{4.5}$$

где $x_0 = w_0$ — начальное условие, $W_e^{(i-1)}$ определяется δ_e — дельта-функцией вида:

$$\text{при } i > 1 \;\; W_e^{(i-1)} = \delta_e(X^{(i-1)}, Y^{(i-1)}) = \begin{cases} X^{(i-1)} & \text{при } e = 0, \\ Y^{(i-1)} & \text{при } e = 1, \end{cases}$$

(при $i = 1$ $W_e^{(0)} = x_0 = w_0$).

Доказательство

Действительно, из условий теоремы при $e = 0$ имеем:

$$y_i = f_i^{(V,0)}(X^{(i)}, A) = x_i \oplus \varphi_i^{(V)}(\delta_0(X^{(i-1)}, Y^{(i-1)}), A) = x_i \oplus \varphi_i^{(V)}(X^{(i-1)}, A) . \tag{4.6}$$

Далее в соотношение (4.5) вместо x_i подставим y_i и заменим $e = 0$ на $e = 1$:

$$x'_i = f_i^{(V,1)}(Y^{(i)}, A) = y_i \oplus \varphi_i^{(V)}(\delta_1(Y^{(i-1)}, X'^{(i-1)}), A) = y_i \oplus \varphi_i^{(V)}(X'^{(i-1)}, A). \qquad (4.7)$$

Последовательно выбирая в (4.7) значения i от 1 до n и используя (4.6), получим:

$$x'_1 = y_1 \oplus \varphi_1^{(V)}(x_0, A) = (x_1 \oplus \varphi_1^{(V)}(x_0, A)) \oplus \varphi_1^{(V)}(x_0, A) = x_1,$$

$$x'_2 = y_2 \oplus \varphi_2^{(V)}(x_0, x'_1, A) = y_2 \oplus \varphi_2^{(V)}(x_0, x_1, A) =$$
$$= (x_2 \oplus \varphi_2^{(V)}(x_0, x_1, A)) \oplus \varphi_2^{(V)}(x_0, x_1, A) = x_2,$$

и т. д. до $x'_n = x_n$, т. е. формируем для подстановки $\sigma_A^{(V)}(X)$ параметрически обратную подстановку $\sigma_A^{(V)^{-1}}$ ($Y = \sigma_A^{(V,0)}(X)$, $X = \sigma_A^{(V,1)}(Y)$). Что и требовалось доказать.

Особенность данного преобразования заключается в необходимости специальной схемной или программной реализации последовательной серии из (n–1)-го переключателя $\{\delta_e(X^{(i-1)}, Y^{(i-1)})\}$.

Определение 4.42

Модификация УПО называется *параметрически схемно-обратимой*, если она параметрически обратима в соответствии с условием (4.5). Множество таких реализаций образует *параметрически схемно-обратимую последовательную модель* $\mathbf{S}'_{\sigma,e_S}$, а сами реализации модели называются *управляемыми схемными преобразователями* (УСП).

Из данного определения и теоремы 4.3 вытекает следствие:

Следствие 4.7

Любая реализация последовательной модели $\mathbf{S}'_{\sigma,e}$ имеет реализацию в модели $\mathbf{S}'_{\sigma,e_S}$.

Необходимо отметить, что главной особенностью реализаций УПО последовательной модели $\mathbf{S}'_{\sigma,e_S}$ является возможность одновременного выполнения прямого преобразования сразу по всем компонентам вектора X, тогда как обратное преобразование проводится с временной задержкой последовательно от младшего ($i = 1$) до старшего компонента ($i = n$) вектора X в соответствии с алгоритмом из доказательства теоремы 4.4.

Следует заметить, что кроме схемных существуют и *математически (функционально) обратимые* реализации последовательной модели УПО *(сумматоры)*, которые являются предметом рассмотрения следующего раздела.

Определение 4.43

Реализация УПО называется *параметрически функционально-обратимой*, если ее параметрическая обратимость обеспечивается за счет изменения параметра $e \in \{0, 1\}$ в математической записи преобразования, осуществляемого этой УПО.

4.3.4. Рекурсивная модель управляемой подстановочной операции

Определенный недостаток модели $\mathbf{S}'_{\sigma,e}$ заключается в возрастании сложности образующих функций $y_i = f_i^{(V,e)}(X^{(i)}, A)$ с ростом i, что не позволяет эффективно обеспечить зависимость каждого i-го бита выхода УПО от всех i значений аргументов. Этот недостаток устраняется в частных реализациях рекурсивного представлении УПО в модели $\mathbf{S}'_{\sigma}$. Кроме того, рекурсивный подкласс УПО существенно расширяет возможности последовательной модели $\mathbf{S}_{\sigma}$ за счет свойства *параметрической функциональной обратимости*, т. е. обратимости, обеспечиваемой за счет изменения параметра $e \in \{0, 1\}$ в математической записи преобразования (см. определение 4.43).

Определение 4.44

Рекурсивной моделью УПО называется частный вид последовательной модели $\mathbf{S}_{\sigma}$, для которой выполняются следующие условия:

$$\sigma_A^{(V)}(X) = \{f_1^{(V)}(x_1, u_0, A), f_2^{(V)}(x_2, u_0, u_1, A), ..., f_n^{(V)}(x_n, u_0, ..., u_{n-1}, A)\} \tag{4.8}$$

где $\forall j \in \{1, 2, \ldots, n-1\}$ $u_j = \Psi_j^{(V)}(x_j, u_0, ..., u_{j-1}, A)$, $u_0 = const$ — *начальное условие*.

Очевидно, что рекурсивную модель (4.8) можно рассматривать как некоторый зависящий от случайного параметра V *недетерминированный автомат* вида:

$$\mathbf{\Omega}_{\sigma}(\{X, A\}, Y, \{U^{(i)}\}, \{ f_i^{(V)} \}, \{ \Psi_j^{(V)} \}),$$

где $\{X, A\}$ — *входной алфавит*, Y — *выходной алфавит*,
$\{U^{(i)}\} = \{(u_1, u_2, \ldots, u_i)\} \subseteq \mathrm{GF}(2)^i$ — *алфавит внутренних состояний*,
$\{ f_i^{(V)} \}$ — *функции выхода*, $\{ \Psi_j^{(V)} \}$ — *переходные функции*.

В связи с этим рекурсивную модель (4.8) можно обозначить следующим образом:

$$\mathbf{\Omega}_{\sigma} = \{\{f_i^{(V)}(x_i, U^{(i-1)}, A)\}, \{\Psi_j^{(V)}(x_j, U^{(j-1)}, A)\}\},$$

где $\{ f_i^{(V)} \}$ и $\{ \Psi_j^{(V)} \}$ — *образующие функции* модели $\mathbf{\Omega}_{\sigma}$.

Необходимо отметить, что рекурсивную модель можно представить и в более общем виде, а именно использовать на каждом шаге рекурсии не одну, а множе-

ство переходных функций, т. е. записать *обобщенную рекурсивную модель* следующим образом:

$$\mathbf{\Omega}_{\sigma}^{*} = \{\{f_i^{(V)}(x_i, U_1^{(i-1)}, ..., U_k^{(i-1)}, A)\}, \{\Psi_{1,j}^{(V)}(x_j, U_1^{(j-1)}, A), ..., \Psi_{k,j}^{(V)}(x_j, U_k^{(j-1)}, A)\}\}.$$

Однако для простоты изложения в дальнейшем будет использоваться исходное определение для *базовой рекурсивной модели* $\mathbf{\Omega}_{\sigma}$ с одной переходной функцией, принимая во внимание то обстоятельство, что приводимые доказательства и рассуждения легко распространяются и на обобщенную модель $\mathbf{\Omega}_{\sigma}^{*}$.

Утверждение 4.18

В соответствии с теоремой 4.2 *достаточным* условием биективности (обратимости) реализаций рекурсивной модели $\mathbf{\Omega}_{\sigma} \subset \mathbf{S}_{\sigma}$ являются следующие ограничения, накладываемые на функции выхода:

$$y_i = f_i^{(V)}(x_i, u_0, ..., u_{i-1}, A) = f_i^{(V)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A).$$

Такие реализации образуют *биективную (обратимую) рекурсивную модель* $\mathbf{\Omega}'_{\sigma} \subset \mathbf{\Omega}_{\sigma}$.

Отсюда класс *параметрически обратимых реализаций рекурсивной модели* можно представить в виде следующей модели:

$$\mathbf{\Omega}'_{\sigma,e} = \{\{f_i^{(V,e)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i^{(V,e)}(U^{(i-1)}, A)\}, \{\Psi_j^{(V,e)}(x_j, U^{(j-1)}, A)\}\}.$$

Рассмотрим *частную параметрически обратимую рекурсивную модель* $\mathbf{\Omega}''_{\sigma,e} \subset \mathbf{\Omega}'_{\sigma,e}$, в которой обратное преобразование осуществляется только за счет параметра обратимости e в функциях перехода $\{\Psi_j^{(V,e)}\}$:

$$\mathbf{\Omega}''_{\sigma,e} = \{\{f_i^{(V)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A)\},\ \{\Psi_j^{(V,e)}(x_j, U^{(j-1)}, A)\}\}.$$

В рамках данной рекурсивной модели, представленной на рис. 4.4, достаточно эффективно реализуются параметрически функционально и схемно-обратимые УПО. В связи с этим справедливы следующие утверждения, позволяющие в полной мере определить модель $\mathbf{\Omega}''_{\sigma,e}$:

Лемма 4.1

Если в рекурсивной модели $\mathbf{\Omega}''_{\sigma,e} \subset \mathbf{\Omega}'_{\sigma,e}$ каждая функция $\varphi_i^{(V)}(U^{(i-1)}, A)$ существенно зависит от u_{i-1}, то любая реализация этой модели будет параметрически обратимой *тогда и только тогда*, когда значения переходных функций являются инвариантными относительно прямого и обратного преобразования, т. е. $\forall i \in \{1, \ldots, n-1\}$ $u_i = u'_i$, где $u_i = \Psi_i^{(V,e)}(x_i, U^{(i-1)}, A)$, $u'_i = \Psi_i^{(V,e\oplus 1)}(y_i, U'^{(i-1)}, A)$.

Рис. 4.4. Общая схема параметрически обратимой рекурсивной модели УПО $\boldsymbol{\Omega}''_{\sigma,e}$

($\boldsymbol{e} = 0$ — "сложение", $\boldsymbol{e} = 1$ — "вычитание")

Доказательство

Рассмотрим подстановки $\sigma_A^{(V,e)}$ в общем виде:

$$y_i = f_i^{(V,e)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i^{(V,e)}(U^{(i-1)}, A)\,. \tag{4.9}$$

Для получения обратного преобразования в функциях $\{\varphi_i^{(V,e)}\}$ заменим e на $e \oplus 1$ и вместо x_i подставим y_i из (4.9):

$$x_i' = y_i \oplus \varphi_i^{(V,e\oplus 1)}(U'^{(i-1)}, A) = x_i \oplus \varphi_i^{(V,e)}(U^{(i-1)}, A) \oplus \varphi_i^{(V,e\oplus 1)}(U'^{(i-1)}, A)\,.$$

Для обеспечения равенства $x_i' = x_i$ необходимо и достаточно, чтобы выполнялось следующее условие:

$$\varphi_i^{(V,e)}(U^{(i-1)}, A) \oplus \varphi_i^{(V,e\oplus 1)}(U'^{(i-1)}, A)\,. \tag{4.10}$$

Так как по условию функции $\{\varphi_i^{(V)}\}$ не зависят от параметра e, то условие (4.10) приобретает следующий вид: $\varphi_i^{(V)}(U^{(i-1)}, A) \oplus \varphi_i^{(V)}(U'^{(i-1)}, A)$.

Для доказательства леммы рассмотрим функции в виде алгебраической нормальной формы и используем метод математической индукции.

При $i = 1$ имеем $\varphi_1^{(V)}(u_0, A) = \varphi_1^{(V)}(u_0, A)$ и $u_0 = u_0$, т. е. лемма справедлива.

Предположим, что лемма справедлива для $i = j$ ($u_{j-1} = u'_{j-1}$) и докажем, что она справедлива для $i = j + 1$ ($u_j = u'_j$). В этом случае имеем равенство:

$$\varphi_{j+1}^{(V)}(u_0, u_1, ..., u_{j-1}, u_j, A) = \varphi_{j+1}^{(V)}(u_0, u_1, ..., u_{j-1}, u'_j, A)\,.$$

Выделим в полиномах алгебраической нормальной формы этих функций члены, содержащие u_j и u'_j:

$$\varphi_{j+1}^{(V)}(u_0, u_1, ..., u_{j-1}, u_j, A) = u_j H(U^{(j-1)}, A) \oplus H'(U^{(j-1)}, A);$$

$$\varphi_{j+1}^{(V)}(u_0, u_1, ..., u_{j-1}, u'_j, A) = u'_j H(U^{(j-1)}, A) \oplus H'(U^{(j-1)}, A),$$

Разность этих полиномов должна равняться 0 при любых значениях аргументов: $\Delta\varphi_{j+1}^{(V)} = (u_j \oplus u'_j)H(U^{(j-1)}, A) \equiv 0$. Так как $\varphi_{j+1}^{(V)}$ существенно зависит от u_j, то тождество $H(U^{(j-1)}, A) \equiv 0$ выполняться не может, следовательно, $\Delta\varphi_{j+1}^{(V)} \equiv 0$ тогда и только тогда, когда $u_j = u'_j$. Что и требовалось доказать.

Теорема 4.5

Если в рекурсивной модели $\mathbf{\Omega}''_{\sigma,e} \subset \mathbf{\Omega}'_{\sigma,e}$ каждая функция $\varphi_i^{(V)}(U^{(i-1)}, A)$ существенно зависит от u_{i-1}, то любая реализация этой модели будет параметрически функционально-обратимой *тогда и только тогда*, когда переходные функции и функции выхода связаны следующим *определяющим рекуррентным соотношением*:

$$u_i = \Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = [e\varphi_i^{(V)}(U^{(i-1)}, A) \oplus x_i]L_i^{(V)}(U^{(i-1)}, A) \oplus L_i'^{(V)}(U^{(i-1)}, A),$$

где $L_i^{(V)}(U^{(i-1)}, A), L_i'^{(V)}(U^{(i-1)}, A)$ — некоторые *образующие полиномы*, а u_0 — *начальное условие*.

Доказательство

В соответствии с результатами леммы 4.1 любая реализация рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$ будет параметрически обратимой тогда и только тогда, когда $\forall i \in \{1, \ldots, n-1\}$ $u_i = u'_i$, т. е.

$$\Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = \Psi_i^{(V,e\oplus 1)}(x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A), U^{(i-1)}, A).$$

Рассмотрим полиномы алгебраической нормальной формы переходных функций $\{\Psi_i^{(V,e)}\}$ и $\{\Psi_i^{(V,e\oplus 1)}\}$. Представим эти полиномы в определенном виде, который назовем *каноническим*, выделив члены, содержащие e и x_i, и найдем разность этих полиномов:

$$u_i = \Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = eR_i^{(V)}(x_i, U^{(i-1)}, A) \oplus R_i'^{(V)}(x_i, U^{(i-1)}, A) =$$

$$= e(x_i M_i^{(V)}(U^{(i-1)}, A) \oplus M_i'^{(V)}(U^{(i-1)}, A)) \oplus x_i M_i''^{(V)}(U^{(i-1)}, A) \oplus M_i'''^{(V)}(U^{(i-1)}, A) =$$

$= ex_i M_i^{(V)} \oplus eM_i'^{(V)} \oplus x_i M_i''^{(V)} \oplus M_i'''^{(V)}$ — *канонический вид переходной функции.*

Далее имеем:

$$u_i' = \Psi_i^{(V,e\oplus 1)}(x_i \oplus \varphi_i^{(V)}, U^{(i-1)}, A) = (e \oplus 1)[x_i \oplus \varphi_i^{(V)}]M_i^{(V)} \oplus$$

$$\oplus (e \oplus 1)M_i'^{(V)} \oplus [x_i \oplus \varphi_i^{(V)}]M_i''^{(V)} \oplus M_i'''^{(V)} =$$

$$= ex_i M_i^{(V)} \oplus e\varphi_i^{(V)} M_i^{(V)} \oplus eM_i'^{(V)} \oplus x_i M_i^{(V)} \oplus$$

$$\oplus \varphi_i^{(V)} M_i^{(V)} \oplus M_i'^{(V)} \oplus \varphi_i^{(V)} M_i''^{(V)} \oplus x_i M_i''^{(V)} \oplus M_i'''^{(V)}.$$

$$\Delta\Psi_i^{(V,e)} = u_i \oplus u_i' = e\varphi_i^{(V)} M_i^{(V)} \oplus x_i M_i^{(V)} \oplus \varphi_i^{(V)} M_i^{(V)} \oplus M_i'^{(V)} \oplus \varphi_i^{(V)} M_i''^{(V)}.$$

Отсюда необходимое и достаточное условия параметрически функциональной обратимости приобретают вид: $\forall i \in \{1, \ldots, n-1\}$ $\Delta\Psi_i^{(V,e)} = 0$.

Так как для обеспечения этих условий для любого e функции $\Delta\Psi_i^{(V,e)}$ не должны зависеть от параметра e и условия должны выполняться для любых пар $\{U^{(i-1)}, A\}$, то $\forall i \in \{1, 2, \ldots, n-1\}$ должны иметь место тождества $M_i^{(V)}(U^{(i-1)}, A) \equiv 0$, отсюда:

$$\Delta\Psi_i^{(V)} = M_i'^{(V)}(U^{(i-1)}, A) \oplus \varphi_i^{(V)}(U^{(i-1)}, A) M_i''^{(V)}(U^{(i-1)}, A) = 0. \tag{4.11}$$

Совокупность функций $\{\Delta\Psi_i^{(V,e)}\}$ для всех $i \in \{1, \ldots, n-1\}$ называется *ядром параметрической обратимости* рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$.

Таким образом, для параметрически функциональной обратимости реализации рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$ необходимо и достаточно, чтобы $\forall i \in \{1, \ldots, n-1\}$ в *каноническом представлении переходных функций*

$$\begin{gathered}\Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = \\ = ex_i M_i^{(V)} \oplus eM_i'^{(V)}(U^{(i-1)}, A) \oplus x_i M_i''^{(V)}(U^{(i-1)}, A) \oplus M_i'''^{(V)}(U^{(i-1)}, A).\end{gathered} \tag{4.12}$$

$\forall i \in \{1, \ldots, n-1\}$ выполнялось тождество $M_i^{(V)} \equiv 0$ и условие (4.11).

Из (4.11) следует, что $M_i'^{(V)}(U^{(i-1)}, A) = \varphi_i^{(V)}(U^{(i-1)}, A) M_i''^{(V)}(U^{(i-1)}, A)$. Подставляя это выражение в каноническое представление (4.12), получаем требуемое определяющее рекуррентное соотношение:

$$u_i = \Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = [e\varphi_i^{(V)}(U^{(i-1)}, A) \oplus x_i] M_i''^{(V)}(U^{(i-1)}, A) \oplus M_i'''^{(V)}(U^{(i-1)}, A). \tag{4.13}$$

(т. е. $L_i^{(V)} = M_i''^{(V)}, L_i'^{(V)} = M_i'''^{(V)}$). Что и требовалось доказать.

Критерий 4.1

Алгоритм синтеза множества всех параметрически функционально-обратимых реализаций рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$ имеет следующий вид:

1. Для всех i от 1 до n произвольно в алгебраической нормальной форме фиксируем булевы функции $\{\varphi_i^{(V)}(U^{(i-1)}, A)\}$.
2. Для всех i от 1 до n–1 также произвольно фиксируем образующие полиномы $\{L_i^{(V)}(U^{(i-1)}, A)\}$, $\{L_i'^{(V)}(U^{(i-1)}, A)\}$.
3. По *определяющему рекуррентному соотношению* (4.13) в алгебраической нормальной форме формируем переходные булевы функции для всех i от 1 до n–1 и получаем одну из возможных реализаций модели $\mathbf{\Omega}''_{\sigma,e}$:

$$\Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = [e\varphi_i^{(V)}(U^{(i-1)}, A) \oplus x_i]L_i^{(V)}(U^{(i-1)}, A) \oplus L_i'^{(V)}(U^{(i-1)}, A).$$

Если в этом алгоритме использовать булевы функции вида:

$$\varphi_i^{(V)}(U^{(i-1)}, A) = a_i \oplus \nu_i \oplus \varphi_i''^{(V^{(i-1)})}(U^{(i-1)}, A^{(i-1)}),$$

то все модификации УПО будут не только параметрически функционально-обратимыми по X, но и не будут иметь коллизий по A, а сами УПО будут эффективно полными (см. определение 4.34).

Критерий 4.2

Для проверки *произвольной реализации рекурсивной модели* $\mathbf{\Omega}''_{\sigma,e}$ на параметрически функциональную обратимость достаточно:

1. Для всех $i \in \{1, \ldots, n–1\}$ полиномы переходных функций $\Psi_i^{(V,e)}(x_i, U^{(i-1)}, A)$ записать в каноническом виде, выделив члены, содержащие отдельно e и x_i (если такая запись невозможна, то обратимость в рамках модели $\mathbf{\Omega}''_{\sigma,e}$ отсутствует):

$$\Psi_i^{(V,e)}(x_i, U^{(i-1)}, A) = eM_i'^{(V)}(U^{(i-1)}, A) \oplus x_i M_i''^{(V)}(U^{(i-1)}, A) \oplus M_i'''^{(V)}(U^{(i-1)}, A).$$

2. Сформировать ядро параметрически функциональной обратимости $\{\Delta\Psi_i^{(V)}\}$ и проверить его на равенство нулю для каждого $i \in \{1, \ldots, n–1\}$:

$$\Delta\Psi_i^{(V)} = M_i'^{(V)}(U^{(i-1)}, A) \oplus \varphi_i^{(V)}(U^{(i-1)}, A)M_i''^{(V)}(U^{(i-1)}, A) = 0.$$

Следствие 4.8

Реализации модели $\mathbf{\Omega}''_{\sigma,e}$, для которых выполняются условия теоремы 4.4, образуют *параметрически функционально-обратимую рекурсивную модель* $\mathbf{\Omega}''_{\sigma,e_f} \subset \mathbf{\Omega}''_{\sigma,e}$, схема одноразрядного преобразователя которой представлена на рис. 4.5.

Рис. 4.5. Схема одноразрядного преобразователя θ_{e_f} для модели $\mathbf{\Omega}''_{\sigma,e_f}$ (см. рис. 4.4)

($\boldsymbol{e} = 0$ — "сложение", $\boldsymbol{e} = 1$ — "вычитание")

В соответствии с леммой 4.1 необходимым и достаточным условием параметрической обратимости реализаций УПО в рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$ является инвариантность значений переноса u_i относительно прямого и обратного преобразования. Функционально указанное условие обеспечивается в модели $\mathbf{\Omega}''_{\sigma,e_f}$, однако условие леммы 4.1 может быть обеспечено и за счет схемной реализации.

Теорема 4.6

Условие параметрически схемной обратимости для модели $\mathbf{\Omega}''_{\sigma,e}$ имеет следующий вид:

$$\forall i \in \{1, \ldots, n\} \;\; y_i = f_i^{(V)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A)$$

$$\text{и } \forall j \in \{1, \ldots, n-1\} \;\; u_j = \Psi_j^{(V,e)}(x_j, U^{(i-1)}, A) = \Psi_j^{(V)}(\delta_e(x_j, y_j), U^{(i-1)}, A).$$

Доказательство

В соответствии с результатами леммы 4.1 необходимо доказать, что значения функций перехода являются инвариантными относительно прямого ($e = 0$) и обратного преобразования ($e = 1$).

Для доказательства воспользуемся методом математической индукции. Рассмотрим обратное преобразование из условия теоремы при $i = 1$, т. е. вместо x_1 подставим y_1 и заменим $e = 0$ на $e = 1$:

$$y'_1 = y_1 \oplus \varphi_1^{(V)}(u_0, A) = x_1 \oplus \varphi_1^{(V)}(u_0, A) \oplus \varphi_1^{(V)}(u_0, A) = x_1,$$

$$u'_1 = \Psi_1^{(V)}(\delta_1(y_1, y'_1), u_0, A) = \Psi_1^{(V)}(\delta_1(y_1, x_1), u_0, A) = \Psi_1^{(V)}(x_1, u_0, A) = u_1,$$

т. е. условие инвариантности u_1 выполняется.

Предположим, что теорема справедлива для $i = k$. Докажем ее справедливость для $i = k + 1$:

$$y'_{k+1} = y_{k+1} \oplus \varphi_{k+1}^{(V)}(U'^{(k)}, A) = x_{k+1} \oplus \varphi_{k+1}^{(V)}(U^{(k)}, A) \oplus \varphi_{k+1}^{(V)}(U'^{(k)}, A) = x_{k+1},$$

т. к. $U^{(k)} = U'^{(k)}$.

Далее имеем

$$u'_{k+1} = \Psi_{k+1}^{(V,1)}(y_{k+1}, U'^{(k)}, A) = \Psi_{k+1}^{(V)}(\delta_1(y_{k+1}, y'_{k+1}), U'^{(k)}, A) =$$

$$= \Psi_{k+1}^{(V)}(y'_{k+1}, U'^{(k)}, A) = \Psi_{k+1}^{(V)}(x_{k+1}, U'^{(k)}, A) = \Psi_{k+1}^{(V)}(x_{k+1}, U^{(k)}, A) = u_{k+1}.$$

Что и требовалось доказать.

Следствие 4.9

Реализации модели $\mathbf{\Omega}'_{\sigma,e}$, для которых выполняются условия теоремы 4.5, образуют *параметрически схемно-обратимую рекурсивную модель* $\mathbf{\Omega}''_{\sigma,e_s} \subset \mathbf{\Omega}''_{\sigma,e}$, схема одноразрядного преобразователя которой представлена на рис. 4.6.

Рис. 4.6. Схема одноразрядного преобразователя θ_{e_s} из схемы (см. рис. 4.5) для модели $\mathbf{\Omega}''_{\sigma,e_s}$ ($e = 0$ — "сложение", $e = 1$ — "вычитание")

Параметрически функционально-обратимая модель $\mathbf{\Omega}''_{\sigma,e_f}$ принадлежат одновременно последовательной модели $\mathbf{S}_\sigma$ (4.2), и в соответствии с теоремой 4.3 для них должны существовать варианты параметрически схемно-обратимой реализации. В связи с этим примечательным является тот факт, что реализация таких УПО в соответствии с условием (4.5) приобретает тривиальный вид, а именно:

$$\forall i \in \{1, \ldots, n\} \;\; y_i = f_i^{(V,e)}(X^{(i)}, A) = x_i \oplus \varphi_i^{(V)}(W_e^{(i-1)}, A) = x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A),$$

так как в соответствии с леммой 4.1 и теоремой 4.4 функционально обеспечивается инвариантность переходных функций, т. е. $U^{(i-1)} = U'^{(i-1)}$ и, следовательно,

$$W_e^{(i-1)} = \delta_e(X^{(i-1)}, Y^{(i-1)}) = \delta_e(U^{(i-1)}, U'^{(i-1)}) = U^{(i-1)}.$$

С учетом тривиальности схемных реализаций модели $\mathbf{\Omega}''_{\sigma,e_f}$, справедливо следующее следствие.

Следствие 4.10

Параметрически обратимая рекурсивная модель $\mathbf{\Omega}''_{\sigma,e}$ состоит из объединения реализаций моделей $\mathbf{\Omega}''_{\sigma,e_f}$ и $\mathbf{\Omega}''_{\sigma,e_s}$, т. е. $\mathbf{\Omega}''_{\sigma,e} = \mathbf{\Omega}''_{\sigma,e_f} \cup \mathbf{\Omega}''_{\sigma,e_f}$ и $\mathbf{\Omega}''_{\sigma,e_f} \cap \mathbf{\Omega}''_{\sigma,e_s} = \varnothing$.

4.3.5. Синтез элементарных управляемых сумматоров

Если в качестве функций выхода и переходных функций в рекурсивной модели УПО $\mathbf{\Omega}''_{\sigma,e_f}$ выбрать функции, не зависящие от управляющего вектора V и имеющие следующий вид:

$$\forall i \in \{1, ..., n\}\ y_i = f_i^{(e)}(x_i, U^{(i-1)}, A) = x_i \oplus \varphi_i(u_{i-1}, A) = x_i \oplus \varphi_i(u_{i-1}, a_i) = \\ = x_i \oplus (a_i \oplus u_{i-1}),$$

$$\forall i \in \{1, ..., n-1\}\ u_i = \Psi_i^{(e)}(x_i, U^{(i-1)}, A) = \Psi_i^{(e)}(x_i, u_{i-1}, A) = (x_i \oplus e)(a_i \oplus u_{i-1}) \oplus (a_i u_{i-1}),$$

то получим детерминированный автомат, реализующий *сумматор по модулю* 2^n (рис. 4.7 и 4.8). Такой сумматор осуществляет сложение операндов X и A и удовлетворяет условию параметрически функциональной обратимости (см. критерий 4.2), так как:

$$\Psi_i^{(e)}(x_i, u_{i-1}, A) = e(a_i \oplus u_{i-1}) \oplus x_i(a_i \oplus u_{i-1}) \oplus a_i u_{i-1},$$

$$\Delta\Psi_i^{(V,e)} = (a_i \oplus u_{i-1})(a_i \oplus u_{i-1}) \oplus (a_i \oplus u_{i-1}) = (a_i \oplus u_{i-1} \oplus 1)(a_i \oplus u_{i-1}) = 0.$$

Рис. 4.7. Схема сумматора по модулю 2^n ($\boldsymbol{e} = 0$ — "сложение", $\boldsymbol{e} = 1$ — "вычитание")

Рис. 4.8. Схема одноразрядного сумматора σ_e ($\boldsymbol{e} = 0$ — "сложение", $\boldsymbol{e} = 1$ — "вычитание")

Данный алгоритм можно использовать в качестве *прототипа* для синтеза УПО, получивших название *элементарных управляемых сумматоров* (ЭУС). ЭУС управляются двоичными векторами $V \in \mathrm{GF}(2)^n$ $(m = n)$ и определяются по следующим правилам:

$$\forall i \in \{1, ..., n\}\ y_i = f_i^{(v_i, e)}(x_i, u_{i-1}, a_i) = x_i \oplus \varphi_i^{(v_i)}(u_{i-1}, a_i) = \\ = x_i \oplus [a_i \oplus (u_{i-1} * v_i)], \tag{4.14}$$

$$\forall i \in \{1, ..., n-1\}\ u_i = \Psi_i^{(v_i, e)}(x_i, u_{i-1}, a_i) = \\ = (x_i \oplus e)[a_i \oplus (u_{i-1} * v_i)] \oplus [a_i(u_{i-1} * v_i)], \tag{4.15}$$

где "*" — некоторая *булева операция* и u_0 — начальное значение, задающие тип *реализации* ЭУС. Значения вектора $V = (v_1, \ldots, v_n)$ определяют множество *модификаций* для каждой реализации ЭУС. Таким образом, последовательность $\{\gamma_i\} = \{(u_{i-1} * v_i)\}$ и элемент u_0, где u_i называются *битами переноса*, являются *синтезаторами* различных реализаций и модификаций ЭУС (рис. 4.9). В связи с этим элементарные сумматоры обозначаются как Σ_V("*", u_0).

Рис. 4.9. Схема элементарного управляемого сумматора Σ_V("*", u_0)$\in \boldsymbol{\Omega}''_{\sigma, e_f}$

($\boldsymbol{e} = 0$ — "сложение", $\boldsymbol{e} = 1$ — "вычитание")

Из способа построения ЭУС следует, что их можно трактовать как обычные сумматоры по модулю 2^n, в которых биты переноса "разрешаются" или "запрещаются" по правилам, определяемым управляющим вектором и видом операции "*". Данное представление ЭУС означает, что по некоторому псевдослучайному закону часть битов векторов X и A складывается в соответствии с операцией "$\oplus$", а часть — с правилами сложения по модулю 2^n (рис. 4.10). В указанном смысле ЭУС является комплексной недетерминированной операцией, которая статистически "варьируется" между двумя указанными выше операциями. При этом для любого значения элемента "*" эта операция в целом не будет групповой операцией, что считается положительным свойством при проектировании блочных шифров. Фактически третий управляющий операнд V задает (псевдослучайно выбирает) различные бинарные операции, поэтому ЭУС получили названия *двухместных управляемых операций*.

Рис. 4.10. Схема действия ЭУС Σ_V("$\wedge$", 1) при $\gamma_i = u_{i-1}v_i$

Кроме указанного выше способа представления ЭУС можно рассматривать как участок поточного шифрования из n битов, в рамках которого на сумму векторов $X \oplus A = \{x_i \oplus a_i\}$ "накладывается" некоторый поток элементов *(гамма)* $\{\gamma_i\} = \{(u_{i-1}*v_i)\}$, формируемый последовательно от младшего к старшему биту по рекуррентному соотношению (4.15).

В соответствии с условиями (4.14) и (4.15), а также со следствием 4.3 все типы ЭУС Σ_V("*", u_0) не имеют коллизий отдельно по X и по A, а по критерию 4.2 являются параметрически функционально-обратимыми, так как каноническое представление переходных функций $\{\Psi_i^{(v_i,e)}\}$ имеет вид:

$$\Psi_i^{(v_i,e)}(x_i, u_{i-1}, a_i) = e[a_i \oplus (u_{i-1}*v_i)] \oplus x_i[a_i \oplus (u_{i-1}*v_i)] \oplus a_i(u_{i-1}*v_i),$$

где

$$M_i'^{(V)}(U^{(i-1)}, A) = M_i''^{(V)}(U^{(i-1)}, A) = \varphi_i\,(u_{i-1}, a_i) = [a_i \oplus (u_{i-1} * v_i)],$$

$$M_i'''^{(V)}(U^{(i-1)}, A) = a_i(u_{i-1} * v_i)$$

и ядро параметрически функциональной обратимости тождественно равно 0:

$$\Delta\Psi_i^{(V,e)} = [a_i \oplus (u_{i-1}*v_i)][a_i \oplus (u_{i-1}*v_i)] \oplus [a_i \oplus (u_{i-1}*v_i)] =$$
$$= [a_i \oplus (u_{i-1}*v_i) \oplus 1)][a_i \oplus (u_{i-1}*v_i)] \equiv 0$$

Поскольку любая булева операция "*" ($u*v$) является отображением $\mathrm{GF}(2)^2 \to \mathrm{GF}(2)$, то во избежание путаницы в названиях этих операций формально их можно представить как *элементарные* полиномы следующего вида: $\alpha_0 \oplus \alpha_1 v \oplus \alpha_2 u \oplus \alpha_3 uv$, где $\boldsymbol{\alpha} = (\alpha_0, \alpha_1, \alpha_2, \alpha_3) \in \mathrm{GF}(2)^4$. Всего имеется $2^4 - 2 = 14$ булевых операций (число полиномов, существенно зависящих от переменных v и/или u). Булевы операции разделяются на *унарные* и *бинарные*, т. е. на операции, существенно зависящие от одной (v или u) или двух (v и u) переменных. Всего имеется 4 унарных и 10 бинарных операций. Поскольку вид ЭУС зависит только

от типа бинарной операции "*" и значения начального условия u_0, то всего имеется 20 различных реализаций ЭУС.

Образующими элементами ЭУС Σ_V("*", u_0), как следует из его определения, являются значения из гаммы $\{\gamma_i\} = \{(u_{i-1}*v_i)\}$. При этом теоретически возможно, что существуют пары {"*", u_0}, для которых в итоге формируются реализации ЭУС, содержащие эквивалентные модификации. Кроме того, не исключена возможность, что разные реализации ЭУС формируют одинаковые множества модификаций. В связи с этим обратимся к следующему утверждению.

Теорема 4.7

Если булева операция "*" бинарная ($u*v$ существенно зависит от каждого операнда u и v), то ЭУС Σ_V("*", u_0) является полным, т. е. реализует 2^n уникальных модификаций.

Доказательство

Возьмем произвольные V и $V' \neq V$ и некоторую пару $\{X, A\}$, тогда существует минимальное значение i такое, что $v_i \neq v'_i$.

Так как $\forall j < i \;\; v_j = v'_j$, то $u_{i-2} = u'_{i-2}$. Из условия (4.15) следует, что при заданных v_{i-1} и u_{i-2} функция $u_{i-1} = \Psi_{i-1}^{(v_{i-1},e)}(x_{i-1}, u_{i-2}, a_{i-1})$ существенно зависит от пары (x_{i-1}, a_{i-1}).

Отсюда существуют две пары (x_{i-1}, a_{i-1}) и (x'_{i-1}, a'_{i-1}) такие, что

$$\Psi_{i-1}^{(v_{i-1},e)}(x_{i-1}, u_{i-2}, a_{i-1}) \neq \Psi_{i-1}^{(v_{i-1},e)}(x'_{i-1}, u_{i-2}, a'_{i-1}).$$

Если элемент ($u_{i-1}*v_i$), синтезирующий ЭУС, существенно зависит от u_{i-1} и v_i, то для одной из указанных выше пар (x_{i-1}, a_{i-1}) или (x'_{i-1}, a'_{i-1}) получим $(u_{i-1}*v_i) \neq (u_{i-1}*v'_i)$. Так как в соответствии с (4.14) $y_i = x_i \oplus a_i \oplus (u_{i-1}*v_i)$ и $y'_i = x_i \oplus a_i \oplus (u_{i-1}*v'_i)$, то $y_i \neq y'_i$.

Следовательно, если $V \neq V'$, то можно построить пару векторов $\{X, A\}$ таких, что $\boldsymbol{G}_V(X, A) \neq \boldsymbol{G}_{V'}(X, A)$. Таким образом, каждому из 2^n значений управляющего вектора V соответствует своя уникальная модификация. Что и требовалось доказать.

Теорема 4.8

При произвольном фиксированном начальном условии u_0 каждый ЭУС Σ_V("*", u_0) реализует различные множества модификаций, т. е. в соответствии с наличием 10 булевых операций имеется два множества {Σ_V("*", 1)} и {Σ_V("*", 0)} по 10 уникальных реализаций ЭУС в каждом.

Доказательство

Очевидно, что если операции "*" и "°" различны, то $\exists$ пара $\{u, v\}$ такая, что $u*v \neq u°v$. Если для некоторого значения переходных функций u_i и u'_i для ЭУС Σ_V("*", u_0) и Σ_V("°", u_0) выполняется $u_i = u$ и $u'_i = u$, то при $v_{i+1} = v \Rightarrow y_{i+1} = x_{i+1} \oplus a_{i+1} \oplus (u_i*v_{i+1}) = x_{i+1} \oplus a_{i+1} \oplus (u*v)$ и $y'_{i+1} = x_{i+1} \oplus a_{i+1} \oplus (u'_i °v_{i+1}) = x_i \oplus a_i \oplus (u°v) \Rightarrow \Rightarrow y_i \neq y'_i$, т. е. ЭУС Σ_V("*", u_0) и Σ_V("°", u_0) не эквивалентны. Если показать, что вышеуказанное условие существования пары $u_i = u$ и $u'_i = u$ всегда выполняется, то теорема доказана. Предположим, что данное условие не выполняется, тогда возможно два варианта: (1) $\forall i \in \{1, \ldots, n-1\}$ $u'_i = \bar{u}_i$ или (2) $\forall i \in \{1, ..., n-1\}$ $u_i = u'_i = \bar{u}$. В первом случае из (4.15) следует:

$$u_i \oplus u'_i = (x_i \oplus e)[(u_{i-1}*v_i) \oplus (u'_{i-1} °v_i)] \oplus a_i[(u_{i-1}*v_i) \oplus (u'_{i-1} °v_i)] =$$
$$= [(u_{i-1}*v_i) \oplus (u'_{i-1} °v_i)](x_i \oplus e \oplus a_i) \equiv 1,$$

что невозможно, так как второй сомножитель $(x_i \oplus e \oplus a_i)$ зависит от параметра e и может принимать значение 0. Второй из указанных случаев невозможен, так как из (4.15) следует, что последовательности u_i не являются константами. Что и требовалось доказать.

Рассмотрим некоторые вероятностные свойства ЭУС Σ_V("*", u_0). Прежде всего, отметим, что в соответствии с результатами следствия 4.6 все типы ЭУС обеспечивают *равновероятность* появления 0 или 1 на выходе операции.

В ЭУС Σ_V("*", u_0) процесс преобразования осуществляется по формуле: $\forall i \in \{1, \ldots, n\}$ $y_i = x_i \oplus a_i \oplus (u_{i-1}*v_i)$, что соответствует, как упоминалось ранее, "наложению" гаммы $\gamma_i = (u_{i-1}*v_i)$ на сумму "$x_i \oplus a_i$". Рассмотрим свойство образующей для γ_i рекуррентной последовательности битов переноса $\{u_i\}$.

Вычислим вероятность $\mathbf{Pr}\{u_i = 1\}$. Из формулы (4.15) следует, что $u_i = 1$ в таких случаях: $a_i = 1$, $u_{i-1}*v_i = 1$; $a_i = 1$, $u_{i-1}*v_i = 0$, $x_i \oplus e = 1$; $a_i = 0$, $u_{i-1}*v_i = 1$, $x_i \oplus e = 1$.

Из предположения, что $\mathbf{Pr}\{a_i = 1\} = \mathbf{Pr}\{a_i = 0\} = 1/2$ следует:

$$\mathbf{Pr}\{u_i = 1\} = (1/2)\{\mathbf{Pr}\{u_{i-1}*v_i = 1\} + \mathbf{Pr}\{u_{i-1}*v_i = 0\}\mathbf{Pr}\{x_i \oplus e = 1\} + \\ + \mathbf{Pr}\{u_{i-1}*v_i = 1\}\mathbf{Pr}\{x_i \oplus e = 1\}\}. \tag{4.16}$$

Если рассмотреть в качестве примера ЭУС Σ_V("$\vee$", 0), то

$$\mathbf{Pr}\{u_{i-1} \vee v_i = 1\} = 1 - \mathbf{Pr}\{u_{i-1} = 0\}\mathbf{Pr}\{v_i = 0\}.$$

Учитывая, что $\mathbf{Pr}\{v_i = 1\} = \mathbf{Pr}\{v_i = 0\} = 1/2$, имеем:

$$\mathbf{Pr}\{u_{i-1} \vee v_i = 1\} = 1 - (1/2)\mathbf{Pr}\{u_{i-1} = 0\} = 1/2 + (1/2)\mathbf{Pr}\{u_{i-1} = 1\}$$
$$\text{и } \mathbf{Pr}\{u_{i-1} \vee v_i = 0\} = 1/2 - (1/2)\mathbf{Pr}\{u_{i-1} = 1\}.$$

Подставляя полученные значения вероятностей в формулу (4.16), получим следующее рекуррентное соотношение:

$$\mathbf{Pr}\{u_i = 1\} = 1/2 + (1/2)\mathbf{Pr}\{x_i \oplus e = 1\} + (1/2)\mathbf{Pr}\{u_{i-1} = 1\}.$$

При $\mathbf{Pr}\{x_i = 1\} = \mathbf{Pr}\{x_i = 0\} = 1/2$ имеем: $\mathbf{Pr}\{u_i = 1\} = 1/2 + (1/4)\mathbf{Pr}\{u_{i-1} = 1\}$.

Из решения данного рекуррентного соотношения следует, что при $u_0 = 0$

$$\mathbf{Pr}\{u_i = 1\} = 2/3(1 - 1/(4^i)) \text{ и } \lim_{i\to\infty} \mathbf{Pr}\{u_i = 1\} = 2/3,$$

т. е. для данного типа ЭУС ("$*$" = "$\vee$" и $u_0 = 0$) имеется преобладание единичного бита переноса u_i над нулевым битом.

Аналогично для других типов ЭУС, синтезируемых различными булевыми операциями "$*$" и начальными условиями u_0, можно получить *таблицу предельных вероятностей* $\lim_{i\to\infty} \mathbf{Pr}\{u_i = 1\}$ (табл. 4.1). Доказательство предлагается провести читателю в качестве упражнения.

Таблица 4.1. *Предельные вероятности* $\lim_{i\to\infty} \mathbf{Pr}\{u_i = 1\}$

$*$ α	$\vee$ 0,1,1,1	$\overline{\vee}$ 1,1,1,1	$\wedge$ 0,0,0,1	$\overline{\wedge}$ 1,0,0,1	$\oplus$ 0,1,1,0	$\overline{\oplus}$ 1,1,1,0	$\rightarrow$ 1,0,1,1	$\overline{\rightarrow}$ 0,0,1,1	$\leftarrow$ 1,1,0,1	$\overline{\leftarrow}$ 0,1,0,1
$u_0 = 0$	2/3	2/5	1/3	3/5	1/2	1/2	3/5	1/3	2/3	2/5
$u_0 = 1$	2/3	2/5	1/3	3/5	1/2	1/2	3/5	1/3	2/3	2/5

Исходя из изложенного, переходные функции ЭУС с синтезаторами вида $(u_{i-1} \oplus v_i)$ или $(u_{i-1} \overline{\oplus} v_i)$ обладают наилучшими вероятностными свойствами. Такие сумматоры кроме параметрически функциональной обратимости по X и отсутствия коллизий по A являются в соответствии со следствием 4.4 эффективно полными.

К общим недостаткам всех ЭУС можно отнести их недостаточно хорошие дифференциальные свойства и высокую линейность.

По способу построения ЭУС занимают самое нижнее положение в классификации рекурсивной модели УПО. Очевидно, что, рассматривая в соответствии с критерием 4.1 более сложные виды образующих полиномов $\{ M_i''^{(V)} \}$, $\{ M_i'''^{(V)} \}$ и функций $\{ \varphi_i^{(V)} \}$, удовлетворяющих условию $\Delta\Psi_i^{(V,e)} = 0$, можно синтезировать новые семейства управляемых сумматоров, обладающих различными свойствами. Такие сумматоры получили название *комбинированных управляемых сумматоров* (КУС).

4.3.6. Небиективная модель управляемой подстановочной операции

В предыдущих разделах основное внимание уделялось биективным моделям УПО как базовым криптографическим примитивам блочных шифров. Требование биективности накладывает определенные ограничения на другие криптографические свойства этих преобразований, в частности, на их общую нелинейность и дифференциальные характеристики. Однако в ряде случаев, например при формировании раундовой функции в схеме Фейстеля, наличие биективности желательно, но не является обязательным. В этом случае бывает целесообразным использование небиективных преобразований небольшой сложности, обеспечивающих повышенные параметры нелинейности и распространения ошибок, но незначительно сужающих пространство двоичных векторов на выходе преобразования.

Такого рода преобразования достаточно эффективно реализуются в рамках *последовательной небиективной модели УПО* $\overline{\mathbf{S}}_\sigma \subset \mathbf{S}_\sigma$, формируемой *циклическим сдвигом аргументов* образующих булевых функций в соответствии со следующими правилами:

$$\forall i \in \{1, ..., n\} \;\; f_i^{(V)}(X, A) = f_i^{(V)}(X^{(i, i-k_i+1)}, A),$$

где $k_i \le n$, $X^{(i,\, i-k_i+1)} = (x_{i-k_i+1}, x_{i-k_i+2}, \ldots, x_i)$, и арифметические действия с индексами выполняются по модулю n.

Практические эксперименты по реализации указанной модели показали, что такие криптографические примитивы сужают пространство двоичных векторов X менее чем в два раза (в пределах 15—40%). Доказательство этого факта, а также рассмотрение других свойств небиективной последовательной модели $\overline{\mathbf{S}}_\sigma$ выходят за рамки данной книги. Прообразом таких преобразований служат широко известные в физике и математике клеточные автоматы [137].

С учетом результатов, полученных в *разд. 4.3.1—4.3.5*, имеется следующая схема моделей управляемых подстановочных операций (рис. 4.11).

4.3.7. Синтез комбинированных управляемых сумматоров и управляемых схемных преобразователей

Комбинированные управляемые сумматоры (КУС) и управляемые схемные преобразователи (УСП) (см. определение 4.42) осуществляют более гибкое применение механизмов управления в подстановочных операциях и обеспечивают повышенную нелинейность и другие полезные качества криптографических примитивов по сравнению с элементарными управляемыми сумматорами (ЭУС). В указанном смысле они являются наиболее перспективными элементами, обеспечивающими существенное повышение стойкости создаваемых на их основе

Рис. 4.11. Схема классификации моделей управляемых подстановочных операций

блочных шифров к современным криптографическим атакам. Особенно целесообразным представляется создание КУС в рамках параметрически функционально-обратимой модели $\mathbf{\Omega}''_{\sigma,e_f}$ и в параметрически схемно-обратимой рекурсивной модели $\mathbf{\Omega}''_{\sigma,e_s}$, а УСП — в биективной последовательной модели $\overline{\mathbf{S}}_\sigma$ и небиективной последовательной модели $\overline{\mathbf{S}}_\sigma$.

При синтезе различных УПО следует иметь в виду, что и типы функций выхода $f_i^{(V,e)}$, и типы переходных функций $\Psi_i^{(V,e)}$ могут быть, вообще говоря, различными, что обеспечивает достаточно гибкий механизм усложнения общего преобразования, осуществляемого УПО. Однако задачи, связанные со снижением сложности и уменьшением времени реализации преобразований, могут оказаться доминирующими. Решение указанных проблем осуществляется за счет:

- ❒ полной или частичной унификации функций выхода и переходных функций;
- ❒ ограничения числа входящих в них переменных.

Наборы таких функций называются *прототипами* образующих УПО булевых функций.

Необходимые математические свойства КУС и УСП, относящиеся к уникальности модификаций, полноте, отсутствию коллизий, параметрической обратимости, регулярности и др. могут быть достигнуты в результате проектирования этих операций в соответствии с результатами, полученными в *разд. 4.3.2—4.3.5*.

В качестве базовых оптимизированных схем, учитывающих различные аспекты требований к УПО, можно предложить следующие схемы:

1. В случае необходимости высокоскоростных параметрически обратимых преобразований небольшой сложности, целесообразно использование ЭУС Σ_V("*", u_0) модели $\mathbf{\Omega}''_{\sigma,e_f}$ или элементарных УСП модели $\mathbf{S}'_{\sigma,e_s}$.

2. При необходимости проведения быстрого нелинейного биективного или небиективного преобразования, не требующего обратимости, например в раундовой функции F схемы Фейстеля, предлагается использование УСП моделей $\mathbf{S}'_\sigma$ и $\overline{\mathbf{S}}_\sigma$ с усеченными и унифицированными функциями $\varphi_i^{(V)}$ следующего вида:

   $$\forall i \in \{1, \ldots, n\}\ y_i = f_i^{(V,e)}(X^{(i)}, A) = x_i \oplus \varphi_i^{(V)}(X^{(i-1)}, A) = x_i \oplus \varphi^{(V)}(X^{(i-1,\, i-k)}, A),$$

   где $X^{(i-1,\, i-k)} = (x_{i-k}, x_{i-k+1}, \ldots, x_{i-1})$ и для модели $\mathbf{S}'_\sigma$ — $X^{(0,\, 1-k)} = (x_0, x_{-1}, \ldots, x_{1-k})$ — начальные условия или k значений, полученных в результате расширяющего преобразования $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^{n+k}$ вектора X.

   Указанные реализации моделей $\mathbf{S}'_\sigma$ и $\overline{\mathbf{S}}_\sigma$ не требуют схемотехнического решения и имеют высокое быстродействие, поскольку все преобразования

$y_i = x_i \oplus \varphi^{(V)}(X^{(i-1)}, A)$ могут выполняться одновременно в параллельном режиме *(см. разд. 4.5.2)*.

3. В случае необходимости обратимого преобразования, обеспечивающего существенное повышение стойкости блочных шифров к современным методам математического анализа, предлагается использование УСП модели $\mathbf{S}'_{\sigma,e_S}$ и КУС моделей $\mathbf{\Omega}''_{\sigma,e_f}$ и $\mathbf{\Omega}''_{\sigma,e_s}$ с усеченными и унифицированными функциями выхода и переходными функциями следующего вида:

   - (модель $\mathbf{S}'_{\sigma,e_s}$): $\forall i \in \{1, ..., n\}\ y_i = x_i \oplus \varphi^{(V)}(\delta_e(X^{(i-1,\, i-k)}, Y^{(i-1,\, i-k)}), A)$;
   - (модель $\mathbf{\Omega}''_{\sigma,e_f}$): $\forall i \in \{1, ..., n\}\ y_i = x_i \oplus \varphi^{(V)}(U^{(i-1,\, i-k)}, A)$,

     $\forall j \in \{1, ..., n-1\}\ u_j = \Psi^{(V,\, e)}(x_j, U^{(j-1,\, j-l)}, A)$;
   - (модель $\mathbf{\Omega}''_{\sigma,e_s}$): $\forall i \in \{1, ..., n\}\ y_i = x_i \oplus \varphi^{(V)}(U^{(i-1,\, i-k)}, A)$,

     $\forall j \in \{1, ..., n-1\}\ u_j = \Psi^{(V)}(\delta_e(x_j, y_j), U^{(j-1,\, j-l)}, A)$,

   где $U^{(0,\, 1-\max\{k,\, l\})}$ и $X^{(0,\, 1-k)}$ — начальные условия или значения, полученные в результате "расширяющего" преобразования вектора X.

В заключение раздела следует обратить внимание на то обстоятельство, что в целях увеличения скорости выполнения КУС и УСП, как правило, применяются упрощающие принципы формирования таких операций. В связи с этим необходимо иметь в виду, что лишь использование в УПО более сложных и не полностью унифицированных образующих функций высокой степени $\mathbf{deg}(f)$ позволяет в полной мере осуществлять высоко нелинейные подстановочные преобразования. В упрощенном же виде такие КУС и УСП могут осуществлять только скоростные вспомогательные операции, требующие дальнейших преобразований в сочетании с другими криптографическими примитивами.

4.3.8. Свойства булевых функций, образующих управляемые подстановочные операции

Кроме указанных выше структурных свойств КУС и УСП существенное значение имеют специальные математические свойства самих булевых функций, образующих данные УПО и во многом определяющих стойкость преобразований к различным методам криптоанализа. Эти свойства в полной мере определяются общими требованиями, предъявляемыми к криптографическим примитивам и образующим их булевым функциям *(см. разд. 4.2.11)*. К таким свойствам можно отнести следующие характеристики:

- ❒ степень нелинейности функции $\mathbf{N}(f)$ или $\mathbf{N}^*(f)$ *(см. разд. 4.2.4 и 4.2.9)*;
- ❒ алгебраическую степень булевой функции $\mathbf{deg}(f)$ *(см. разд. 4.2.2)*;

- соответствие критерию строгого лавинного эффекта (**SAC**(k)) или критерию распространения изменений (**PC**(l, k) *(см. разд. 4.2.5 и 4.2.6)*;
- корреляционную иммунность (**CI**(l), **CI**(l, X^*)) или (n, m, k)-эластичность $\boldsymbol{R}(n, m, k)$ *(см. разд. 4.2.7)*.

Указанные выше свойства находятся во взаимной зависимости и в определенной степени могут быть противоречивыми. Так, например, самые нелинейные из булевых функций — бент-функции *(см. разд. 4.2.8)* удовлетворяют критерию **PC**(n), но могут не удовлетворять критерию **PC**(l, k). Вместе с тем, бент-функции не обладают и совершенными корреляционно-иммунными показателями, поскольку в соответствии с определением для них всегда выполняется условие $\forall \boldsymbol{\alpha} \in \mathrm{GF}(2)^n\ U^*_{\alpha}(f) = \pm 2^{n/2}$, т. е. эти функции имеют корреляцию с каждой линейной булевой функцией. Кроме того, бент-функции могут содержать только четное число переменных n и имеют ограничение на степень алгебраической нормальной формы $\mathbf{deg}(f) \le n/2$.

Отсюда следует, что невозможно построить идеальные образующие булевы функции, в максимальной степени удовлетворяющие всем перечисленным свойствам. Таким образом, при выборе таких функций для УПО необходимо учитывать конкретные потребности и особенности проектируемых блочных шифров. Значимость тех или иных свойств образующих булевых функций становится ясной в результате анализа влияния УПО на стойкость шифров к различным методам криптоанализа, в частности, к наиболее мощным из них — дифференциальному и линейному криптоанализу.

Вместе с тем, следует признать тот факт, что бент-функции играют особую роль в криптографических преобразованиях, поскольку являются максимально нелинейными, обладают минимальной общей автокорреляцией и одинаково низкой корреляцией с каждой линейной булевой функцией. В связи с этим представляется целесообразным более подробно остановиться на использовании бент-функций в качестве базовых элементов при конструировании УПО.

4.3.9. Оптимизация управляемых подстановочных операций на базе специальных булевых функций

В вопросах оптимального построения образующих УПО функций важное значение имеет тот факт, что в соответствии с утверждением 4.17 *(см. разд. 4.2.9)*, если булева функция $f(X)$ ($X \in \mathrm{GF}(2)^n$) представляется в виде $f(X) = h_{aff}(X_1) \oplus h(X_2)$, где $X_1 \in \mathrm{GF}(2)^l$, $X_2 \in \mathrm{GF}(2)^{n-l}$ и $h_{aff}(X_1)$ является аффинной функцией, то максимальная нелинейность для $f(X)$ достигается при $h(X_2) = h_{bent}(X_2)$ и определяется формулой $\mathbf{N}(h) = 2^{n-1} - 2^{\frac{1}{2}(n+l)-1}$.

Примечательным фактом является то обстоятельство, что функции выхода для КУС биективной модели $\boldsymbol{\Omega}'_{\sigma}$ могут иметь только следующие четыре вида *(см. разд. 4.3.3 и 4.3.4)*:

- $f_i^{(V)} = x_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A)$;
- $f_i^{(V)} = x_i \oplus a_i \oplus \varphi_i^{(V)}(U^{(i-1)}, A^{(i-1)})$;
- $f_i^{(V)} = x_i \oplus v_i \oplus \varphi_i^{(V^{(i-1)})}(U^{(i-1)}, A)$;
- $f_i^{(V)} = x_i \oplus a_i \oplus v_i \oplus \varphi_i^{(V^{(i-1)})}(U^{(i-1)}, A^{(i-1)})$.

Для УСП биективной модели $\mathbf{S}'_\sigma$ — это, соответственно, образующие функции:

- $f_i^{(V)} = x_i \oplus \varphi_i^{(V)}(X^{(i-1)}, A)$;
- $f_i^{(V)} = x_i \oplus a_i \oplus \varphi_i^{(V)}(X^{(i-1)}, A^{(i-1)})$;
- $f_i^{(V)} = x_i \oplus v_i \oplus \varphi_i^{(V^{(i-1)})}(X^{(i-1)}, A)$;
- $f_i^{(V)} = x_i \oplus a_i \oplus v_i \oplus \varphi_i^{(V^{(i-1)})}(X^{(i-1)}, A^{(i-1)})$.

Поскольку прототипы образующих функций биективных моделей $\mathbf{S}'_\sigma$ и $\mathbf{\Omega}'_\sigma$ имеют упомянутый выше вид из утверждения 4.17, т. е. $f(X) = h_{aff}(X_1) \oplus \varphi(X_2)$, и параметр l принимает минимальные значения ($l = 1$, $l = 2$ и $l = 3$), то при $\varphi(X_2) = \varphi_{bent}(X_2)$ функции $f(X)$ будут иметь *максимальную нелинейность в рамках этих моделей*. Кроме того, функции указанного вида имеют характеристики распространения изменений $\mathbf{PC}(n)$, близкие к оптимальным.

Действительно, функция $h(X) = h(X_1 \| X_2) = h_{aff}(X_1) \oplus h_{bent}(X_2)$ удовлетворяет критерию $\mathbf{PC}(n)$ на всем $\mathrm{GF}(2)^n$, кроме множества векторов $\mathrm{GF}(2)^l \times \{0\}_{n-l}$, где $\{0\}_{n-l}$ — нулевой вектор размерности $n - l$. Следовательно, критерий $\mathbf{PC}(n)$ выполняется на множестве из $2^n - 2^l$ элементов, что при $l = 1$ (2 или 3) близко к максимальному значению 2^n. Указанный выше подход в проектировании УПО является главной предпосылкой для повышения стойкости блочных шифров на базе УПО прежде всего к линейному криптоанализу и уже во вторую очередь к дифференциальному криптоанализу.

При оптимизации УПО в первую очередь, с точки зрения дифференциального криптоанализа, необходимо соблюдать несколько иные принципы. В качестве булевых функций φ_i более перспективны функции, удовлетворяющие критерию $\mathbf{PC}(l, k)$ высокой степени l и порядка k, которые, в соответствии с результатами *разд. 4.2.6*, помимо высокой нелинейности обладают хорошими автокорреляционными параметрами при фиксированных значениях части переменных. Особенностью использования таких функций является то обстоятельство, что в силу своих свойств они обеспечивают минимальную зависимость между компонентами лавинных векторов *(см. разд. 4.2.6)* функции $\boldsymbol{G}_V(X, A) = \{f_1^{(V,e)}, f_2^{(V,e)}, ..., f_m^{(V,e)}\}$, реализующей УПО. Наличие указанного свойства играет определяющую роль в вопросах повышения стойкости к дифференциальному криптоанализу за счет

усреднения вероятностей $\mathbf{Pr}_X(\boldsymbol{G}_{V\oplus\Delta V}(X\oplus\Delta X, A\oplus\Delta A) \oplus \boldsymbol{G}_V(X, A) = \beta)$ (см. утверждение 4.9).

Принципиальная особенность оптимизации КУС, существенно влияющая на качество преобразования, в отличие от УСП состоит в необходимости обеспечения равновероятности и независимости компонентов векторов $U^{(i-1)}$, образуемых переходными функциями $\{\Psi_j^{(V,e)}\}$ и являющихся аргументами функций выхода $\{f_i^{(V,e)}\}$. Необходимость данного условия связана с тем, что высокая нелинейность функций выхода, аргументами которых являются переходные функции $\{\Psi_j^{(V,e)}\}$, в полной мере может быть гарантирована лишь при наличии равновероятности и независимости значений переходных функций. Указанные выше свойства в соответствии с результатами *разд. 4.2.7* обеспечиваются за счет использования сбалансированных корреляционно-иммунных относительно множеств $U^{(j-1)}$ или эластичных булевых функций $u_j = \Psi_j^{(V,e)}(x_j, U^{(j-1)}, A)$. Таким образом, при проектировании переходных функций для КУС базовым направлением является их выбор из многообразия корреляционно-иммунных булевых функций, обладающих высокой нелинейностью и повышенной степенью алгебраической нормальной формы $\mathbf{deg}(f)$. Примеры бент-функций, корреляционно-иммунных функций и функций, имеющих высокие показатели распространения изменений, пригодных для применения в качестве образующих для КУС и УСП, представлены в *разд. 4.2.6—4.2.8*. Однако при создании параметрически функционально-обратимых КУС модели $\boldsymbol{\Omega}''_{\sigma,e_f}$ необходимо учитывать особенность, связанную с тем, что в таких моделях переходные функции аналитически выражаются через функции выхода в соответствии с требованиями критерия 4.1 и 4.2, что существенно ограничивает возможности синтеза переходных функций с высокими корреляционно-иммунными параметрами.

4.3.10. Общие замечания относительно управляемых подстановочных операций

Принимая во внимание изложенные в данной главе достоинства УПО, необходимо все же четко определять границы их возможностей как нелинейных криптографических примитивов. Эти ограничения связаны с тем обстоятельством, что, несмотря на достаточно высокую сложность преобразований, УПО по своей природе имеют *двойственный характер*, т. е. занимают промежуточное положение между двумя "полюсами" — нелинейными математическими операциями и **S**-блоками большого размера. Степень приближения к тому или иному "полюсу" зависит от модели УПО, разнообразия состава и сложности прототипов образующих УПО булевых функций. Таким образом, при синтезе УПО возможно варьирование их различных криптографических характеристик от низких до дос-

таточно высоких значений. С учетом двойственности природы УПО возможности этих криптографических примитивов ограничены в соответствии со следующими положениями.

- При оценке нелинейности УПО следует иметь в виду, что в силу метода построения образующие УПО булевы функции $\{ f_i^{(V,e)} \}$ либо не зависят от всех компонентов преобразуемого вектора X (модель $\mathbf{S}'_\sigma$), либо такая зависимость обеспечивается лишь в рекурсивном виде (модель $\mathbf{\Omega}'_\sigma$), поэтому результирующее преобразование $\boldsymbol{G_V}(X, A)$ не может достигать максимально возможных значений нелинейности и распространения изменений *(см. разд. 4.2.8 и 4.2.10)*, доступных только векторным бент-функциям, имеющим сложную реализацию и ряд криптографических недостатков.
- При небольшом количестве различных образующих функций $\{ f_i^{(V,e)} \}$ в линейных комбинациях вида $\bigoplus_{i=1}^{m} \alpha_i f_i(X)$ иногда имеет место поглощение некоторых членов, что в соответствии с определениями 4.15 и 4.16 может привести к уменьшению показателей общей и средней нелинейности УПО.
- Лавинный эффект преобразования $\boldsymbol{G_V}(X, A)$, задающего УПО, распространяет изменения только в пределах участка выходного вектора, равного по длине количеству переменных в соответствующих образующих булевых функциях $f_i^{(V,e)}$.
- Между компонентами внутри каждого лавинного вектора преобразования $\boldsymbol{G_V}(X, A)$ *(см. разд. 4.2.6)* в большинстве случаев имеется определенная зависимость, что может снижать равномерность распределения их вероятностей и тем самым ухудшать дифференциальные свойства векторного преобразования в целом.

Указанные выше недостатки УПО во многом могли бы нивелировать их преимущества. Однако это было бы реально только в случае создания специальных **S**-блоков, имеющих размеры 32—128 битов, что практически невозможно ввиду высокой математической сложности таких преобразований. Действительно, высокоскоростная и эффективная реализация **S**-блоков в блочных шифрах возможна лишь при их табличном задании или использовании несложных функциональных представлений, что ограничивает размеры **S**-блоков 4—13 битами. Такие преобразования по нелинейности соизмеримы с преобразованиями, осуществляемыми образующими УПО булевыми функциями, но уступают им по другим криптографическим параметрам. Кроме того, несмотря на высокие показатели нелинейности и лавинного эффекта в пределах небольшой длины одного **S**-блока, результирующее векторное преобразование, состоящее из нескольких параллельно применяемых **S**-блоков, не может иметь высоких криптографических характеристик. Таким образом, если рассматривать 32—128-битное преоб-

разование блока данных в целом, то криптографические свойства практически применимых небольших **S**-блоков просто не могут быть оптимальными. Если же учитывать упомянутые в *разд. 4.2.3* и *4.2.4* преимущества УПО и недостатки **S**-блоков и по ряду других параметров, то можно сделать вывод, что первые криптографические примитивы во многих случаях оказываются более эффективными по сравнению со вторыми.

В заключение этого раздела необходимо напомнить, что стратегическим направлением в проектировании управляемых подстановочных операций является создание высокоскоростных нелинейных преобразований невысокой сложности, пригодных для аппаратной и, в ряде случаев, программной реализации. Данный подход не позволяет в полной мере использовать максимальные возможности УПО как нелинейных криптографических примитивов и требует регулирования по линии "скорость-качество". В связи с этим, как уже отмечалось в *разд.4.3.7*, для повышения эффективности общего преобразования в рамках одного раунда схемы Фейстеля и других алгоритмов блочного шифрования необходимо применять управляемые подстановочные операции в гибком сочетании с другими криптографическими примитивами.

4.3.11. Выводы

В заключение *разд. 4.3* можно сделать следующие выводы:

- Результаты данного раздела являются иллюстрацией нестандартного подхода в проектировании новых криптографических примитивов. Предлагаемые управляемые подстановочные операции могут служить определенной альтернативой традиционным **S**-блокам и стандартным математическим операциям.
- В рамках проведенного моделирования различных типов управляемых подстановочных операций выделена последовательная модель, обеспечивающая следующие свойства новых криптографических примитивов:
  - биективность отображений, в том числе параметрическая обратимость, что позволяет реализовывать процесс прямого и обратного преобразования без изменения схемы блочного шифра;
  - регулярность отображений, что обеспечивает стойкость к прямым статистическим атакам;
  - свойства уникальности, полноты и, в том числе, эффективной полноты преобразований по ключевым элементам, а также отсутствие коллизий по преобразуемым данным;
  - высокие корреляционно-иммунные показатели преобразований, определяющие низкую статистическую независимость значений каждой функций выхода от значений аргументов на входе управляемой подстановочной операции.

- В соответствии с проведенной классификацией моделей управляемых подстановочных операций выделены два основных класса моделей:
  - параметрически схемно-обратимые;
  - параметрически функционально-обратимые.
- Определено, что в рамках параметрически схемно-обратимой модели наиболее эффективной являются ее реализации в виде управляемого схемного преобразователя, при этом для раундовой функции схемы Фейстеля особенно перспективны преобразователи, выполняющие только высокоскоростные прямые преобразования.
- Отмечено, что среди параметрически функционально-обратимых управляемых подстановочных операций особое место занимают параметрически обратимая рекурсивная модель, в рамках которой определены общие условия параметрической обратимости операций и проведена классификация элементарных и комбинированных управляемых сумматоров, реализующих широкий спектр нелинейных преобразований.
- Среди разновидностей управляемых схемных преобразователей и комбинированных управляемых сумматоров выделены операции, наиболее подходящие с практической точки зрения, и сформулированы основные принципы оптимизации этих операций на основе булевых функций специального типа.
- Блочные шифры на базе управляемых подстановочных операций имеют потенциальные возможности повышения стойкости блочных шифров к дифференциальному и линейному криптоанализу за счет использования в качестве функций выхода булевых функций на базе бент-функций и функций, удовлетворяющих критерию $\mathbf{PC}(l, k)$, и в качестве переходных функций — сбалансированных корреляционно-иммунных функций $\mathbf{CI}(l)$ высокой нелинейности.
- Управляемые подстановочные операции, предназначенные для скоростной аппаратной реализации, имеют регулируемые в определенных границах параметры нелинейности, корреляционной иммунности и распространения изменений.
- Для обеспечения высокой эффективности блочных шифров в рамках одного раунда преобразований целесообразно использование управляемых подстановочных операций в сочетании с другими управляемыми преобразованиями, в частности, с блоками управляемых перестановок.

Для более полной оценки принципов эффективного синтеза управляемых подстановочных операций в *разд. 4.4* рассматриваются механизмы влияния этих операций на стойкость блочных шифров к наиболее мощным криптографическим атакам.

4.4. Влияние управляемых подстановочных операций на стойкость блочных шифров к криптоанализу

4.4.1. Общие сведения об аналитических и технических атаках на блочные шифры

При проектировании блочных шифров, основанных на использовании управляемых подстановочных операций (УПО) и других криптографических примитивов, первостепенное значение имеет обеспечение стойкости этих шифров к аналитическим и техническим криптоатакам. Практика последних лет показала, что наиболее мощными аналитическими методами криптоанализа шифров являются *дифференциальный (разностный) криптоанализ* (ДКА) [44] и *линейный криптоанализ* (ЛКА) [86], а также производные от них методы, которые и определяют в основном степень стойкости блочных алгоритмов. К техническим атакам на блочные шифры можно отнести *метод генерации аппаратных ошибок* [46] и *метод замера мощности излучения*. [79].

Применение в блочных шифрах новых криптографических примитивов — управляемых перестановочных и подстановочных операций — создает целый ряд предпосылок для создания высокоскоростных алгоритмов, стойких к современным методам криптоанализа. Предпосылки стойкости таких шифров связаны с особенностями построения управляемых операций, определенными механизмами их управления, а также со способами применения этих операций в различных криптосхемах.

Анализ предлагаемых блочных шифров на базе управляемых операций показал их высокую стойкость к классическим методам ДКА и ЛКА как главных критериев криптостойкости алгоритмов. Кроме указанных методов, блочные шифры на основе новых операций показали стойкость и к другим методам криптоанализа.

В частности, высокая степень и большая сложность результирующих булевых функций раундового преобразования таких шифров не позволяет успешно применять методы *интерполяционного криптоанализа* (interpolation attack) [73] и *криптоанализа на основе дифференциалов высокого порядка* (high order differentials) [78].

Зависимость управляющего вектора управляемой операции от всех битов блока данных и отсутствие разбиения преобразуемого блока на части делает невозможным эффективное применение методов *усеченного дифференциального криптоанализа* (truncated differentials) [77, 78] в отличие от других шифров, использующих управляемые циклические сдвиги (RC5, RC6) и **S**-блоки.

Отсутствие редких дифференциалов (разностей) преобразуемых данных делает неэффективными методы *дифференциального криптоанализа на основе невозможных или маловероятных разностей* (impossible differentials) [42].

Однородность преобразования в рамках всего шифра не дает необходимого эффекта от применения такой разновидности дифференциального криптоанализа, как *"бумеранговая атака"* (boomerang attack) [132].

Несмотря на отсутствие предварительных процедур вычисления рабочих раундовых ключей и на применение в шифре непосредственно секретных ключей, наличие в алгоритме специального расписания их использования и специальные процедуры делают шифр стойким к различным видам так называемой *"скользящей атаки"* (slide attack) [47, 48] и атаки на основе *относительных ключей* (related key attack) [76], дающих определенные результаты, например, при анализе блочного шифра ГОСТ.

Применение в каждом раунде большого количества ключей делает малоэффективным и такой мощный метод криптоанализа, как атака на основе *генерации аппаратных ошибок*, которая в таких же условиях может дать высокий эффект при анализе других шифров (DES, ГОСТ и др.).

Высокая статистическая неопределенность *(робастность)* управляемых операций в рамках концепции управляемых преобразований, потенциально увеличивает стойкость блочных шифров не только к существующим, но и возможным будущим криптографическим атакам.

В условиях высоких темпов развития микроэлектронных технологий внедрение блочных шифров на базе управляемых операций, предназначенных для массового применения в высокоскоростных системах защиты информации, делает их более перспективными по сравнению с другими блочными шифрами, не учитывающими современные тенденции в криптоанализе и в развитии компьютерных технологий.

Подробное описание всех перечисленных выше способов криптоанализа блочных шифров на базе управляемых операций выходит за рамки данной книги. В последующих разделах в качестве иллюстрации аппарата исследований в этой достаточно многообразной области математически рассматриваются основные методы криптоанализа — базовый дифференциальный и линейный криптоанализ, а также затрагиваются вопросы стойкости этих шифров к техническим атакам.

Методы ДКА и ЛКА считаются самостоятельными направлениями криптографического анализа алгоритмов, однако между ними имеется глубокая взаимосвязь, о чем свидетельствуют приводимые для них далее характеристики. Вместе с тем, существуют примеры, когда блочные алгоритмы стойки к одному из этих методов, но нестойки к другому.

Главным содержанием ДКА является анализ степени влияния распространения изменений в открытом тексте на изменения в шифртексте *(propagation properties)*.

Характеристики такого рода тесно связаны с автокорреляционными свойствами булевых функций, формирующих раундовые преобразования блочных шифров. В соответствии с результатами, представленными в *разд. 4.2.5* и *4.2.6*, автокорреляционные свойства этих преобразований определяются в основном:

- критерием распространения изменений $\mathbf{PC}(n, l)$ для каждой из образующих криптографический примитив булевой функции;
- совместным распределением лавинных векторов всего преобразования в раунде шифрования.

В основе ЛКА лежит линейная аппроксимация нелинейных раундовых преобразований. Линеаризация такого типа *(linear properties)* тесно связана с корреляцией образующих эти преобразования булевых функций с множеством аффинных функций. Качество линеаризации определяется представленными в *разд. 4.2.4, 4.2.8—4.2.10* свойствами линейности и нелинейности образующих криптографический примитив булевых функций ($\mathbf{L}(f)$, $\mathbf{N}(f)$) и всего раундового преобразования в целом ($\mathbf{L}(F)$, $\mathbf{N}(F)$).

Таким образом, с математической точки зрения ДКА следует считать автокорреляционным, а ЛКА — корреляционным анализом алгоритмов. В том и другом случае основным аппаратом исследований свойств криптографических примитивов может служить преобразование Уолша-Адамара, являющееся частным видом дискретного преобразования Фурье *(см. разд. 4.2.3)*. Об общности природы ДКА и ЛКА [52, 59] свидетельствует тот факт, что идеальных показателей стойкости к этим видам криптоанализа преобразования достигают лишь в том случае, когда они являются векторными бент-функциями *(см. разд. 4.2.8)*, т. е. единственными функциями, обладающими максимальной нелинейностью $\mathbf{N}(F)$ и самой высокой степенью распространения изменений $\mathbf{PC}(n)$. К сожалению, использование в блочных шифрах преобразований на основе только бент-функций в чистом виде нецелесообразно, а иногда и невозможно, поскольку такие преобразования несбалансированы и небиективны, что является безусловной слабостью. В связи с этим при практическом проектировании криптографических примитивов и, в частности УПО, приходится использовать, прежде всего, биективные отображения на основе композиции булевых функций, сочетающих различные криптографические свойства *(см. разд. 4.2)*, в том числе используются и бент-функции. Для таких отображений задача проведения ДКА и ЛКА всегда актуальна.

Проверка блочных шифров на стойкость к ДКА и ЛКА составляет целый комплекс сложных математических методов и требует определенного искусства в решении этой проблемы. Дело в том, что при проведении такого криптоанализа необходимо рассматривать разностные и линейные свойства не только отдельных криптографических примитивов, составляющих раундовое преобразование блочных шифров, но и учитывать их взаимное расположение и, следовательно, влияние друг на друга. В случае высокой сложности совокупной системы криптографических примитивов решение проблемы криптоанализа в общем виде мо-

жет оказаться весьма трудоемкой математической задачей. В итоге не исключена ситуация, при которой совокупность сложных преобразований приведет в целом к криптографически слабому шифру. Возможны и случаи, когда добавление к стойкому раундовому преобразованию дополнительных внешне усложняющих элементов может привести к ослаблению алгоритма. В связи с этим при проектировании блочных шифров стараются использовать теоретически просчитываемые конструкции, обеспечивающие определенность в оценке стойкости блочных алгоритмов к современным методам криптоанализа.

Ввиду большой сложности проблематики криптоанализа в задачи книги не входит достаточно полное освещение этого вопроса. В последующих разделах в той или иной степени будут затрагиваться лишь базовые принципы ДКА и ЛКА, а также общие механизмы влияния управляемых преобразований на стойкость блочных шифров к этим криптоатакам. В *гл. 6* в качестве иллюстрации приводятся примеры практического дифференциального криптоанализа некоторых блочных алгоритмов на базе управляемых операций.

4.4.2. Теоретические основы дифференциального криптоанализа

Для определения механизмов влияния криптографических примитивов и, в частности УПО, на стойкость блочных шифров к ДКА введем некоторые понятия, а также рассмотрим основные принципы проведения ДКА для DES-подобных блочных алгоритмов, имеющих четный размер блока преобразуемых данных n и осуществляющих r-раундовое преобразование. Напомним, что *алгоритм схемы Фейстеля* имеет следующий вид:

1. **For** $I = 1$ **to** $I = r–1$ **do**

   $[X_R(i) = X_L(i–1), X_L(i) = F(X_L(i–1), K_i) \oplus X_R(i–1), X(i) = X_L(i) \| X_R(i),$

   где $\forall i \in \{1, \ldots, r\}$ $X(i) \in \mathrm{GF}(2)^n$, $X_L(i)$ и $X_R(i) \in \mathrm{GF}(2)^{n/2}$,

   $X = X_L \| X_R = X_L(0) \| X_R(0) = X(0)$ — *исходный блок* данных *(открытый текст)*,

   $F(X_L(i–1), K_i)$ — *раундовая функция*,

   $K_i \in \mathrm{GF}(2)^{n/2}$ — *текущий раундовый ключ* из множества
   $\boldsymbol{K} = \{K_1, \ldots, K_r\} \in \mathrm{GF}(2)^{rn/2}$].

2. **For** $i = r$ **do**

   $[Y_R = F(X_L(r), K_r) \oplus X_R(r), Y_L = X_L(r), Y = Y_L \| Y_R,$

   где $Y \in \mathrm{GF}(2)^n$ — выходной блок (закрытый текст)].

   В соответствии с данным алгоритмом i-е раундовое преобразование по схеме Фейстеля определяется следующей функцией:

   $X(i) = \boldsymbol{E}(X(i–1), K_i) = [F(X_L(i–1), K_i) \oplus X_R(i–1)] \| X_L(i–1).$

Определение 4.45

Разностями между двумя блоками X и X^* или Y и Y^* называются выражения вида [84]: $\Delta X = X \circ (X^*)^{-1}$, $\Delta Y = Y \circ (Y^*)^{-1}$, где "$\circ$" — некоторая групповая операция (совокупность операций) над векторами из $\mathrm{GF}(2)^n$, а $(X^*)^{-1}$ и $(Y^*)^{-1}$ являются обратными элементами для X^* и Y^* в этой группе.

Как правило, указанная выше операция соответствует способу воздействия раундового ключа K_i на исходный блок X_L в раундовой функции $F(X_L(i–1), K_i)$. Как следует из приведенного выше примера, для DES и других блочных шифров такой операцией чаще всего является операция "$\oplus$", для которой разности приобретают следующий вид: $\Delta X = X \oplus X^* = \Delta X_L \parallel \Delta X_R$, $\Delta Y = Y \oplus Y^* = \Delta Y_L \parallel \Delta Y_R$. Далее для простоты изложения в качестве групповой операции используется операция "$\oplus$". Для указания принадлежности разностей к конкретным раундам преобразования будут применяться обозначения $\Delta Y(i) = Y(i) \oplus Y^*(i) = \Delta Y_L(i) \parallel \Delta Y_R(i)$. В случае только одного раунда преобразований для раундовой функции используется обозначение $F(X_L, K)$, а для самого однораундового преобразования — обозначение $\boldsymbol{E}(X, K)$.

Определение 4.46

S-раундовой характеристикой блочного шифра называется последовательность вида: $\{\Delta X, \Delta Y(1), \Delta Y(2), \ldots, \Delta Y(s)\}$, где $\forall i \in \{1, \ldots, s\}$ $\Delta X, \Delta Y(i) \in \mathrm{GF}(2)^n$, $s \le r$.

Определение 4.47

S-раундовой разностью (дифференциалом) называется пара вида $\{\Delta X, \Delta Y(s)\}$, где $\Delta X, \Delta Y(s) \in \mathrm{GF}(2)^n$, $s \le r$.

Определение 4.48

S-раундовая разность $\{\Delta X, \Delta Y(s)\}$ (s-раундовая характеристика $\{\Delta X, \Delta Y(1), \Delta Y(2), \ldots, \Delta Y(s)\}$) называется *выполненной (правильной)* для блоков X, X^* и раундовых ключей $\boldsymbol{K_S} = \{K_1, \ldots, K_s\}$, если в ходе промежуточных раундовых преобразований ΔX отображается в $\Delta Y(s)$, последовательно переходя значения $\Delta Y(1), Y(2), \ldots, \Delta Y(s)$.

Из определения разности следует, что при $\Delta X = 0$ $\forall s \in \{1, \ldots, r\}$ $\Delta Y(s) = 0$, и указанная нулевая разность не несет никакой информации о раундовых ключах. Далее будем считать, что исходные нулевые разности исключаются из рассмотрения. Вероятность выполнения s-раундовой характеристики $\{\Delta X, \Delta Y(1), \Delta Y(2), \ldots, \Delta Y(s)\}$ и s-раундовой разности $\{\Delta X, \Delta Y(s)\}$ при использовании раундовых ключей $\boldsymbol{K}$ можно рассматривать, соответственно, как условные вероятности по X вида:

$$\mathbf{Pr}_X(\Delta Y(s), \ldots, \Delta Y(1) / \Delta X, \boldsymbol{K_S}) \text{ и } \mathbf{Pr}_X(\Delta Y(\mathrm{s}) / \Delta X, \boldsymbol{K_S}),$$

которые по формуле полной вероятности связаны следующим соотношением:

$$\mathbf{Pr}_X(\Delta Y(s)/\Delta X, \boldsymbol{K}_S) = \sum_{\forall \Delta Y(1),\ldots,\Delta Y(s-1)\in GF(2)^n} \mathbf{Pr}_X(\Delta Y(s),\ldots,\Delta Y(1)/\Delta X, \boldsymbol{K}_S). \quad (4.17)$$

В соответствии с [84] большое значение для проведения ДКА имеет *гипотеза о стохастической эквивалентности*, которая считается выполненной, если $\forall s\in\{1, \ldots, r\}$ $\mathbf{Pr}_X(\Delta Y(s) / \Delta X, \boldsymbol{K_S}) \approx \mathbf{Pr}_{X,K}(\Delta Y(s) / \Delta X)$ *для большинства раундовых ключей* $\boldsymbol{K_S}$ (рассматривается совместное распределение вероятностей по X и по множеству ключей $\boldsymbol{K_S}$ при условии ΔX). Таким образом, для большинства ключей указанная вероятность не зависит от значения раундовых ключей, что можно записать в виде приближенного равенства $\mathbf{Pr}_X(\Delta Y(s) / \Delta X) \approx \mathbf{Pr}_{X,K}(\Delta Y(s) / \Delta X)$. Поскольку для большинства существующих блочных алгоритмов указанная гипотеза выполняется, то в дальнейшем будем рассматривать только такие шифры.

Определение 4.49

Раундовое преобразование $\boldsymbol{E}(X(i-1), K_i)$ называется слабым, если по известным значениям $Y(i)$, $Y^*(i)$ и $\Delta X(i-1)$ можно определить раундовый ключ K_i.

Возможность проведения ДКА связана с наличием $(r-1)$-раундовой разности $\{\Delta X, \Delta Y(r-1)\}$ (или характеристики $\{\Delta X, \Delta Y(1), \Delta Y(2), \ldots, \Delta Y(r-1)\}$), для которой выполняется условие $\mathbf{Pr}_X(\Delta Y(r-1)/\Delta X) > 1/2^n$.

При выполнении гипотезы о стохастической эквивалентности и наличии слабого раундового преобразования используется следующий базовый алгоритм ДКА, являющийся разновидностью криптографической атаки на основе подобранных открытых текстов:

1. Выбирается $(r-1)$-раундовая разность $\{\Delta X, \Delta Y(r-1)\}$ или соответствующая характеристика, имеющая максимальную или близкую к максимальной условную вероятность $p_{r-1} = \mathbf{Pr}_X(\Delta Y(r-1)/\Delta X) > 1/2^n$. При этом для простоты вычислений, как правило, стараются использовать векторы ΔX с небольшим весом Хэмминга.
2. Случайно и равновероятно (или специальным образом) выбирается исходный текст $X\in \mathrm{GF}(2)^n$ и формируется другой текст $X^* = X \oplus \Delta X$.
3. Тексты X и X^* преобразуются блочным шифром в закрытые тексты $Y(r)$, $Y^*(r)$ по неизвестным раундовым ключам $\boldsymbol{K} = \{K_1, \ldots, K_r\}$.
4. Используя слабость раундового преобразования $\boldsymbol{E}(X(r-1), K_r)$ и предполагая, что $\Delta Y(r-1)$ известно с вероятностью p_{r-1}, по имеющимся текстам $Y(r)$, $Y^*(r)$ аналитически определяется множество $\mathbf{K}(X, \Delta X, \Delta Y(r-1))$ всех возможных значений раундового ключа K_r. Для каждого выбранного X проводится накопление этих значений в множестве $\mathbf{K}^*(\Delta X, \Delta Y(r-1)) = \bigcup_{X\in GF(2)^n} \mathbf{K}(X, \Delta X, \Delta Y(r-1))$.

5. Шаги 1—4 повторяются до тех пор, пока в множестве $\mathbf{K}^*(\Delta X, \Delta Y(r-1))$ не появится локальный максимум для одного или для ограниченного множества значений раундового ключа K_r, среди которых и находится его истинное значение.
6. Далее, после снятия влияния последнего r-го раунда, шаги 1—5 повторяются для ослабленного (r–1)-раундового преобразования до получения ключа K_{r-1} и так вплоть до K_1. В некоторых случаях, например в DES, остальные раундовые ключи могут определяться тотальным перебором, исходя из способа их формирования.

Если для проведения ДКА используют характеристики разностей, то они, как правило, формируются в виде периодически повторяющихся цепочек из небольших последовательностей однораундовых характеристик. Такие цепочки получили название итеративных характеристик. Вероятность этих характеристик p_{r-1} с определенной степенью приближения является произведением вероятностей составляющих их однораундовых характеристик в предположении их независимости. Если в качестве меры сложности $\mathbf{M_D}(r)$ ДКА блочного шифра, состоящего из r раундов, рассматривать необходимое количество пар входных и выходных блоков данных для определения раундового ключа K_r, то справедлива следующая оценка [59]:

$$\mathbf{M_D}(r) \geq 1/p_{r-1}^{(\max)}, \text{ где } p_{r-1}^{(\max)} = \max_{\forall \Delta X, \forall \Delta Y(r-1)} p_{r-1}. \tag{4.18}$$

Отсюда следует, что при $p_{r-1}^{(\max)}$ близких к $1/2^n$ значения сложности $\mathbf{M_D}(r)$ будет соизмеримо или превышать общее количество 2^n двоичных текстов длины n битов и, следовательно, разностный криптоанализ станет неэффективным.

Более точно сложность проведения ДКА следует оценивать с учетом так называемого *отношения "сигнал-шум"* (S/N) [44], где под сигналом понимается среднее число повторений правильного ключа, а под шумом — среднее число случайных ключей в множестве $\mathbf{K}^*(\Delta X, \Delta Y(r-1))$. Количественно это отношение определяется следующим образом: $S/N = (mp)/(m\alpha\beta / 2^k) = (2^k p)/(\alpha\beta)$, где k — число вычисляемых битов ключа, m — общее количество исследуемых пар открытых текстов $(X, X \oplus \Delta X)$, p — вероятность применяемой характеристики $(\Delta X, \ldots, \Delta Y)$, α — среднее число получаемых ключей в множестве $\mathbf{K}(X, \Delta X, \Delta Y(r-1))$ для каждой пары $(X, X \oplus \Delta X)$ и β — отношение количества пар $(X, X \oplus \Delta X)$, которые на выходе приводят к непротиворечивой разности ΔY (то есть не отбраковываются), к общему числу m исследуемых пар открытых текстов. При этом в соответствии с определением 4.48 пары, полностью удовлетворяющие характеристике на (r–1)-м раунде, называются *правильными парами*. Остальные пары, которые не отбраковываются после r раундов, но лишь частично соответствуют (r–1)-раундовой характеристике, называются *неправильными парами*. Важным результатом указанных рассуждений является тот факт, что общее число правильных пар, необходимых для успешного проведения ДКА, непосредственно зависит от

отношения S/N. Для сокращенного до 15 раундов блочного шифра DES экспериментально определено [44], что при $S/N \approx 1$ требуются всего 3—4 правильные пары для эффективного проведения ДКА.

Практически значимые оценки сложности ДКА получены для большого класса блочных алгоритмов [102], которые назвали *марковскими шифрами*.

Определение 4.50

Блочный алгоритм называется *марковским* [84], если $\forall X \in \mathrm{GF}(2)^n$ при случайном и равновероятном распределении раундовых ключей $\boldsymbol{K} = \{K_1, \ldots, K_r\}$ имеет место соотношение: $\mathbf{Pr}_K(\Delta Y(1)/\Delta X, X) = \mathbf{Pr}_{X,K}(\Delta Y(1)/\Delta X)$.

В случае выполнения гипотезы о стохастической эквивалентности, указанное в определении равенство приобретает вид: $\mathbf{Pr}(\Delta Y(1)/\Delta X, X) = \mathbf{Pr}_X(\Delta Y(1)/\Delta X)$.

Таким образом, отличительной чертой марковских алгоритмов является независимость распределения вероятностей однораундовых разностей от выбора исходного текста. Например, DES, ГОСТ 28147–89, IDEA и др. являются марковскими шифрами.

Очевидно, что свойство марковости и гипотеза о стохастической эквивалентности тесно связаны, поскольку отражают близкие по значению характеристики блочных шифров относительно исходных текстов в первом случае и ключей во втором случае.

Определение 4.51

Последовательность разностей $\Delta X, \Delta Y(1), \ldots, \Delta Y(r)$ образует цепь Маркова [6, 23], если $\forall s \in \{1, \ldots, r\}$ имеет место соотношение:

$$\mathbf{Pr}_X(\Delta Y(s)/\Delta Y(s-1), \ldots, \Delta Y(1)) = \mathbf{Pr}_X(\Delta Y(s)/\Delta Y(s-1)).$$

Цепь Маркова для блочного шифра определяется следующими параметрами:

- начальным распределением вероятностей исходных состояний $p_i^{(0)} = \mathbf{Pr}_X(\Delta X = \Delta Y(0) = \alpha_j)$, где $\alpha_1, \ldots, \alpha_{2^n}$ — упорядоченная в лексикографическом порядке последовательность всех возможных значений двоичных векторов длины n (состояний цепи);
- однораундовой *матрицей разностей (переходных вероятностей)* $\mathbf{P} = \| p_{i,j} \|_{(2^n \times 2^n)}$, где $p_{i,j} = \mathbf{Pr}_X(\Delta Y(1) = \alpha_j / \Delta Y(0) = \alpha_i) = \mathbf{Pr}_X(F(X, K) \oplus F(X \oplus \alpha_i, K) = \alpha_j)$.

В соответствии с результатами, приведенными в *разд. 4.2.5*, переходные вероятности связаны с автокорреляционными характеристиками однораундового преобразования следующим образом [59]:

$$p_{i,j} = \frac{1}{2^n} \sum_{\omega \in GF(2)^n} (-1)^{\omega \bullet \alpha_j} \mathbf{R}_{\alpha_i, \omega}(F(X, K)).$$

На языке матрицы **P** стойкость шифра к ДКА зависит от значения $p^{(\max)} = \max\limits_{\forall i,j} p_{i,j}$. Для биективных отображений всегда справедливо неравенство $p^{(\max)} \geq 1/2^{n-1}$ [59].

В случае марковских алгоритмов, образующих цепь Маркова из последовательности s-раундовых характеристик, справедливо следующее соотношение [84]:

$$\mathbf{Pr}_X(\Delta Y(s), ..., \Delta Y(1)/\Delta X) = \prod_{i=1}^{s} \mathbf{Pr}_X(\Delta Y(i)/\Delta Y(i-1)) =$$

$$= \prod_{i=1}^{s} \mathbf{Pr}_X(\Delta Y(1) = \Delta Y(i)/\Delta X = \Delta Y(i-1)).$$

С учетом соотношения (4.17) для s-раундовых разностей имеем:

$$\mathbf{Pr}_X(\Delta Y(s)/\Delta X) = \sum_{\forall \Delta Y(1),...\Delta Y(s-1) \in GF(2)^n} \prod_{i=1}^{s} \mathbf{Pr}_X(\Delta Y(1) = \Delta Y(i)/\Delta X = \Delta Y(i-1)).$$

Данные формулы теоретически позволяют получать распределения вероятностей s-раундовых характеристик и s-раундовых разностей всего марковского шифра на основе однораундовых распределений. Однако на практике такого рода формулы могут оказаться достаточно сложными для реальных вычислений.

В связи с этим для марковских шифров большое практическое значение имеет неравенство $p_{r-1}^{(\max)} < 2(p^{(\max)})^2$ при $r \geq 4$ [102], из которого следует, что для обеспечения стойкости к ДКА достаточно построить раундовую функцию таким образом, чтобы для однораундовых разностей обеспечивалась по возможности низкая вероятность $p^{(\max)}$. В этом случае в соответствии с условием (4.18) гарантирована высокая трудоемкость $\mathbf{M_D}(r)$ проведения ДКА для всего блочного шифра.

Как правило, достаточным условием марковости блочных алгоритмов является использование операции "$\oplus$" при "наложении" ключа на преобразуемые данные ($X \oplus K$) [84]. Обобщая эти результаты, нетрудно показать, что таким же достаточным условием марковости блочного шифра, как правило, является биективность отдельно по K и по X *(см. разд. 4.3.1)* функции "наложения" ($Y = f(X, K)$) раундового ключа K на преобразуемые данные X (доказательство этих результатов выходит за рамки данной книги). В соответствии с выводами из *разд. 4.3.3* такую функцию в полной мере могут реализовать УПО, что является еще одним положительным качеством этого криптографического примитива.

В завершение раздела следует напомнить, что рассуждения относительно стойкости блочных шифров к ДКА проводились в условиях выполнения гипотезы о стохастической эквивалентности. В случае, когда эта гипотеза не выполняется,

оценки вероятностей разностей или характеристик должны проводиться не только по значениям исходных текстов $X \in \mathrm{GF}(2)^n$, но и по всему множеству ключей $\boldsymbol{K} \in \mathrm{GF}(2)^{m/2}$, что значительно затрудняет анализ стойкости проектируемых шифров к ДКА.

Необходимо также обратить внимание на то обстоятельство, что приведенные в данном разделе оценки обеспечивают лишь *достаточные условия* стойкости марковских блочных шифров к ДКА. В действительности может существовать масса блочных алгоритмов, в том числе и немарковских, для которых указанные условия не выполняются, но шифры все же оказываются стойкими к этому виду криптоанализа. Зачастую такие выводы достаточно сложно получить в общем виде даже для одного раунда шифрования. В таком же положении находится и криптоаналитик, пытающийся определить слабости криптоалгоритма относительно ДКА.

В этой ситуации задача решается в частном виде, т. е. рассматриваются лишь те характеристики или разности, вероятности которых можно реально оценить. Как упоминалось ранее, обычно практически применимые характеристики для всего блочного шифра являются цепочками итеративных характеристик, т. е. небольшими периодически повторяющимися последовательностями однораундовых характеристик. Как правило, однораундовые характеристики имеют небольшой вес Хэмминга или какой-то специальный вид. Задача криптоаналитика состоит в поиске итеративных характеристик или разностей, имеющих наибольшую вероятность, достаточную для проведения криптоатаки на шифр. С другой стороны, целью создателей блочных шифров является доказательство стойкости шифра к ДКА в общем виде или исследование вероятностей всего многообразия реально просчитываемых характеристик с целью показать невозможность отличия блочного шифра от случайной подстановки. В *гл. 6* приводятся конкретные примеры оценки стойкости к ДКА ряда новых блочных шифров на базе управляемых операций с использованием двух и трехраундовых итеративных характеристик.

4.4.3. Влияние управляемых подстановочных операций на стойкость блочных шифров к дифференциальному криптоанализу

Задача оценки стойкости блочных шифров к ДКА в общем виде является, как отмечалось ранее, достаточно сложной математической проблемой. Поэтому в настоящем разделе приводится лишь иллюстрация механизмов влияния новых криптографических примитивов — управляемых подстановочных операций (УПО) на стойкость блочных шифров к ДКА. Применяемый для этих целей математический аппарат показывает определенную гибкость УПО, позволяющую варьировать их структуру и проводить настройку криптографических свойств этих операций в соответствии с заданными требованиями. В качестве примера оценки

стойкости к ДКА блочных шифров на базе УПО рассмотрим упомянутый в *разд. 4.3.2* случай использования такой операции в схеме Фейстеля вместо операции "⊕" (см. рис. 4.2 и 4.3). Предположим, что рассматриваемый блочный шифр является марковским, и для него выполняется гипотеза о стохастической эквивалентности *(см. разд. 4.4.2)*. Для упрощения дальнейших рассуждений будем использовать только математическую запись процесса зашифрования ($e = 0$), опуская параметр e при написании формул.

В качестве представления УПО обратимся к варианту подстановки $\sigma_F^{(V)}(X_R)$ последовательной параметрически обратимой модели УПО $\mathbf{S}'_{\sigma,e_S}$ *(см. разд. 4.3.3)*. Поскольку в рассматриваемой схеме Фейстеля УПО преобразует лишь половину блока данных, то подстановка имеет следующий вид:

$$\sigma_F^{(V)}(X_R) = \{f_1^{(V)}, f_2^{(V)}, ..., f_{n/2}^{(V)}\},$$

где $\forall i \in \{1, \ldots, n/2\}$ $f_i^{(V)} = x_i^{(R)} \oplus \varphi^{(V)}(X_R^{(i-1,i-k)}, F)$ и $F = F(X_L, K)$, $X_R^{(i-1,i-k)} = (x_{i-k}, \ldots, x_{i-1})$, $X^{(0,\,1-k)} = (x_0, x_{-1}, \ldots, x_{1-k})$ — начальные условия или k значений, полученных в результате расширяющего преобразования $\mathrm{GF}(2)^{n/2} \to \mathrm{GF}(2)^{n/2+k}$ вектора X_R.

Исходя из приведенных в *разд. 4.4.2* выводов о возможности оценки стойкости всего блочного шифра по одораундовым показателям стойкости, сравним вероятности разностей традиционной и модифицированной схемы Фейстеля для одного раунда преобразований, т. е. для $r = 1$.

Поскольку в традиционной схеме Фейстеля $Y_L = F \oplus X_R$, то для переходных вероятностей одораундовых разностей матрицы $\mathbf{P}^{(\oplus)} = \| p_{i,j}^{(\oplus)} \|$ имеем:

$$p_{\Delta X,\Delta Y_L}^{(\oplus)} = \mathbf{Pr}_X^{(\oplus)}(\Delta Y_L / \Delta X) = \mathbf{Pr}_X^{(\oplus)}(\Delta F \oplus \Delta X_R = \Delta Y_L / \Delta X),$$

где $\Delta F = F(X_L, K) \oplus F(X_L \oplus \Delta X_L, K)$.

Для модифицированной схемы Фейстеля, в которой подстановка $\sigma_F^{(V)}(X_R)$ используется вместо "⊕", раундовое преобразование в векторном представлении имеет следующий вид:

$$Y_L = \sigma_F^{(V)}(X_R) = F \oplus X_R \oplus \boldsymbol{\varphi}^{(V)},$$

где $V = V(X_L, K)$ и $\boldsymbol{\varphi}^{(V)} = \boldsymbol{\varphi}^{(V)}(X_R, F) = (\varphi_1^{(V)}(X_R^{(0,1-k)}, F), ..., \varphi_{n/2}^{(V)}(X_R^{(n/2-1,n/2-k)}, F))$.

Отсюда для модифицированной схемы условные вероятности одораундовых разностей матрицы $\mathbf{P}^{(\sigma)} = \| p_{i,j}^{(\sigma)} \|$ можно представить в следующем виде:

$$p_{\Delta X,\Delta Y_L}^{(\sigma)} = \mathbf{Pr}_X[\sigma_F^{(V\oplus\Delta V)}(X_R) \oplus \sigma_{F\oplus\Delta F}^{(V\oplus\Delta V)}(X_R \oplus \Delta X_R) = \Delta Y_L / \Delta X],$$

Далее имеем: $p^{(\sigma)}_{\Delta X,\Delta Y_L} = \mathbf{Pr}_X(\Delta F \oplus \Delta X_R \oplus \Delta\varphi^{(V)} = \Delta Y_L / \Delta X)$,

где $\Delta\boldsymbol{\varphi}^{(V)} = \boldsymbol{\varphi}^{(V)}(X_R, F) \oplus \boldsymbol{\varphi}^{(V\oplus\Delta V)}(X_R \oplus \Delta X_R, F \oplus \Delta F)$ и $\Delta V = V(X_L, K) \oplus V(X_L \oplus \Delta X_L, K)$.

Таким образом, получаем следующую аналитическую связь между вероятностями однораундовых разностей модифицированной и традиционной схемы Фейстеля:

$$p^{(\sigma)}_{\Delta X,\Delta Y_L} = \sum_{\forall \Delta Y'_L \in GF(2)^{n/2}} \mathbf{Pr}^{(\oplus)}_X(\Delta Y_L = \Delta Y'_L \oplus \Delta\varphi^{(V)} / \Delta F \oplus \Delta X_R = \Delta Y'_L) p^{(\oplus)}_{\Delta X,\Delta Y'_L}, \qquad (4.19)$$

т. е. имеется усреднение условных вероятностей разностей $p^{(\sigma)}_{\Delta X,\Delta Y_L}$ по распределению вероятностей $p^{(\oplus)}_{\Delta X,\Delta Y_L}$, что создает благоприятные условия для выполнения неравенства $p^{(\sigma)}_{\max} < p^{(\oplus)}_{\max}$ и, следовательно, может привести к повышению трудоемкости $\mathbf{M_D}(r)$ проведения ДКА *(см. разд. 4.4.2)*. Последнее означает, что модифицированная функция преобразования способна повысить стойкость алгоритма к ДКА, при этом параметры стойкости непосредственно зависят от выбора векторной булевой функции $\boldsymbol{\varphi}^{(V)}$. Важным фактором повышения стойкости к ДКА является также то обстоятельство, что при использовании УПО в отличие от **S**-блоков размерность пространства, на котором необходимо анализировать разности, увеличивается до размеров всего блока преобразуемых данных, что значительно усложняет криптоатаку.

Для получения более точных оценок рассмотрим, например, в качестве компонентов векторной булевой функции $\boldsymbol{\varphi}^{(V)}$ бент-функции *(см. разд. 4.2.8)*. В этом случае каждый компонент $\Delta\varphi_i^{(V)}$ векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$ будет сбалансированной булевой функцией при любых ненулевых значениях той части компонентов расширенного вектора $\Delta = \Delta X \| \Delta F \| \Delta V$, которая входит в $\Delta\varphi_i^{(V)}$. Отсюда следует, что распределение вероятностей каждого компонента $\Delta\varphi_i^{(V)}$ векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$ не зависит от значений $\Delta F \oplus \Delta X_R$ в выражении (4.19), которое приобретает следующий вид:

$$p^{(\sigma)}_{\Delta X,\Delta Y_L} = \sum_{\forall \Delta Y'_L \in GF(2)^{n/2}} \mathbf{Pr}^{(\oplus)}_X(\Delta Y_L = \Delta Y'_L \oplus \Delta\varphi^{(V)}) p^{(\oplus)}_{\Delta X,\Delta Y'_L}.$$

Таким образом, каждый компонент $\Delta y_i^{(L)}$ вектора ΔY_L будет сбалансированной булевой функцией относительно всех ненулевых значений аргументов функции $\Delta\varphi_i^{(V)}$. Однако компоненты векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$, вообще говоря, зависимы, следовательно, их совместное распределение по X может быть неравномерным, что должно учитываться при проведении криптоанализа сложных УПО. Для обеспечения независимости компонентов векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$ рассмотрим,

например, простейшие функции выхода для следующей реализации модели $\mathbf{S}'_{\sigma,e_S}$ *(см. разд. 4.3.3)*:

$$\forall i \in \{1, \ldots, n/2\} \;\; y_i^{(L)} = f_i^{(V)} = x_i^{(R)} \oplus a_i \oplus \varphi_i(x_{i-1}^{(R)}, a_{i-1}, \nu_{2i}, \nu_{2i-1}),$$

где $V = V(X_L, K)$ $(V \in \mathrm{GF}(2)^n)$ — некоторая функция расширения ключа и $F = (a_1, \ldots, a_{n/2}) \in \mathrm{GF}(2)^{n/2}$.

В соответствии с результатами, полученными в *разд. 4.3*, такая подстановка обеспечивает отображение $X_R \times F \to Y_L$, которое не имеет коллизий по F, биективно по X и имеет 2^n уникальных модификаций. Для подстановки $\sigma_F^{(V)}(X)$ i-й компонент векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$ приобретает вид:

$$\Delta\varphi_i = \varphi_i(x_{i-1}^{(R)}, a_{i-1}, \nu_{2i}, \nu_{2i-1}) \oplus \varphi_i(x_{i-1}^{(R)} \oplus \Delta x_{i-1}^{(R)}, a_{i-1} \oplus \Delta a_{i-1}, \nu_{2i} \oplus \Delta\nu_{2i}, \nu_{2i-1} \oplus \Delta\nu_{2i-1}),$$

где Δa_j и $\Delta x_j^{(R)}$ являются j-ми компонентами векторов ΔF и ΔX_R размерности $n/2$, а Δv_j является j-м компонентом вектора ΔV размерности n. В качестве простого примера бент-функций φ_i можно привести функции следующего вида: $\forall i \in \{1, \ldots, n/2\}$ $\varphi_i = x_{i-1}^{(R)}\nu_{2i-1} \oplus a_{i-1}\nu_{2i}$. В этом случае, если раундовая функция F обладает хорошими рассеивающими свойствами, то все компоненты векторной функции $\Delta\boldsymbol{\varphi}^{(V)}$ можно считать независимыми и сбалансированными булевыми функциями по четырем переменным. Каждый результирующий компонент $\Delta y_i^{(L)} = \Delta x_i^{(R)} \oplus \Delta a_i \oplus \Delta\varphi_i(x_{i-1}^{(R)}, a_{i-1}, \nu_{2i}, \nu_{2i-1})$ при этом будет несбалансированной булевой функцией лишь на трех из 63 возможных ненулевых векторах

$$(\Delta x_i^{(R)}, \Delta a_i, \Delta x_{i-1}^{(R)}, \Delta a_{i-1}, \Delta v_{2i-1}, \Delta v_{2i}) = (\gamma_1, \gamma_2, 0, 0, 0, 0), \; \gamma_j \in \{0, 1\}.$$

В этом случае $\forall i \in \{1, \ldots, n/2\}$ вероятность значений "1" на выходе каждой функции $\Delta y_i^{(L)}$ определяется следующим соотношением:

$$\Pr_X(\Delta y_i^{(L)} = 1) = (2/3)(3/63) + (1/2)(60/63) = 32/63 \approx 0{,}508.$$

Отсюда $\forall \Delta X \in \mathrm{GF}(2)^n$ $(\Delta X \neq 0)$ и $\forall \Delta Y_L \in \mathrm{GF}(2)^{n/2}$ имеем:

$$p_{\Delta X, \Delta Y_L}^{(\sigma)} = \sum_{\forall \Delta Y'_L \in GF(2)^{n/2}} 0{,}508^{\mathbf{WH}(\Delta Y_L \oplus \Delta Y'_L)}, 492^{(n/2 - \mathbf{WH}(\Delta Y_L \oplus \Delta Y'_L))} p_{\Delta X, \Delta Y'_L}^{(\oplus)},$$

где $\mathbf{WH}(\Delta Y_L \oplus \Delta Y'_L)$ — вес Хэмминга вектора $\Delta Y_L \oplus \Delta Y'_L$.

Это означает, что сконструированная указанным выше способом подстановка $\sigma_F^{(V)}(X_R)$ на базе бент-функций обеспечивает высокую степень усреднения исходных вероятностей разностей $p_{\Delta X, \Delta Y_L}^{(\oplus)}$ до значений $p_{\Delta X, \Delta Y_L}^{(\sigma)}$ достаточно близких к $1/(2^{n/2})$, т. к. $\lim\limits_{n \to \infty} (p_{\Delta X, \Delta Y_L}^{(\sigma)} - \dfrac{1}{2^{n/2}}) = 0$. Этот результат является иллюстрацией ме-

ханизма повышения стойкости блочных шифров к ДКА на базе специально построенных УПО. При этом одновременно преобразование обеспечивает биективность отображения $\Delta X_R \to \Delta Y_L$ и отсутствие коллизий по F в однораундовом преобразовании.

Подобные оценки можно получить и для более сложных векторных функций $\boldsymbol{\varphi}^{(V)}$ с зависимыми компонентами, если, например, в качестве образующих функций для $\boldsymbol{\varphi}^{(V)}$ рассматривать булевы функции следующего вида:

$$\varphi_i' = \varphi_i(x_{i-1}', a_{i-1}, \nu_{2i}, \nu_{2i-1}) \oplus \varphi_i''(X_R^{(i-2)}, A^{(i-2)}), \tag{4.20}$$

где φ_i — бент-функции, а φ_i'' — бент-функции или булевы функции с высокой степенью нелинейности и другими математическими свойствами, в частности свойствами **PC**(l, k) и **CI**(l) *(см. разд. 4.2.6 и 4.2.7)*.

В заключение необходимо все же отметить, что практически применимые результаты ДКА могут быть получены лишь на основе конкретного изучения взаимодействия всех составляющих элементов раундовых преобразований и, в частности, особенностей сочетания раундовой функции F с УПО.

4.4.4. Теоретические основы линейного криптоанализа

Использование в блочных шифрах управляемых подстановочных операций так же, как и в случае дифференциального криптоанализа, позволяет повышать стойкость этих алгоритмов и к линейному криптоанализу (ЛКА).

Вопросы ЛКА уже затрагивались в *гл. 3* при рассмотрении линейных свойств управляемых перестановочных операций *(см. разд. 3.4)*. В данном разделе обратимся к проблемам ЛКА под другим углом зрения и постараемся обобщить полученные ранее результаты. Не претендуя на исчерпывающее изложение этой математически сложной и содержащей элементы искусства тематики, попытаемся определить основы базовой методики проведения ЛКА. Из приводимых далее рассуждений и простейших примеров можно представить себе те сложности, которые могут возникать при проведении линейного криптоанализа в общем виде.

Определение 4.52

Линейной аппроксимацией раундовой функции $F(X_L, K)$ называется выражение вида: $\boldsymbol{b}'\bullet F(X_L, K) = \boldsymbol{a}'\bullet X_L$, где $\boldsymbol{a}', \boldsymbol{b}' \in \mathrm{GF}(2)^{n/2}$ именуются, соответственно, входными и выходными векторами-масками и "•" обозначает скалярное побитовое умножение двух векторов [86].

Определение 4.53

Линейной аппроксимацией блочного шифра после r раундов называется уравнение вида: $\boldsymbol{b}\bullet Y(r) \oplus \boldsymbol{a}\bullet X \oplus \boldsymbol{c}\bullet \boldsymbol{K} = 0$, где $\boldsymbol{a}, \boldsymbol{b} \in \mathrm{GF}(2)^n$ входные и выходные вектора-маски,

$\boldsymbol{K} = (K_1, K_2, \ldots, K_r)$ и $\boldsymbol{c} = (c_1, c_2, \ldots, c_l) \in \mathrm{GF}(2)^l$ $(l = rn/2)$ — вектор-маска ключа. Линейная аппроксимация соответствует *линейной характеристике* шифра $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, p)$, где p — значение *смещения* от 1/2 вероятности выполнения этой аппроксимации *(см. разд. 3.4.1)* [86, 101].

Линейностью раундовой функции $F(X_L, K)$ в соответствии с общим определением из *разд. 4.2.4* является значение следующего выражения:

$$\mathbf{L}(F) = \underset{\forall a',\, b' \in GF(2)^{n/2},\, b \neq 0}{2\max} \left| \mathbf{Pr}_{X_{L,K}}(b' \bullet F(X_L, K) = a' \bullet X_L) - \frac{1}{2} \right|.$$

При этом линейность и нелинейность функции $F(X_L, K)$ в соответствии с результатами из *разд. 4.2.4* связаны следующим соотношением:

$$\mathbf{L}(F) = 1 - 1/2^{(n/2)-1}\mathbf{N}(F).$$

Метод ЛКА является достаточно сложным вероятностно-алгебраическим методом, применение которого зависит от целого ряда особенностей того или иного блочного шифра. Схематично один из распространенных вариантов ЛКА DES-подобных блочных алгоритмов заключается в следующей последовательности действий:

1. Выбираются линейные аппроксимации раундовой функции (маски $\boldsymbol{b}'$ и $\boldsymbol{a}'$), вероятность выполнения которых $\mathbf{Pr}_{X_{L,K}}(b' \bullet F(X_L, K) = a' \bullet X_L)$ максимально отличается от 1/2.
2. $r - 2$ линейных раундовых аппроксимаций последовательно увязываются *(согласовываются)* в цепочку для получения линейной аппроксимации блочного шифра после $r - 2$ раундов, начиная со 2-го и до предпоследнего раунда. При этом маски входа каждого последующего раунда цепочки должны быть равны маскам выхода предыдущих раундов. Для получившейся цепочки рассматривается результирующая вероятность $\mathbf{Pr}_X(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{K}_{r-2})$ следующего события:

$$\boldsymbol{a} \bullet Y(1) \oplus \boldsymbol{b} \bullet Y(r-1) \oplus \boldsymbol{c} \bullet (0, K_2, \ldots, K_{r-1}, 0) = 0, \tag{4.21}$$

   при этом вероятность данного события формируется в соответствии с правилами согласования раундовых линейных характеристик *(см. разд. 3.4.1)* и должна существенно отличаться от 1/2, т. е. результирующая линейная характеристика должна иметь большое смещение.
3. Из условия слабости раундового преобразования значения $Y(1)$ и $Y(r - 1)$ аналитически выражаются соответственно через X, K_1 и $Y(r)$, K_r и подставляются в уравнение (4.21).

Предположим, что, используя особенности того или иного блочного алгоритма, удается в уравнение (4.21) "вовлечь" лишь часть битов из K_1 и K_r (для DES это 12 битов). Тогда в случае правильного предположения относительно битов из K_1 и K_r перебор различных X обеспечивает выполнение уравнения (4.21) с вероят-

ностью $\mathbf{Pr}_X(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{K}_{r-2})$, в противном случае уравнение (4.21) выполняется случайно с вероятностью близкой к 1/2. Указанное обстоятельство позволяет на основе перебора различных "вовлеченных" в уравнение битов из K_1 и K_r статистическим методом определить их истинное значение.

Простейшим или, иначе, базовым, методом ЛКА является метод, при котором рассматриваются линейные аппроксимации вида $\boldsymbol{a}\bullet X \oplus \boldsymbol{b}\bullet Y(r) \oplus \boldsymbol{c}\bullet(K_1, K_2, \ldots, K_r) = 0$ и их вероятности $g_r = \mathbf{Pr}_X(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{K})$. Если линейная характеристика $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, q_r = g_r - 1/2)$ этой аппроксимации имеет существенное r-раундовое смещение q_r, то данный метод позволяет определить четность битов ключа, участвующих в аппроксимации.

Если в качестве меры сложности $\mathbf{M_L}(r)$ базового метода ЛКА рассматривать количество пар исходных и шифрованных текстов X и $Y(r)$, необходимых для статистического определения четности битов ключей, то справедлива следующая оценка [101]: $\mathbf{M_L}(r) \geq 1/(q_r^{(\max)})^2$, где $q_r^{(\max)} = \max\limits_{\forall a, \forall b, \forall c} |q_r|$. Отсюда следует, что при $q_r^{(\max)}$ близких к $1/2^{n/2}$ значения сложности $\mathbf{M_L}(r)$ для проведения ЛКА будет соизмеримо или превышать общее количество (2^n) двоичных текстов длины n битов и, следовательно, линейный криптоанализ станет неэффективным.

Отметим, что из определения линейности преобразования имеет место следующее соотношение: $q_r^{(\max)} = \frac{1}{2}\mathbf{L}(Y(r))$ и, следовательно, $\mathbf{M_L}(r) \geq 4(\mathbf{L}(Y(r))^{-2}$.

Так же, как и в случае ДКА, при $r \geq 4$ для марковских блочных шифров справедливо неравенство [87]: $q_r^{(\max)} < 2(q^{(\max)})^2$, где $q_r^{(\max)} = q_1^{(\max)}$ Отсюда и из соотношения $q_1^{(\max)} = \frac{1}{2}\mathbf{L}(\boldsymbol{F})$ вытекает еще одна оценка для сложности ЛКА:

$$\mathbf{M_L}(r) > 4\mathbf{L}(\boldsymbol{F})^{-4} \tag{4.22}$$

Из указанных оценок следует, что для стойкости к ЛКА достаточно построить раундовое преобразование шифра таким образом, чтобы обеспечивалось низкое однораундовое смещение $q^{(\max)}$, что гарантирует высокую трудоемкость $\mathbf{M_L}(r)$ проведения ЛКА. Данное условие равносильно построению раундового преобразования с низкими показателями линейности $\mathbf{L}(\boldsymbol{F})$ или, соответственно, высокими показателями нелинейности $\mathbf{N}(\boldsymbol{F})$.

Из приведенных выше результатов следует, что линейные свойства итеративных шифров (подобно дифференциальным свойствам в ДКА с матрицей переходных вероятностей $\mathbf{P}$) можно характеризовать с помощью *корреляционной матрицы* $\mathbf{C} = || c_{i,j} ||$, определяемой следующим образом *(см. разд. 4.2.4)*:

$$c_{i,j} = \mathbf{Corr}(\alpha_j\bullet Y(1), \alpha_i\bullet X) = 2[(\mathbf{Pr}_{X,K}(\alpha_j\bullet Y(1) = \alpha_i\bullet X) - 1/2)],$$

где $\alpha_1, \ldots, \alpha_{2^n}$ — упорядоченная в лексикографическом порядке последовательность из всех 2^n значений масок ($\alpha_i, \alpha_j \in \mathrm{GF}(2)^n$). При этом

$$c^{(\max)} = \max_{\forall i, \forall j} \left| c_{i,j} \right| = \mathbf{L}(\boldsymbol{F}) = 1 - \frac{1}{2^{(n/2)-1}} \mathbf{N}(\boldsymbol{F}) = 2q^{(\max)}.$$

Аналогично ограничениям на значение максимальных элементов матрицы **P** для ДКА имеется следующая оценка элементов матрицы **C** для биективных отображений [59]: $c^{(\max)} \geq 1/2^{n-2}$ и, соответственно, $q^{(\max)} \geq 1/2^{n-1}$.

Связь между матрицами **P** и **C**, характеризующими разностные и линейные свойства блочных шифров и тем самым определяющими их стойкость к ДКА и ЛКА, задается следующими соотношениями *(см. разд. 4.2.3)*:

$$p_{i,j} = \frac{1}{2^n} \sum_{\forall \beta_k, \beta_m \in GF(2)^n} (-1)^{\beta_m \bullet \alpha_i \oplus \beta_k \bullet \alpha_j} c_{k,m}^2, \quad c_{k,m}^2 = \frac{1}{2^n} \sum_{\forall \alpha_k, \alpha_m \in GF(2)^n} (-1)^{\beta_m \bullet \alpha_i \oplus \beta_k \bullet \alpha_j} p_{i,j}.$$

Указанные формулы показывают двойственность и глубокую взаимосвязь ДКА и ЛКА.

Поскольку элементы матрицы **P** связаны с квадратами элементов матрицы **C**, то часто вместо последней используют матрицу $\mathbf{C}' = \| c_{i,j}^2 \|$. С учетом этих замечаний для криптографических преобразований вида $\mathrm{GF}(2)^n \to \mathrm{GF}(2)^m$ справедливы следующие соотношения между матрицей **P**, матрицей **C′** и автокорреляционной матрицей **R** *(см. разд. 4.2.5)* [136], имеющие важное практическое значение:

$\mathbf{C}' = \mathbf{H}_n \mathbf{P} \mathbf{H}_m$, $\mathbf{C}' = \mathbf{H}_n \mathbf{R}$, $\mathbf{R} = \mathbf{P} \mathbf{H}_m$, где $\mathbf{H}_n$ и $\mathbf{H}_m$ — матрицы Уолша-Адамара из *разд. 4.2.3*.

Для блочных шифров, соответствующих структуре подстановочно-перестановочных сетей *(**SPN**-structure)*, таких, как победитель конкурса AES алгоритм Rijndael, большое прикладное значение имеют следующие результаты относительно их стойкости к ДКА и ЛКА [71].

Пусть раундовое преобразование блочного шифра имеет слой ***S*** из серии **S**-блоков ($\mathbf{S}_{k \times m}$), т. е. состоит из k **S**-блоков по m битов каждый *(substitution layer)*, а также содержит рассеивающий слой ***D*** *(diffusion layer)*, обеспечивающий распространение изменений или лавинный эффект.

Под *"дифференциально активными* **S***-блоками"* будем понимать блоки, на вход которых при проведении ДКА поступают ненулевые разности (дифференциалы). Соответственно, *"линейно активными* **S***-блоками"* будут считаться блоки, на выходе которых при проведении ЛКА используются ненулевые маски.

Дифференциальным $\mathbf{H}_d(x)$ (или линейным $\mathbf{H}_l(x)$) весом вектора x, проходящего через слой ***S***, будем называть количество дифференциально (или линейно) активных **S**-блоков, соответствующих этому вектору.

Рассмотрим структуру, состоящую из двух слоев ***SD***, с учетом ее периодической повторяемости при блочном шифровании. Пусть $\Delta x = x \oplus x^*$ и $\Delta y = \boldsymbol{D}(x) \oplus \boldsymbol{D}(x^*)$. Обозначим через a_x и b_y, соответственно, входные и выходные маски ***SD***-структуры. Будем понимать под дифференциальным (или линейным) весом ***SD***-структуры значение функции $n_d(\boldsymbol{D}) = \min\limits_{\Delta x \neq 0} \mathbf{H}_{\boldsymbol{d}}(\Delta x) \oplus \mathbf{H}_{\boldsymbol{d}}(\Delta y)$ (или $n_l(\boldsymbol{D}) = \min\limits_{\Delta x \neq 0} \mathbf{H}_{\boldsymbol{l}}(a_x) \oplus \mathbf{H}_{\boldsymbol{l}}(b_y)$).

Тогда, если раундовые ключи такого блочного шифра независимы и распределены равновероятно, а также выполняется условие $n_d(\boldsymbol{D}) = n + 1$ ($n_l(\boldsymbol{D}) = n + 1$), то вероятность любой дифференциальной характеристики ***SD***-структуры не превосходит $(p_S)^k$ (а вероятность любой линейной аппроксимации не превосходит $(q_S)^k$), где p_S и q_S, соответственно, вероятности наиболее возможных дифференциальных характеристик и линейных аппроксимаций среди всех **S**-блоков. Для блочного шифра, содержащего r ***SD***-структур, указанные вероятности имеют, соответственно, значения $(p_S)^{rk}$ и $(q_S)^{rk}$.

Таким образом, предлагаемые оценки определяют, как качество **S**-блоков и степень распространения изменений в блочных шифрах, образующих подстановочно-перестановочные сети, влияют на стойкость этих шифров к ДКА и ЛКА.

Из приведенных выше рассуждений относительно общих оценок стойкости блочных алгоритмов к ЛКА становятся понятными трудности, с которыми может столкнуться криптоаналитик или разработчик блочных шифров, содержащих сложные криптографические примитивы. Поэтому по аналогии с ДКА оценка стойкости блочного шифра к ЛКА проводится, как правило, на основе некоторой совокупности реально просчитываемых линейных характеристик.

Обычно практически применимые линейные характеристики всего шифра являются цепочками итеративных линейных характеристик, т. е. небольшими периодически повторяющимися последовательностями однораундовых линейных характеристик. Маски этих характеристик так же, как и дифференциальные характеристики в ДКА, имеют, как правило, небольшой вес Хэмминга или какой-то специальный вид. Звенья этих цепочек увязываются и оцениваются по вероятности в соответствии с *правилами согласования*, представленными в *разд. 3.4.1*.

4.4.5. Влияние управляемых подстановочных операций на стойкость блочных шифров к линейному криптоанализу

Так же, как и в случае дифференциального криптоанализа *(см. разд. 4.4.3)*, в качестве иллюстрации механизма влияния управляющих подстановочных операций на стойкость блочных шифров к линейному криптоанализу (ЛКА) рассмотрим вариант замены операции "⊕" в схеме Фейстеля на УПО, т. е. на подстановку

$\sigma_F^{(V)}(X_R)$ (см. рис. 4.2 и 4.3). Модифицированную схему можно условно представить как традиционную схему Фейстеля с операцией "⊕", но с новой раундовой функцией следующего вида: $F_\sigma = F_\sigma(X, K) = F_\oplus(X_L, K) \oplus \boldsymbol{\varphi}^{(V)}$, где $F_\oplus = F_\oplus(X_L, K) = F(X_L, K)$ — раундовая функция исходного алгоритма. Если компоненты векторной функции $\boldsymbol{\varphi}^{(V)}$ являются бент-функциями или удовлетворяют условию (4.20), то в соответствии со свойствами максимальной или высокой нелинейности этих функций раундовую функцию F_σ можно сконфигурировать таким образом, что $\mathbf{N}(F_\sigma) > \mathbf{N}(F_\oplus)$, а тогда $\mathbf{L}(F_\sigma) < \mathbf{L}(F_\oplus)$. Отсюда в соответствии с (4.22) обеспечивается увеличение трудоемкости $\mathbf{M_L}(r)$ проведения ЛКА. Следовательно, использование подстановки $\sigma_F^{(V)}(X_R)$ вместо операции "⊕" может существенно повысить стойкость исходной схемы Фейстеля к ЛКА, при этом уровень стойкости зависит от конструкции векторной функции $\boldsymbol{\varphi}^{(V)}$.

Рассмотрим в качестве примера образующие булевы функции из *разд. 4.4.3*, т. е. $\forall i \in \{1, \ldots, n/2\}$ $\varphi_i = x_{i-1}^{(R)} v_{2i-1} \oplus a_{i-1} v_{2i}$, где $V = V(X_L, K)$ ($V \in \mathrm{GF}(2)^n$) — некоторая функция расширения ключа и $F_\oplus = (a_1, \ldots, a_{n/2}) \in \mathrm{GF}(2)^{n/2}$. Для оценки нелинейности функции $F_\sigma(X, K)$ в соответствии с общим определением нелинейности векторных булевых функций из *разд. 4.2.4* $\forall \beta \in \mathrm{GF}(2)^{n/2}$ рассмотрим произвольную линейную комбинацию ее компонентов:

$$\lambda^{(\beta)} = \lambda^{(\beta)}(X, K) = \beta \bullet F_\sigma = \beta \bullet F_\oplus \oplus \beta \bullet \boldsymbol{\varphi}^{(V)} =$$
$$= \sum_{i=1}^{n/2} \beta_i a_i \oplus \sum_{i=1}^{n/2} \beta_i (x_{i-1}^{(\mathrm{R})} v_{2i-1} \oplus a_{i-1} v_{2i});$$

$$\beta \bullet \varphi^{(V)} = \beta_1 (x_0^{(R)} v_1 \oplus a_0 v_2) \oplus \sum_{i=2, \beta_i \neq 0}^{\frac{n}{2}-1} \beta_i (x_{i-1}^{(R)} v_{2i-1} \oplus a_{i-1} v_{2i}).$$

Результирующая булева функция $\lambda_{bent}^{(\beta)} \sum_{i=2, \beta_i \neq 0}^{\frac{n}{2}-1} \beta_i (x_{i-1}^{(\mathrm{R})} v_{2i-1} \oplus a_{i-1} v_{2i})$ является бент-функцией (при условии, что хотя бы один коэффициент $\beta_2, \ldots, \beta_{n/2}$ отличен от нуля), так как она образована суммой квадратичных бент-функций с независимыми аргументами. Таким образом, если начальные условия $x_0^{(R)} = 1$ и $a_0 = 1$, то $\lambda^{(\beta)} = \beta \bullet F_\oplus \oplus v_1 \oplus v_2 \oplus \lambda_{bent}^{(\beta)}$ при $\beta_1 = 1$ или $\lambda^{(\beta)} = \beta \bullet F_\oplus \oplus \lambda_{bent}^{(\beta)}$ при $\beta_1 = 0$. Нелинейность функции $\lambda_{bent}^{(\beta)}$ зависит от количества отличных от нуля коэффициентов $\beta_2, \ldots, \beta_{n/2}$. Если число таких ненулевых коэффициентов $m \in \{1, \ldots, (n/2) - 1\}$, то в соответствии с утверждением 4.6 из *разд. 4.2.4* для нелинейности функции $\lambda^{(\beta)}$ при $\beta_1 = 1$ имеет место следующая оценка:

$$\mathbf{N}_1^{(m)}(\lambda^{(\beta)}) \geq \mathbf{N}(\beta \bullet F_{\oplus})2^{4m+2} + \mathbf{N}(v_1 \oplus v_2 \oplus \lambda_{bent}^{(\beta)})2^{\rho_\beta} - 2\mathbf{N}(\beta \bullet F_{\oplus})\mathbf{N}(v_1 \oplus v_2 \oplus \lambda_{bent}^{(\beta)}) =$$

$$= \mathbf{N}(\beta \bullet F_{\oplus})2^{4m+2} + (2^{4m+1} - 2^{2m+1})2^{\rho_\beta} - 2\mathbf{N}(\beta \bullet F_{\oplus})(2^{4m+1} - 2^{2m+1}) =$$

$$= \mathbf{N}(\beta \bullet F_{\oplus})2^{2m+2} + (2^{4m+1} - 2^{2m+1})2^{\rho_\beta},$$

где ρ_β — количество различных переменных в полиноме $\beta \bullet F_{\oplus}$. Кроме того, из утверждения 4.6 справедливо неравенство: $\mathbf{N}_1^{(m)}(\lambda^{(\beta)}) \geq \mathbf{N}(\beta \bullet F_{\oplus})2^{4m+2}$.

При $\beta_1 = 0$ оценка нелинейности приобретает следующий вид:

$$\mathbf{N}_0^{(m)}(\lambda^{(\beta)}) \geq \mathbf{N}(\beta \bullet F_{\oplus})2^{4m} + \mathbf{N}(\lambda_{bent}^{(\beta)})2^{\rho_\beta} - 2\mathbf{N}(\beta \bullet F_{\oplus})\mathbf{N}(\lambda_{bent}^{(\beta)}) =$$

$$= \mathbf{N}(\beta \bullet F_{\oplus})2^{2m} + (2^{4m-1} - 2^{2m-1})2^{\rho_\beta}.$$

Кроме того, имеет место неравенство: $\mathbf{N}_0^{(m)}(\lambda^{(\beta)}) \geq \mathbf{N}(\beta \bullet F_{\oplus})2^{4m}$.

Поскольку в ЛКА рассматриваются лишь ненулевые значения векторов β, то из приведенных оценок следует, что только в одном случае, когда все коэффициенты $\beta_2, \ldots, \beta_{n/2}$ равны нулю и $\beta_1 = 1$, нелинейность функции $\lambda^{(\beta)}$ не изменится при использовании УПО вместо операции "$\oplus$", т. е. $\mathbf{N}_1^{(m)}(\lambda^{(\beta)}) = \mathbf{N}(f_{\oplus}^{(1)})$, где $f_{\oplus}^{(1)}$ — первый компонент вектора $F_{\oplus}$. Во всех других случаях нелинейность новой раундовой функции будет больше исходной.

Для устранения указанного выше нежелательного случая необходимо, чтобы начальные условия $x_0^{(R)}$ и a_0 не являлись константами, а формировались в результате некоторой процедуры расширения ключа $V = V(X_L, K)$. В этом случае оценка нелинейности функции $\lambda^{(\beta)}$ всегда будет иметь следующий вид:

$$\mathbf{N}^{(m)}(\lambda^{(\beta)}) \geq \mathbf{N}(\beta \bullet F_{\oplus})2^{4m} + (2^{4m-1} - 2^{2m-1})2^{\rho_\beta} - 2\mathbf{N}(\beta \bullet F_{\oplus})(2^{4m-1} - 2^{2m-1}) =$$

$$= \mathbf{N}(\beta \bullet F_{\oplus})2^{2m} + (2^{4m-1} - 2^{2m-1})2^{\rho_\beta},$$

где $m \in \{1, \ldots, (n/2) - 1\}$ — число ненулевых коэффициентов $\beta = (\beta_1, \beta_2, \ldots, \beta_{n/2})$.

В соответствии с определением 4.15 нелинейности векторной булевой функции из *разд. 4.2.4* имеем: $\mathbf{N}(F_{\oplus}) = \min\limits_{\forall \beta \in GF(2)^{n/2}} \mathbf{N}(\beta \bullet F_{\oplus}) = \boldsymbol{D}$.

Тогда

$$\mathbf{N}(F_{\oplus}) = \min_{\forall \beta \in GF(2)^{n/2}} \mathbf{N}(\lambda^{(\beta)}) = \mathbf{N}(\lambda^{(\beta^{(\sigma)})}) \geq 2^{2m}\mathbf{N}(\beta^{(\sigma)} \bullet F_{\oplus}) + (2^{4m-1} - 2^{2m-1})2^{\rho_\beta(\sigma)} \geq$$

$$\geq 2^{2m}D + (2^{4m-1} - 2^{2m-1})2^{\rho_\beta^{(\sigma)}},$$

где $\beta^{(\sigma)}$ — вектор, на котором достигается минимум нелинейности $\mathbf{N}(\lambda^{(\beta)})$. При этом средняя нелинейность преобразования $\mathbf{N}'(F_\sigma)$ в соответствии с определением 4.16 имеет следующее значение:

$$\mathbf{N}'(F_\sigma) = \frac{1}{2^{n/2}-1} \sum_{\forall \beta \in GF(2)^{n/2}, \beta \neq 0} \mathbf{N}(\lambda^{(\beta)}) = N'(F_\oplus) \oplus \frac{1}{2^{n/2}-1} \sum_{\forall \beta \in GF(2)^{n/2}, \beta \neq 0} \mathbf{N}(\beta \bullet \varphi^{(V)}) =$$

$$= \mathbf{N}'(F_\oplus) \oplus \frac{1}{2^{n/2}-1} \sum_{m=1}^{n/2} \binom{n/2}{m} (2^{4m-1} - 2^{2m-1}).$$

Таким образом, в случае замены в схеме Фейстеля операции "$\oplus$" на УПО $\sigma_F(X_R)$, где компоненты векторной функции $\boldsymbol{\varphi}^{(V)}$ являются простейшими бент-функциями вида $\varphi_i = x_{i-1}^{(R)} v_{2i-1} \oplus a_{i-1} v_{2i}$, справедливо соотношение $\mathbf{N}(F_\sigma) >> \mathbf{N}(F)$ ($\mathbf{N}'(F_\sigma) >> \mathbf{N}'(F_\oplus)$) и, следовательно, $\mathbf{L}(F_\sigma) << \mathbf{L}(F_\oplus)$ ($\mathbf{L}'(F_\sigma) << \mathbf{L}'(F_\oplus)$), т. е. в соответствии с результатами из *разд. 4.4.4* весь блочный алгоритм становится более стойким к ЛКА.

Как и в случае с ДКА, аналогичные результаты могут быть получены и для более сложных векторных функций $\boldsymbol{\varphi}^{(V)}$ с зависимыми компонентами. Например, в качестве образующих булевых функций для $\boldsymbol{\varphi}^{(V)}$ можно рассматривать функции, удовлетворяющие условию (4.20).

В заключение раздела необходимо еще раз подчеркнуть, что практически применимые результаты ЛКА так же, как и в случае ДКА, могут быть получены лишь на основе конкретного изучения взаимодействия всех составляющих элементов раундовых преобразований, т. е. особенностей сочетания всех криптографических примитивов с УПО.

4.4.6. Влияние управляемых операций на стойкость блочных шифров к техническим атакам

Одним из основных недостатков используемых в настоящее время стандартов блочных шифров является их уязвимость по отношению к реальным техническим методам криптоанализа, в частности к атакам на основе *генерации аппаратных ошибок*. Суть данного метода заключается в использовании тепловых, высокочастотных, ионизирующих или других типов внешних воздействий на защитную аппаратуру с целью получения искажения битовой информации в определенных местах алгоритма преобразований. Методы анализа таких искажений позволяют сравнивать данные, полученные до и после внесения искажений, и получать значение раундового ключа на последнем раунде и далее на всех раундах алгоритма. Проведенные в недавнее время исследования блочных шифров DES и ГОСТ 28147-89 показали их нестойкость к указанному типу атаки [29].

Попытки усложнить исходный алгоритм DES привели к созданию тройного DES (Triple DES), что несколько повысило его стойкость к математическим методам криптоанализа, но не избавило от уязвимости к методу генерации аппаратных ошибок. Указанные выше недостатки можно отнести и к другим блочным алгоритмам, созданным за последнее время. Таким образом, к концу 1990-х годов стали очевидными слабости имеющихся блочных шифров по ряду параметров, и появилась настоятельная необходимость создания новых блочных алгоритмов, стойких к атакам на основе генерации аппаратных ошибок.

В соответствии с базовым методом криптоанализа на основе генерации аппаратных ошибок стойкость шифра к этому виду атаки во многом определяется вычислительной сложностью тотального перебора значения раундового ключа K_r на последнем раунде алгоритма. Действительно, шифрование на последнем раунде блочного шифра, спроектированного по схеме Фейстеля, определяется следующими соотношениями:

- $X_R = X_R \oplus F(X_L, K_r)$ — для преобразования без ошибки;
- $X'_R = X_R \oplus F(X'_L, K_r)$ — для преобразования с генерируемой ошибкой.

Складывая два уравнения, получаем следующее уравнение:

$$X_R \oplus X'_R = F(X_L, K_r) \oplus F(X'_L, K_r).$$

Можно построить систему таких уравнений, каждое из которых соответствует различным блокам открытого и закрытого текстов. Независимо от вида функции F реально нахождение единственного решения системы, т. е. вскрытие раундового ключа K_r методом полного (тотального) перебора. Иногда возможно создание более эффективного метода, как в случае блочного шифра RC5, который подробно рассматривается в *разд. 6.7.1*.

Оценка верхней границы вычислительной сложности вскрытия раундового ключа K_r тотальным перебором имеет вид: $\mathbf{M}_{K_r} = 2^m h \mathbf{M}_F$, где h — число уравнений в системе, $\mathbf{M}_F$ — сложность вычисления значения раундового преобразования, m — длина ключа.

Аналогичным образом находятся значения остальных раундовых ключей. Верхняя граница вычислительной сложности вскрытия всей таблицы из r раундовых ключей имеет следующую оценку: $\mathbf{M} = 2^m h \mathbf{M}_F r$.

Таким образом, трудоемкость вскрытия любого блочного шифра, построенного по схеме Фейстеля, пропорциональна трудоемкости перебора значения одного раундового ключа (2^m). Вышесказанное позволяет сделать вывод о необходимости введения совокупности следующих требований к раундовому преобразованию блочного шифра, построенного по схеме Фейстеля:

- увеличение размера блока преобразуемых шифром данных до 128 и более битов;

- увеличение общего размера раундового ключа;
- обеспечение неразделимости элементов раундового ключа в рамках одного раунда алгоритма;
- применение в блочных шифрах специальных методов, препятствующих проведению математического и технического анализа, в частности введение дополнительных преобразований в начале алгоритма и после завершающего раунда.

Предварительный анализ некоторых алгоритмов, представленных в качестве претендентов AES, показал, что раундовые ключи длиной 128 битов в этих блочных шифрах разбиваются на части, и задача вскрытия сводится к перебору значений более коротких подключей, размер которых составляет от 32 до 64 битов, что обусловлено удобством реализации и требованием к скорости.

Новые блочные шифры на базе управляемых операций в полной мере удовлетворяют указанным выше требованиям и обеспечивают необходимый уровень стойкости к реальным техническим методам криптоанализа. В качестве новых механизмов преобразования информации в таких алгоритмах используется гибкая выборка раундовых ключей в зависимости от преобразуемых данных на основе новых примитивов преобразования — блоков управляемых перестановок и управляемых подстановочных операций. В соответствии с рекомендациями по повышению стойкости к методу генерации аппаратных ошибок в предлагаемых шифрах:

- блоки данных имеют размер 64, 128 и более битов;
- в одном раунде используется не один, а не менее трех неразделимых ключей общей длиной более 192 битов;
- применяются специальные начальное и конечное преобразования данных.

Более подробно практические вопросы криптоанализа блочных шифров на основе метода генерации аппаратных ошибок рассматриваются в *разд. 6.7*.

Другим видом технической атаки на блочные шифры является метод *замера мощности излучения* шифраппаратуры. Целью атаки является получение информации о ключах, используемых в регистрах памяти компьютера или в микросхемах в ходе работы блочного шифра. Предлагаемые новые блочные алгоритмы на базе управляемых операций, как правило, предполагают аппаратную реализацию, при которой ключи вводятся один раз во время инициализации алгоритма, и в дальнейшем изменения мощности излучения происходят только в области преобразуемых данных, что препятствует получению информации о применяемых ключах. Кроме того, сама аппаратная реализация допускает возможность создания специальных экранирующих средств, препятствующих или сильно ослабляющих как побочные излучения аппаратуры, так и внешние воздействия на шифрсистему.

4.4.7. Обзор результатов криптоанализа блочного шифра ГОСТ 28147-89 по открытым публикациям

Российский стандарт блочного шифрования ГОСТ 28147-89 (далее просто ГОСТ), принятый в 1989 году [10], является примером DES-подобных криптосистем, созданных по классической итерационной схеме Фейстеля.

Напомним основные характеристики блочного шифра ГОСТ:

- размер блока преобразуемых данных — 64 бита;
- количество раундов — 32;
- секретный ключ имеет размер 256 битов и состоит из последовательности восьми 32-битовых слов $\boldsymbol{K} = (K_1, K_2, \ldots, K_8)$;
- алгоритм формирования рабочих ключей не использует специальной процедуры и задается на первых 24 раундах циклическим повторением восьми 32-битовых слов секретного ключа $\boldsymbol{K}$. На последних 8-ми раундах указанные восемь слов секретного ключа $\boldsymbol{K}$ используются в обратном порядке;
- рабочие ключи K_i в каждом раунде перемешиваются с 32-битовым подблоком преобразуемого текста посредством операции сложения по модулю 2^n;
- нелинейное преобразование в каждом раунде обеспечивается за счет восьми *секретных* биективных **S**-блоков размера 4×4. (**S**-блоки в ГОСТ являются долговременным ключом);
- в качестве перестановочного преобразования в каждом раунде применяется циклический сдвиг на 11 позиций в сторону старших разрядов.

Криптографические свойства алгоритма ГОСТ 28147-89 были проанализированы в ряде открытых публикаций [29, 47, 53, 76]. Среди основных результатов анализа ГОСТ можно выделить следующие положения и выводы.

В работах [53, 76] обращается внимание на тот факт, что секретность алгоритма ГОСТ основана не только на секретности ключа, но и на секретности подстановочных элементов шифра (**S**-блоков), что может привести к криптографически слабым модификациям алгоритма. Такое может произойти, например, если в качестве **S**-блоков будут применены высоко линейные преобразования и/или преобразования, слабые с точки зрения дифференциального криптоанализа.

При использовании секретных **S**-блоков алгоритма ГОСТ следует учитывать, что даже в случае высокой нелинейности **S**-блоков существует возможность их неблагоприятного сочетания с операцией сложения по модулю 2^n, что может уменьшать показатели нелинейности раундового преобразования и тем самым снижать стойкость модификаций алгоритма ГОСТ к линейному криптоанализу [53].

На основе слабостей структуры алгоритма ГОСТ в работе [76] предложен вариант атаки на базе сочетания *метода относительных ключей* (related keys) и диф-

ференциального криптоанализа, позволяющей отличить 32-раундовое преобразование алгоритма от случайной подстановки при условии специально подобранных секретных **S**-блоков. В случае модификации ГОСТ в 24-раундовый алгоритм указанная атака при наличие слабых **S**-блоков обеспечивает реальную возможность для определения секретных ключей шифра.

В работе [47] предлагается новый подход к анализу ГОСТ и других блочных алгоритмов, использующих в качестве рабочих раундовых ключей периодически повторяющуюся последовательность секретных ключей. Указанный метод криптоанализа, получивший название *"скользящей атаки"* (slide attack) , не зависит от количества раундов и основывается на так называемом *"парадоксе дней рождений"*.

Поскольку операция XOR отличается от операции сложения по модулю 2^n лишь отсутствием бита переноса, влияние которого имеет экспоненциальное затухание, то в качестве модели ГОСТ в предлагаемом варианте атаки [47] рассматривается его модификация (ГОСТ-XOR), содержащая операцию XOR вместо сложения по модулю 2^n. Другой анализируемой модификацией ГОСТ (ГОСТ-Н) является шифр, в котором на последних 8-ми раундах применяется рабочий ключ, формируемый по тем же правилам, что и на первых 24-х раундах.

Показано, что ГОСТ-XOR-Н имеет 2^{128} *особых ключей* вида $\boldsymbol{K} = (A, B, C, D, A, B, C, D)$, позволяющих эффективно атаковать такой шифр классической скользящей атакой при любом количестве раундов и любых **S**-блоках на основе использования 2^{32} известных пар открытых и шифрованных текстов или 2^{16} подобранных текстов. Наличие указанных особых ключей достаточно для эффективного криптоанализа последних 20-ти раундов ГОСТ-XOR посредством *усовершенствованной "скользящей атаки" с инверсией ключей* [47]. Для успешного вскрытия секретного ключа алгоритма в этом случае необходимо всего 2^{33} известных пар открытых и шифрованных текстов.

Результаты криптоанализа различных модификаций ГОСТ посредством разновидностей "скользящей атаки" являются свидетельством серьезных слабостей архитектуры этого шифра. В работе [47] предлагается дальнейшее направление совершенствования методов "скользящей атаки" на основе ее сочетания с классическими методами дифференциального криптоанализа. Предполагается, что данный метод может в перспективе в значительной степени продвинуть достигнутые успехи криптоанализа алгоритма ГОСТ.

В работе [47] так же отмечено, что блочный шифр ГОСТ имеет 2^{128} *слабых ключей* вида $\boldsymbol{K} = (A, B, C, D, D, C, B, A)$. В этом случае алгоритм инвариантен относительно изменения на противоположный порядка применения ключей, т. е. процессы зашифрования и расшифрования совпадают (алгоритм выполняет инволюцию). Для указанных ключей повторное зашифрование шифрованного текста приводит к открытому тексту. Кроме того, для каждого из таких 2^{128} ключей существует 2^{32} текстов, инвариантных относительно преобразования шифром ГОСТ. Данные обстоятельства указывают на проблематичность создания надеж-

ных хэш-функций на базе алгоритма ГОСТ, поскольку имеются объективные условия получения *коллизий*, традиционно применяемых для атаки на хэш-функции.

В работе [29] предложен анализ блочного шифра ГОСТ на основе достаточно реальной в современных условиях атаки, использующей возможность внесения простейших *аппаратных ошибок* при выполнении алгоритма (*см. разд. 4.4.6*). В соответствии с приведенными в работе оценками указанная атака на всю ключевую систему блочного шифра ГОСТ имеет вычислительную сложность не более 2^{31} простых операций, что в условиях, благоприятных для указанного вида атаки, может повлечь катастрофические последствия.

Следует отметить, что блочный алгоритм ГОСТ как стандарт шифрования разрабатывался в 70—80-х годах прошлого века и был ориентирован на программную и аппаратную реализацию, соответствующую невысокому уровню микроэлектронной технологии того времени. Принципы преобразования блоков данных с разделением их на мелкие подблоки, применение криптографически несовершенных операций и большое количество раундов не могут обеспечивать необходимый в настоящее время уровень быстродействия алгоритма (более 2 Гбит/сек) при невысокой стоимости его реализации. Эти обстоятельства и указанные выше криптографические слабости алгоритма ГОСТ не позволяют рассматривать его как перспективный блочный шифр, предназначенный для использования в современных высокоскоростных средствах защиты информации, работающих в реальном масштабе времени.

4.5. Практическое использование управляемых подстановочных операций

4.5.1. Общие требования к блочным шифрам и управляемые подстановочные операции

Результат действия произвольного блочного шифра можно представить как множество фиксированных подстановок $Y = \mathbf{C}_K(X)$, выбираемых в зависимости от секретного ключа и заданных на множестве всех входных блоков данных некоторой фиксированной длины n ($Y, X \in \mathrm{GF}(2)^n$). Число различных входных блоков составляет 2^n, при этом число всех обратимых подстановок — $2^n!$. Табличное описание даже одной такой подстановки, например над 2^{64} элементами, практически невыполнимо. В связи с этим в блочном шифре выбор подстановок в зависимости от секретного ключа должен осуществляться алгоритмическим способом. Обратное преобразование при этом проводится некоторым алгоритмом $X = \mathbf{D}_K(Y)$. Наиболее перспективными являются шифры, для которых прямое и обратное преобразование выполняется одним и тем же алгоритмом, но с различным порядком использования ключей. Такие шифры можно представить как параметрически обратимые подстановки вида:

раметрически обратимые подстановки вида: $Y = \mathbf{C}_{\boldsymbol{K}_e}(X)$, где параметр $e \in \{0,1\}$ определяет различный порядок использования ключей для прямого ($e = 0$) и обратного ($e = 1$) преобразования. Основная задача при разработке алгоритма $Y = \mathbf{C}_{\boldsymbol{K}}(X)$ состоит в том, чтобы каждая из подстановок, задаваемая всеми возможными секретными ключами, была псевдослучайной. Последнее означает, что знание большого числа соответствующих друг другу пар исходных блоков данных (исходных текстов) и выходных блоков (закрытых текстов) не позволяет вычислить секретный ключ или определить с существенной вероятностью значение исходного текста по закрытому тексту.

Для решения этой задачи используются специальные, как правило, итеративные процедуры, размножающие ошибки. Изменение одного входного бита в таких процедурах влечет изменение нескольких или всех битов блока выхода. Последовательное использование большого числа преобразований, обладающих таким свойством, позволяет построить блочные шифры, в которых изменение одного произвольного бита на входе приводит к тому, что с вероятностью близкой к 0,5 изменяется каждый из выходных битов. Для реализации этого свойства, которое получило название эффекта распространения ошибок или изменений (*см. разд. 4.2.5 и 4.2.6*), а также свойств нелинейности и корреляционной иммунности шифров *(см. разд. 4.2.4 и 4.2.7)*, широко используются операции подстановки над подблоками данных малого размера и некоторые другие операции. Например, в российском стандарте ГОСТ 28147-89 [10] используются фиксированные подстановки над 4-битовыми данными и операция сложения по модулю 2^n.

В настоящее время подход, связанный с использованием подстановок малого размера, следует считать несколько устаревшим. Современные разработчики блочных шифров широко применяют целый спектр различных криптографических примитивов: подстановки повышенного размера; операцию умножения по модулю 2^n; отображения над полями Галуа по модулю неприводимых многочленов; линейные преобразования; операции циклического сдвига, осуществляемые в зависимости от преобразуемых данных, и др. Такое разнообразие в применении криптографических примитивов связано с множеством требований, предъявляемых к блочным шифрам. Элементная база проектируемых алгоритмов и выбор параметров сложности криптографических примитивов зависят от целевого предназначения блочного шифра и, в частности, связаны с направленностью алгоритмов на аппаратную или программную реализацию.

Как показывает практика последних лет, новые блочные шифры, формируемые на смену существующим стандартам, прежде всего, ориентируются на программную реализацию, и лишь во вторую очередь рассматриваются вопросы их оптимизации под реализацию в аппаратном виде. В данной книге помимо традиционного предлагается и альтернативный подход в создании блочных шифров, при котором изначально алгоритмы ориентируются, прежде всего, на аппаратное выполнение. Такой подход в настоящее время считается достаточно перспектив-

ным. Наряду с уже известными в открытой литературе криптографическими примитивами в данном разделе предлагаются варианты практического использования в блочных шифрах управляемых подстановочных операций, являющихся новыми примитивами нелинейного преобразования данных. В соответствии с результатами исследований, проведенных в *разд. 4.3* и *4.4*, использование УПО в сочетании с другими преобразованиями, в частности с управляемыми перестановочными операциями, позволяет существенно улучшить качество блочных алгоритмов за счет увеличения их статистической неопределенности (робастности), что повышает их устойчивость к различным криптоатакам. Необходимо отметить, что увеличение робастности алгоритмов явилось одним из основных требований, предъявляемых к претендентам на новый стандарт блочного шифрования на конкурсе AES, организованном Национальным институтом по стандартам США (NIST). Такой подход обусловлен тем обстоятельством, что требование повышения неопределенности преобразований связано с необходимостью потенциального обеспечения высокой стойкости шифров не только к известным, но и к новым методам криптоанализа на фоне активно развивающихся исследований в этой области.

4.5.2. Оценки сложности реализации управляемых подстановочных операций

Положительным свойством целых классов УПО является возможность их быстрой аппаратной и в ряде случаев эффективной программной реализации при сохранении достаточно хороших криптографических свойств. Необходимо заметить, что сложность преобразования, осуществляемого УПО, непосредственно зависит от разнообразия и сложности образующих УПО булевых функций. Как было отмечено в *разд. 4.3.10*, УПО имеют двойственный характер, т. е. занимают промежуточное положение между **S**-блоками большого размера и нелинейными операциями. Это свойство контрастно проявляется на примере следующих трех базовых моделей УПО, наиболее перспективных для практической реализации:

- биективной последовательной модели $\mathbf{S}'_\sigma$ *(см. разд. 4.3.3)*;
- небиективной последовательной модели $\overline{\mathbf{S}}_\sigma$ *(см. разд. 4.3.6)*;
- параметрически обратимой рекурсивной модели $\mathbf{\Omega}'_{\sigma,e}$ и, в частности, ее наиболее разработанной подмодели $\mathbf{\Omega}''_{\sigma,e}$ *(см. разд. 4.3.4)*.

Если рассматривать варианты УПО моделей $\mathbf{S}'_\sigma$ и $\overline{\mathbf{S}}_\sigma$, для которых все прототипы образующих функций различны и имеют достаточно сложное аналитическое представление, то по своей сложности такие УПО соответствуют последовательности **S**-блоков, каждый из которых имеет длину, равную среднему количеству переменных в образующих УПО булевых функциях. Как правило, на практике

серии из k **S**-блоков длины n_s ($n = k{\times}n_s$) реализуются табличным образом и требуют $\mathbf{M}_S = k2^{n_s}$ битов памяти. Подобная реализация УПО, аналогично, потребует в среднем $\mathbf{M}_{\text{УПО}} \approx m2^{n'}$ битов памяти, где m — количество различных прототипов образующих булевых функций ($1 \le m \le n$), и n'— среднее число переменных в этих функциях. Если УПО моделей $\mathbf{S}'_\sigma$ или $\overline{\mathbf{S}}_\sigma$, т. е. управляемый схемный преобразователь (УСП), имеет образующие функции лишь одного типа ($m = 1$), то кроме простого табличного задания данная УПО может быть представлена в перспективном для аппаратной реализации виде. Основной характеристикой такой реализации служит величина $\mathbf{T}_{\text{УСП}}$ — время задержки $kt_\oplus$ на выполнение УСП, выраженное в единицах времени выполнения операции "$\oplus$". Сказанное можно пояснить на следующем примере.

Пусть УСП $Y = \boldsymbol{G}_V(X, A)$: $(\mathrm{GF}(2)^{32}{\times}\mathrm{GF}(2)^{32}{\times}\mathrm{GF}(2)^{32} \to \mathrm{GF}(2)^{32})$ имеет лишь один прототип образующих функций *(см. разд. 4.3.3)* и сами функции имеют вид:

$$y_i = f_i^{(V)}(X, A) =$$
$$= x_i{\oplus}v_i{\oplus}x_{i-1}a_i{\oplus}v_{i-1}a_{i-1}{\oplus}v_{i-2}x_{i-3}{\oplus}x_{i-2}a_{i-2}{\oplus}a_{i-3}x_{i-4}{\oplus}x_{i-1}v_{i-2}a_{i-3}{\oplus}x_{i-1}v_{i-1}x_{i-2}a_{i-3},$$

где x_j, a_j, v_j — компоненты, соответственно, векторов $X, A, V{\in}\mathrm{GF}(2)^{32}$ и $(x_{-3}, x_{-2}, x_{-1}, x_0) = (1, 1, 1, 1)$, $(a_{-2}, a_{-1}, a_0,) = (1, 1, 1)$, $(v_{-1}, v_0) = (1, 1)$ — фиксированные начальные условия или биты расширенного ключа. Преобразование, осуществляемое такой УПО, можно представить в следующем векторном виде:

$$Y = X{\oplus}V{\oplus}X^{\to 1}A{\oplus}V^{\to 1}A^{\to 1}{\oplus}V^{\to 2}X^{\to 3}{\oplus}X^{\to 2}A^{\to 2}{\oplus}A^{\to 3}X^{\to 4}{\oplus}X^{\to 1}V^{\to 2}A^{\to 3}{\oplus}X^{\to 1}V^{\to 1}X^{\to 2}A^{\to 3},$$

где операция "$\to i$" обозначает сдвиг вектора вправо на i позиций, а запись вида XA — побитовое векторное умножение по модулю 2. Для биективных моделей УПО младшие разряды сдвигаемых векторов заполняются битами начальных условий или битами расширенного ключа. В случае небиективной модели УПО с циклическим сдвигом исходных данных *(см. разд. 4.3.6)* младшие разряды сдвигаемых векторов циклически заполняются старшими разрядами этих векторов. Несмотря на достаточно высокую сложность образующих данное УСП булевых функций, аппаратно преобразование выполняется сразу над всем блоком данных всего за 5 тактов ($T_{\text{УСП}} = 5t_\oplus$) следующим образом:

1 такт: $Y_1 = X \oplus V$, $Y_2 = X^{\to 1}A$, $Y_3 = V^{\to 1}A^{\to 1}$, $Y_4 = V^{\to 2}X^{\to 3}$, $Y_5 = X^{\to 2}A^{\to 2}$, $Y_6 = A^{\to 3}X^{\to 4}$, $Y_7 = X^{\to 1}V^{\to 2}$, $Y_8 = X^{\to 1}V^{\to 1}$;

2 такт: $Y_1' = Y_1 \oplus Y_2$, $Y_2' = Y_3 \oplus Y_4$, $Y_3' = Y_5 \oplus Y_6$, $Y_4' = Y_7A^{\to 3}$, $Y_5' = Y_8X^{\to 2}$;

3 такт: $Y_1'' = Y_1' \oplus Y_2'$, $Y_2'' = Y_3' \oplus Y_4'$, $Y_3'' = Y_5'A^{\to 3}$;

4 такт: $Y_1''' = Y_1'' \oplus Y_2''$;

5 такт: $Y = Y_1''' \oplus Y_3''$.

Программная реализация данного УПО выполняется на основе табличного задания одной булевой функции от 12 переменных ($n' = 12$, $\mathbf{M}_{\text{УСП}} = 4096$ битов).

Обозначим через $\mathbf{C}_{\varpi}(F)$ *комбинационную сложность* функции $F(X)$, т. е. минимальное число элементов из *базиса* $\varpi = \{\vee, \&, \oplus, \bar{\ }\}$, достаточное для реализации функции $F(X)$ логической схемой, и через $\mathbf{D}_{\varpi}(F)$ — *глубину булевой функции*, т. е. число логических элементов базиса ϖ, расположенных на самом длинном пути, идущем от входа к выходу (минимальная задержка на выполнение логической схемы функции $F(X)$).

В общем случае длительность аппаратной реализации УСП ($\mathbf{T}_{\text{УСП}}$) с одной функцией-прототипом непосредственно зависит:

- ❒ от общего количества позиций для переменных в *алгебраической нормальной форме* (АНФ) функции-прототипа ($\mathbf{L}_{\text{АНФ}}$);
- ❒ от числа переменных в каждом члене АНФ.

Если $\mathbf{L}_{\text{АНФ}} = 2^k$ и в каждом члене АНФ число переменных равно степени числа 2, то величина $\mathbf{T}_{\text{УСП}} = \mathbf{D}_{\varpi}(\text{УСП}) \approx kt_{\oplus}$ и $\mathbf{C}_{\varpi}(\text{УСП}) \le 2^k - 1$. При других значениях $\mathbf{L}_{\text{АНФ}}$ величина $\mathbf{T}_{\text{УСП}}$ определяется следующей оценкой: $\mathbf{T}_{\text{УСП}} = \mathbf{D}_{\varpi}(\text{УСП}) \approx$ $\approx ([\log_2(\mathbf{L}_{\text{АНФ}})] + 1)t_{\oplus}$, при этом $\mathbf{C}_{\varpi}(\text{УСП}) \le \mathbf{L}_{\text{АНФ}}$.

С увеличением количества функций-прототипов управляемая подстановочная операция моделей $\mathbf{S}'_{\sigma}$ или $\overline{\mathbf{S}}_{\sigma}$ по сложности приближается к серии **S**-блоков большого размера и, вместе с тем, обладает рядом конструктивных и криптографических преимуществ по сравнению с ними *(подробную информацию см. в разд. 4.3)*.

Для УПО параметрически обратимой рекурсивной модели $\mathbf{\Omega}'_{\sigma,e}$ наиболее перспективной следует считать их реализацию на базе создания специальных команд *(инструкций)* процессора ЭВМ. Вместе с тем, достаточно эффективной представляется реализация таких УПО и аппаратно на базе *больших интегральных схем* (БИС). При оценке математической сложности УПО модели $\mathbf{\Omega}'_{\sigma,e}$ (ЭУС и КУС) можно исходить из сложности самого элементарного представителя этого семейства операций — сумматора по модулю 2^n. Для векторной булевой функции $Y = \Sigma(X, A)$ $(\text{GF}(2)^{32} \times \text{GF}(2)^{32} \to \text{GF}(2)^{32})$, реализующей сложение по модулю 2^n при $n = 2^m$ в базисе $\varpi = \{\vee, \&, \oplus, \bar{\ }\}$, справедливы следующие оценки [34]:

(1) $m + 1 \le \mathbf{D}_{\varpi}(\Sigma) \le m + \sqrt{2m} + 2$ и

(2) $m + 1 \le \mathbf{D}_{\varpi}(\Sigma) \le m + 7\sqrt{2m} + 14$ при $\mathbf{C}_{\varpi}(\Sigma) \le 9n$.

Из результатов *разд. 4.3.5* следует, что элементарные управляемые сумматоры $Y = \Sigma_V(X, A)$ отличаются от сумматора $Y = \Sigma(X, A)$ лишь заменой битов переноса u_i на значения $(u_i * v_i)$, где "*" — некоторая булева операция, и v_i — управляющий бит. Отсюда для реализации частных вариантов управляемых сумматоров доста-

точно в схемах, выполняющих функцию $Y = \Sigma(X, A)$, добавить один такт задержки ($\approx t_{\oplus}$) на выполнение операции (u_i*v_i) в тех местах параллельно выполняемых функций, где находятся биты переноса u_i. Таким образом, для этих управляемых сумматоров имеют место следующие оценки:

(1) $m + 1 \le \mathbf{D}_{\varpi}(\Sigma_V) \le m + \sqrt{2m} + 3$ и

(2) $m + 1 \le \mathbf{D}_{\varpi}(\Sigma_V) \le m + 7\sqrt{2m} + 15$ при $\mathbf{C}_{\varpi}(\Sigma_V) \le 10n$.

Для получения эффективных оценок для всех ЭУС и других реализаций УПО рекурсивной модели $\mathbf{\Omega}'_{\sigma,e}$ требуются дополнительные исследования. Предварительный анализ показал, что сложность $\mathbf{C}_{\varpi}$(КУС) комбинированных управляемых сумматоров этой модели линейно зависит от n, а время ее выполнения $\mathbf{T}_{\text{КУС}}$ значительно ниже времени выполнения операции умножения по модулю 2^n.

Следует отметить, что для простоты оценки параметров сложности и времени реализации УПО в качестве варианта представления преобразований рассматривалась алгебраическая нормальная форма булевых функций. Очевидно, что при решении задачи оптимизации представления образующих булевых функций, параметры реализации УПО могут быть несколько улучшены.

4.5.3. Итеративные блочные шифры на основе управляемых подстановочных операций

Итеративные блочные шифры можно представить в виде последовательности одинаковых процедур или, иначе, раундовых преобразований $\boldsymbol{E}$, расположенных в соответствии с рис. 4.12.

Рис. 4.12. Итеративная схема шифрования блочного шифра

При синтезе блочных шифров на основе управляемых операций основным механизмом управления преобразованиями является формирование управляющего

вектора V по секретному ключу K и/или преобразуемым данным D ($V = V(K, D)$). Такой механизм обеспечивает вероятностный характер управляемых операций, поскольку модификации операций преобразования на заданных шагах алгоритма не являются фиксированными, а зависят от случайного (переменного) параметра V. Данный вариант блочного шифра соответствует, например, уже рассматриваемому ранее в *разд. 4.3* и *4.4* алгоритму, построенному по усовершенствованной схеме Фейстеля (см. рис. 4.3). В отличие от традиционной схемы Фейстеля (см. рис. 4.2) в этом варианте блочного шифра вместо операции "⊕" использована параметрически обратимая управляемая подстановочная операция, инвертирование которой осуществляется параметром обратимости e, задающим прямую ($e = 0$) или обратную ($e = 1$) последовательность использования раундовых ключей $\{K_i\}$.

Такая замена, как было показано в *разд. 4.4*, позволяет существенно увеличивать стойкость шифров к дифференциальному криптоанализу. Действительно, последний основан на наличии характерных разностей, соответствующих функции F по отношению к фиксированной операции ("метрике"), в классической схеме — по отношению к операции "⊕". Как было показано в *разд. 4.4.3*, при замене этой операции на управляемую вектором V подстановочную операцию, выбор "метрики" для определения разностей является неопределенным, так как на каждом раунде алгоритма случайным образом задается разная операция, зависящая от псевдослучайного параметра V. В соответствии с результатами *разд. 4.4.5* повышение стойкости к ЛКА схемы Фейстеля на базе УПО связано с тем обстоятельством, что линейная операция "⊕" заменяется на максимально нелинейные преобразования (в рамках предлагаемых моделей УПО), зависящие от случайного параметра V и обладающие хорошими криптографическими свойствами.

При синтезе самой раундовой функции F также могут использоваться базовые примитивы, основанные на управляемых подстановочных и перестановочных операциях. Такие криптографические примитивы задают нелинейные преобразования над операндами сравнительно большого размера ($n = 32, 64, 128$) и могут быть легко реализованы в виде современных быстродействующих электронных схем. В некоторых случаях возможна и их высокоскоростная программная реализация. В терминологии *разд. 4.2* совместное использование указанных управляемых операций позволяет повысить нелинейность и улучшить показатели распространения изменений всего раундового преобразования.

В дальнейшем представлении методов синтеза новых блочных шифров будем обозначать УПО, управляемые циклические сдвиги (УЦС) и блоки управляемых перестановок (БУП), определенные ранее в *гл. 3* и *4*, соответственно, как $\boldsymbol{G}_V$, $\boldsymbol{S}_V$ и $\boldsymbol{P}_V$, причем УЦС можно рассматривать как частный случай БУП. При построении различных итеративных схем на базе УПО основное внимание будет уделяться только базовым раундовым функциям без учета специальных элементов, применяемых в современных блочных алгоритмах для обеспечения повышенной стойкости к различным методам криптоанализа. В частности, из рассмотрения ис-

ключаются такие вопросы, как предварительное и заключительное преобразования в первом и последнем раунде алгоритма [120], а также процедуры преобразования (расширения) ключа [120], которые затрагиваются в других разделах книги. Необходимо отметить тот факт, что положительной особенностью блочных шифров на базе УПО и других управляемых операций является возможность обеспечения достаточно эффективных механизмов динамического преобразования ключей в ходе раундовых преобразований за счет самих управляемых операций. Отсутствие процедур предварительных вычислений такого типа является существенным преимуществом, выгодно отличающим такие шифры от других блочных алгоритмов. Однако необходимо иметь в виду, что в некоторых случаях при больших размерах блоков данных блочного шифра ($n \geq 128$) и наличии большого количества управляемых элементов в пределах одного раунда преобразований в предлагаемых схемах может возникать необходимость увеличения размеров ключевой информации, получаемой из секретного ключа ограниченной длины (128—256 битов). Рассмотрение указанного вопроса (процедуры расширения секретного ключа) выходит за рамки данного раздела книги. Таким образом, все ключевые элементы K_i, используемые в рассматриваемых далее итеративных схемах, будут считаться определенными и имеющими достаточные размеры для проведения соответствующих раундовых преобразований.

4.5.4. Варианты синтеза блочных шифров на основе управляемых операций

Обозначим через $[\boldsymbol{GP}]_W$ *произвольный подстановочно-перестановочный блок*, состоящий из некоторой совокупности взаимосвязанных УПО, БУП и УЦС, управляемых различными векторами V_i, сформированными из компонентов некоторого *расширенного управляющего вектора* W, являющегося функцией ключей и преобразуемых данных. Параметрически обратимые подстановочно-перестановочные блоки будут обозначаться, соответственно, $[\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_W$ — в случае прямого и $[\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_W$ — в случае обратного преобразования. Различные блоки обозначаются с использованием дополнительных индексов, например, $[\boldsymbol{G}^{(e)}\boldsymbol{P}^{(e)}]_W^{(i)}$.

В качестве примера построения блочных шифров в рамках схемы Фейстеля можно рассмотреть представленную на рис. 4.13 обобщенную схему раундового преобразования $\boldsymbol{E}$ на базе УПО, УЦС и БУП [12].

В указанной схеме параметр e определяет прямое ($e = 0$) и обратное ($e = 1$) преобразование в блоках и во всем раунде алгоритма, а также задает прямой и обратный порядок использования ключей. В итоге параметр e определяет прямой и обратный процесс всего преобразования в блочном алгоритме.

Если в одном раунде применяется q ключей (в обобщенной схеме на рис. 4.13 $q = 4$), то, вообще говоря, для реализации всего шифра требуется qr различных ключей. Так как размеры исходного секретного ключа ограничены 128—256 би-

тами, то, как правило, возникает необходимость в составлении *расписания* использования ограниченного набора секретных ключей или в проведении *специальной процедуры* для формирования рабочих ключей для каждого раунда шифрования (процедура расширения ключа). Принципы организации такого расписания или процедуры составляют отдельную тему и не рассматриваются в последующих разделах этой главы. В дальнейшем будем считать, что для предлагаемых схем имеется адекватная система организации ключевых данных.

Рис. 4.13. Общая схема использования подстановочно-перестановочных преобразований в одном раунде $\boldsymbol{E}$ усовершенствованной схемы Фейстеля

В соответствии со свойствами схемы Фейстеля раундовая функция F может состоять из совокупности подстановочно-перестановочных операций, осуществляющих преобразование данных, которое не нуждается в обратном преобразовании при формировании обратного алгоритма. В этой связи в схеме рис. 4.13 раундовую функцию можно определить в виде блока $F = [\boldsymbol{GP}]_W$. В соответствии с рекомендациями из *разд. 4.3.7*, для данного блока наиболее перспективными являются высокоскоростные УПО в рамках биективной последовательной модели $\mathbf{S}'_\sigma$ и небиективной последовательной модели $\overline{\mathbf{S}}_\sigma$, реализуемые достаточно эффективно как аппаратно, так и программно. В качестве перестановочных элементов в блоке $[\boldsymbol{GP}]$ при аппаратной реализации целесообразно применение различных БУП, тогда как для программной реализации допускается использование УЦС и/или элементарных БУП.

В качестве блока $\boldsymbol{B}_3^{(e)}$ можно рассматривать два варианта преобразований:

- Операцию $\boldsymbol{B}_3^{(e)} = "\oplus"$, характерную для *традиционной схемы Фейстеля*, или
- УПО $\boldsymbol{B}_3^{(e)} = \boldsymbol{G}_{W_3,e}(X_R, A)$, соответствующую *усовершенствованной схеме Фейстеля*.

Для каждого из этих двух вариантов возможны следующие четыре схемы прямых и обратных преобразований $\boldsymbol{B}_1^{(e)}$ и $\boldsymbol{B}_2^{(e)}$:

1. $e = 0:\ \boldsymbol{B}_1^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_1}$ и $\boldsymbol{B}_2^{(0)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_2}$;

   $e = 1:\ \boldsymbol{B}_1^{(1)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_2}$ и $\boldsymbol{B}_2^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_1}$.

2. $e = 0:\ \boldsymbol{B}_1^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_2}$ и $\boldsymbol{B}_2^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_1}$;

   $e = 1:\ \boldsymbol{B}_1^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_2}$ и $\boldsymbol{B}_2^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_1}$.

3. $e = 0:\ \boldsymbol{B}_1^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_1}^{(1)}$ и $\boldsymbol{B}_2^{(0)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_2}^{(2)}$;

   $e = 1:\ \boldsymbol{B}_1^{(1)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_2}^{(2)}$ и $\boldsymbol{B}_2^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_1}^{(1)}$.

4. $e = 0:\ \boldsymbol{B}_1^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_1}^{(1)}$ и $\boldsymbol{B}_2^{(0)} = [\boldsymbol{G}^{(0)}\boldsymbol{P}^{(0)}]_{W_2}^{(2)}$;

   $e = 1:\ \boldsymbol{B}_1^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_2}^{(2)}$ и $\boldsymbol{B}_2^{(1)} = [\boldsymbol{P}^{(1)}\boldsymbol{G}^{(1)}]_{W_1}^{(1)}$.

Особенностью первого варианта является тот факт, что процесс прямого и обратного преобразования осуществляется за счет изменения порядка использования ключей без логического изменения самой схемы блочного шифра. Необходимо заметить, что в первом и третьем вариантах блоки $\boldsymbol{B}_1$ и $\boldsymbol{B}_2$ имеют идентичный состав из прямых и, соответственно, обратных УПО, УЦС и БУП, но отличаются противоположным порядком их расположения. При этом существенная особенность блоков $\boldsymbol{B}_1$ и $\boldsymbol{B}_2$ заключается в использовании различных расширенных управляющих векторов W_1 и W_2.

Во втором варианте аналогичная реализация шифра одной схемой при обратном порядке ключей возможна лишь в случае, когда $\boldsymbol{B}_1^{(e)} = [\boldsymbol{P}^{(e)}]_{W_1}$ или $\boldsymbol{B}_1^{(e)} = [\boldsymbol{G}^{(e)}]_{W_1}$ и, соответственно, $\boldsymbol{B}_2^{(e)} = [\boldsymbol{P}^{(e)}]_{W_2}$ или $\boldsymbol{B}_2^{(e)} = [\boldsymbol{G}^{(e)}]_{W_2}$. Для третьего и четвертого вариантов всегда необходимы отдельные схемы прямого и обратного преобразования.

В соответствии с рекомендациями, предложенными в *разд. 4.3.7*, при аппаратной реализации в качестве УПО в блоках $[\boldsymbol{G}^{(e)}\boldsymbol{P}^{(e)}]_{W_1}^{(i)}$, $[\boldsymbol{P}^{(e\oplus 1)}\boldsymbol{G}^{(e\oplus 1)}]_{W_2}^{(i)}$ и $\boldsymbol{G}_{W_3,e}$ целе-

сообразно использовать реализации параметрически схемно-обратимой последовательной модели $\mathbf{S}'_{\sigma,e_S}$ или разновидности параметрически обратимой рекурсивной модели $\mathbf{\Omega}''_{\sigma,e}$. В качестве перестановочных элементов в указанных выше блоках (так же, как и в блоке $[\boldsymbol{GP}]_W$) при аппаратной реализации целесообразно применять различные БУП, а при программной — УЦС и/или элементарные БУП. Ввиду особенностей схемы Фейстеля раундовая функция F (блок $[\boldsymbol{GP}]_W$) не требует обращения при обратном преобразовании, что обуславливает возможность значительного ее усложнения в отличие от других элементов раундового преобразования $\boldsymbol{E}$. Возможен, но менее перспективен и альтернативный подход — усложнение преобразования $\boldsymbol{E}$ за счет блоков $\boldsymbol{B}_1^{(e)}$ и $\boldsymbol{B}_2^{(e)}$. В связи с этим на базе УПО и БУП можно формировать различные по сложности комбинированные схемы раундовых функций, обеспечивающие высокую эффективность преобразований. Вследствие указанного свойства схемы Фейстеля при синтезе блочных шифров возможны два основных направления формирования раундовых преобразований. Первое из них связано с созданием упрощенного раундового преобразования $\boldsymbol{E}$ и проведением повышенного количества итераций при выполнении алгоритма, второе — с формированием достаточно сложного раундового преобразования и значительно меньшим количеством итераций. В качестве примеров первого подхода на рис. 4.14 и 4.15 приводятся некоторые простейшие раундовые функции F (блоки $[\boldsymbol{GP}]_W$) и соответствующие им другие элементы раундового преобразования $\boldsymbol{E}$.

Рис. 4.14. Примеры простейших блоков $[\boldsymbol{GP}]_W$ ($W = V_1 \cup V_2$):
а — для аппаратной реализации; *б* — для программной реализации

Блоки $[\boldsymbol{P}^{(e\oplus 1)}\boldsymbol{G}^{(e\oplus 1)}]_{W_2}$, соответствующие блокам $[\boldsymbol{G}^{(e)}\boldsymbol{P}^{(e)}]_{W_1}$, имеют вид, представленный на рис. 4.16.

В соответствии с результатами из *разд. 4.3* преобразования на базе указанных выше блоков при отсутствии коллизий в УПО по входным данным осуществляют высокоскоростные биективные по $\boldsymbol{X}_R$ отображения. Степень стойкости блоч-

ных шифров на основе указанных выше простейших элементов раундового преобразования $\boldsymbol{E}$ может варьироваться в зависимости от количества раундов всего алгоритма.

Рис. 4.15. Примеры простейших блоков $[\boldsymbol{G}^{(e)}\boldsymbol{P}^{(e)}]_{W_1}$ $(W_1 = V_1' \cup V_2')$:
а — для аппаратной реализации; *б* — для программной реализации

Рис. 4.16. Примеры блоков $[\boldsymbol{P}^{(e\oplus 1)}\boldsymbol{G}^{(e\oplus 1)}]_{W_2}$ $(W_2 = V_1'' \cup V_2'')$:
а — для аппаратной реализации; *б* — для программной реализации

4.5.5. Синтез комбинированных раундовых функций на основе управляемых подстановочных операций

Альтернативный подход к упрощенному синтезу раундовых функций блочных шифров связан с созданием комбинированных раундовых функций. На рис. 4.17 показан пример усложненной раундовой функции F, т. е. блоков $[\boldsymbol{GP}]_W$ из общей схемы рис. 4.13. При этом функция F не имеет коллизий отдельно по $\boldsymbol{X}_L$

$(\forall \boldsymbol{X}_L \neq \boldsymbol{X}_L'\ \ F(\boldsymbol{X}_L,\ K_e^{(1)}) \neq\ F(\boldsymbol{X}_L',\ K_e^{(1)}))$ и по $\boldsymbol{K}_e^{(1)}$ $(\forall K_e^{(1)} \neq K_e'^{(1)}\ \ F(\boldsymbol{X}_L,\ K_e^{(1)}) \neq$ $\neq F(\boldsymbol{X}_L,\ K_e'^{(1)}))$. Данные раундовые функции могут быть использованы при построении 128-битовых блочных шифров, относящихся к алгоритмам с высокими показателями стойкости к различным криптоатакам.

Рис. 4.17. Пример усложненной раундовой функции *F* на основе УПО и БУП

Стойкость к методам ДКА и ЛКА в таких преобразованиях, как было показано в *разд. 4.4*, обеспечивается за счет параллельно-последовательного выполнения УПО и БУП, имеющих хорошие совместные показатели нелинейности и распространения изменений, а также высокую степень распространения изменений битов в пределах всего преобразуемого блока $\boldsymbol{X}_L$. Положительная особенность использования усложненных раундовых функций состоит также в том, что в ходе параллельно-последовательного выполнения УПО и БУП происходит внутреннее преобразование ключевых данных, которые на каждом последующем шаге преобразований принимают участие либо в управлении, либо сами преобразуются в зависимости от данных из предыдущего шага. Такой механизм в ряде случаев может заменять *специальные процедуры преобразования (расширения) ключей*, используемые в блочных шифрах других типов и выполняемые каждый раз при инициализации алгоритма. Возможность не применять специальные процедуры указанного вида является еще одним преимуществом блочных шифров на базе управляемых операций и делает такие алгоритмы предпочтительными для использования в системах, где необходима частая смена ключей.

4.5.6. Практическая реализация управляемых подстановочных операций в блочных шифрах

На базе БУП и УПО проведен научный эксперимент по программной реализации макетов различных блочных алгоритмов, в частности блочного шифра SPECTR-H64, раундовое преобразование которого соответствует параметрически

обратимому первому варианту усовершенствованной схемы Фейстеля (см. рис. 4.13). Подробно спецификация этого алгоритма излагается в *гл. 5*. Как указывалось ранее, для прямого и обратного преобразования данных в блочном шифре типа SPECTR-H64 используется одна и та же схема алгоритма.

Раундовое преобразование $\boldsymbol{E}^{(e)}$ блочного алгоритма SPECTR-H64, который имеет несколько модификаций, в качестве одного из базовых криптографических примитивов содержит управляемую подстановочную операцию $\boldsymbol{G}_V$ последовательной биективной модели $\mathbf{S}'_\sigma$ на основе однотипных образующих функций с одной функцией-прототипом *(подробную информацию см. в разд. 4.3.7)*.

В качестве функции-прототипа в двух различных модификациях блочного шифра использовалось два варианта булевых функций:

1. $f = f(x_1, \ldots, x_8) = x_7 \oplus x_8 \oplus \varphi(x_1, \ldots, x_6)$,

   где $\varphi(x_1, \ldots, x_6) = x_1x_2x_3 \oplus x_1x_4 \oplus x_2x_5 \oplus x_3x_6$ — бент-функция *(см. разд. 4.2.8)*.

2. $f = f(x_1, \ldots, x_{12}) = x_{11} \oplus x_{12} \oplus \varphi(x_1, \ldots, x_{10})$,

   где $\varphi(x_1, \ldots, x_{10}) = x_1x_2 \oplus x_3x_4 \oplus x_5x_6 \oplus x_7x_8 \oplus x_9x_{10} \oplus x_1x_5x_9 \oplus x_1x_3x_7x_9$ — бент-функция.

Для определенности рассмотрим второй вариант прототипа, соответствующий примеру 4.3 из *разд. 4.2.8*, и оценим его криптографические свойства. С этой целью используем спектр преобразования Уолша-Адамара *(см. разд. 4.2.3)* функции f, который состоит из 1024 значений $|U_\alpha| = 128$ для векторов $\alpha = (x_1, \ldots, x_{10}, 0, 0)$ и 3072 значений $U_\alpha = 0$ для векторов $\alpha = (x_1, x_2, \ldots, x_{12})$, где $x_{12} \neq 0$ и/или $x_{11} \neq 0$.

Принимая во внимание указанные факты и учитывая другие специальные свойства бент-функций *(см. разд. 4.2.8)*, имеем следующие характеристики функции f:

- нелинейность функции f при $m = 12$ определяется значением $\mathbf{N}(f) = 2^{m-1} - 2^{(1/2)(m+2)-1} = 2^{11} - 2^6 = 1984$, что достаточно близко к максимально возможной нелинейности для булевых функций от 12 переменных $\mathbf{N}_{\max} = 2^{m-1} - 2^{(m/2)-1} = 2^{11} - 2^5 = 2016$ (см. утверждение 4.17);
- f — корреляционно-иммунная функция относительно всех векторов $\alpha = (x_1, x_2, \ldots, x_{12})$, где $x_{12} \neq 0$ и/или $x_{11} \neq 0$, что составляет 75% от всех 4096 векторов *(см. разд. 4.2.7)*.
- булева функция f имеет хорошие автокорреляционные свойства (лавинный эффект и эффект распространения ошибок *(см. разд. 4.2.5 и 4.2.6)*, т. к. для всех ненулевых векторов $(\Delta x_1, \Delta x_2, \ldots, \Delta x_{12})$, кроме трех $(0, 1, 0, \ldots, 0)$, $(1, 0, 0, \ldots, 0)$ и $(1, 1, 0, \ldots, 0)$, выполняется критерий распространения $\mathbf{PC}(12)$;
- булева функция f имеет достаточно высокую степень алгебраической нормальной формы $\mathbf{deg}(f) = 4$ *(см. разд. 4.2.2)*.

В соответствии со схемой шифра имеем $K = K_1 \| K_2$ и $K_1 = (k_1^{(1)}, k_2^{(1)}, ..., k_{32}^{(1)})$, $K_2 = (k_1^{(2)}, k_2^{(2)}, ..., k_{32}^{(2)})$.

Перейдем от функции-прототипа f к конкретным образующим для УПО функциям по следующей подстановке переменных:

1. Для первого варианта функции-прототипа:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 \\ x_{i-2} & x_{i-5} & k_i^{(2)} & k_i^{(2)} & x_{i-3} & x_{i-4} & x_{i-1} & x_i \end{pmatrix}$$

2. Для второго варианта функции-прототипа:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} \\ x_{i\text{-}1} & x_{i\text{-}3} & k_{i-1}^{(2)} & x_{i-5} & k_{i-2}^{(2)} & x_{i-7} & x_{i-6} & x_{i-8} & x_{i-9} & x_{i-11} & x_i & k_i^{(2)} \end{pmatrix}.$$

Во втором случае $\forall i \in \{1, \ldots, n\}$ имеем множество образующих функций следующего вида:

$$y_i = f_i = x_i \oplus k_i^{(2)} \oplus x_{i-1}x_{i-3} \oplus k_{i-1}^{(2)} x_{i-5} \oplus k_{i-2}^{(2)} x_{i-7} \oplus x_{i-6}x_{i-8} \oplus x_{i-9}x_{i-11} \oplus$$
$$\oplus x_{i-1} k_{i-2}^{(2)} x_{i-9} \oplus x_{i-1} k_{i-2}^{(2)} x_{i-6}x_{i-9} =$$
$$= x_i \oplus k_{i-1}^{(2)} x_{i-5} \oplus k_{i-2}^{(2)} x_{i-7} \oplus \phi_{bent}(x_{i-1}, x_{i-3}, x_{i-6}, x_{i-8}, x_{i-9}, x_{i-11}, k_{i-1}^{(2)}, k_{i-2}^{(2)}) \oplus k_i^{(2)},$$

где x_j — компоненты вектора преобразуемых данных $X_L \in \mathrm{GF}(2)^{n/2}$, $k_i^{(2)}$ — компоненты ключа $K_2 \in \mathrm{GF}(2)^{n/2}$ и $x_{-10}, \ldots, x_0$, $k_{-1}^{(2)}$, $k_0^{(2)}$ — начальные условия.

УПО, сформированное на базе представленных выше образующих функций и осуществляющее отображение $X \to Y$: $\mathrm{GF}(2)^{n/2} \times \mathrm{GF}(2)^{n/2} \to \mathrm{GF}(2)^{n/2}$, дополнительно обладает регулярностью по X, K_2, т. е. Y ровно $2^{n/2}$ раза принимает все $2^{n/2}$ различных значений из $\mathrm{GF}(2)^{n/2}$, в то время как обобщенный вектор $X \| K_2$ проходит 2^n значений из $\mathrm{GF}(2)^n$. Как следствие этого, УПО является биективной по X при различных фиксированных значениях ключа K_2 и не имеет коллизий по ключу K_2 при различных фиксированных X.

В соответствии со схемой блочного шифра SPECTR-H64 *(см. гл. 5)* на входе УПО осуществляется линейное преобразование переменных $x_i = x_i \oplus k_i^{(1)}$, не изменяющее исходные значения нелинейности и степени алгебраической нормальной формы образующих функций (см. утверждение 4.16). Дополнительным положительным свойством результирующего УПО является регулярность по X, K_1, $K_2 \in \mathrm{GF}(2)^{n/2}$ (см. теорему 4.3). Как следствие этого свойства, УПО не имеет коллизий и по второму ключу $K_1 \in \mathrm{GF}(2)^n$ при фиксированных значениях X (см. следствия из теоремы 4.2). В результате преобразований образующие функции зависят уже от двух ключей и приобретают следующий вид:

$$f_i = x_i \oplus k_i^{(1)} \oplus k_i^{(2)} \oplus (x_{i-1} \oplus k_{i-1}^{(1)})(x_{i-3} \oplus k_{i-3}^{(1)}) \oplus k_{i-1}^{(2)} (x_{i-5} \oplus k_{i-5}^{(1)}) \oplus k_{i-2}^{(2)} (x_{i-7} \oplus k_{i-7}^{(1)}) \oplus$$

$$\oplus (x_{i-6} \oplus k_{i-1}^{(1)})(x_{i-8} \oplus k_{i-8}^{(1)}) \oplus (x_{i-9} \oplus k_{i-9}^{(1)})(x_{i-11} \oplus k_{i-11}^{(1)}) \oplus (x_{i-1} \oplus k_{i-1}^{(1)}) k_{i-2}^{(2)} (x_{i-9} \oplus k_{i-9}^{(1)}) \oplus$$

$$\oplus (x_{i-1} \oplus k_{i-1}^{(1)}) k_{i-1}^{(2)} (x_{i-6} \oplus k_{i-6}^{(1)})(x_{i-9} \oplus k_{i-9}^{(1)}) = x_i \oplus k_{i-1}^{(2)} x_{i-5} \oplus k_{i-2}^{(2)} x_{i-7} \oplus$$

$$\oplus \phi_{bent}(x_{i-1}, x_{i-3}, x_{i-6}, x_{i-8}, x_{i-9}, x_{i-11}, k_i^{(1)}, k_{i-1}^{(1)}, k_{i-3}^{(1)}, k_{i-6}^{(1)}, k_{i-8}^{(1)}, k_{i-9}^{(1)}, k_{i-11}^{(1)}, k_i^{(2)}, k_{i-1}^{(2)}, k_{i-2}^{(2)}) \oplus$$

$$\oplus \gamma(k_i^{(1)}, k_{i-1}^{(1)}, k_{i-3}^{(1)}, k_{i-6}^{(1)}, k_{i-8}^{(1)}, k_{i-9}^{(1)}, k_{i-11}^{(1)}, k_i^{(2)}, k_{i-1}^{(2)}, k_{i-2}^{(2)}).$$

Таким образом, эквивалентное преобразование, полученное в результате линейного преобразования входного блока данных X_L, соответствует управляемой подстановочной операции $Y = \boldsymbol{G}_{K_2}(X_L, K_1)$. В векторном представлении *(см. разд. 4.5.2)* данная УПО имеет следующий вид:

$$L = X_L \oplus K_1, \;\; Y = \boldsymbol{G}_{K_2}(X_L, K_1) = L \oplus K_2 \oplus L^{\to 1} L^{\to 3} \oplus L^{\to 5} K_2^{\to 1} \oplus L^{\to 7} K_2^{\to 2} \oplus$$

$$\oplus L^{\to 6} L^{\to 8} \oplus L^{\to 9} L^{\to 11} \oplus L^{\to 1} K_2^{\to 2} L^{\to 9} \oplus L^{\to 1} K_2^{\to 1} L^{\to 6} L^{\to 9}.$$

Несмотря на внешнюю сложность, УПО аппаратно реализуется всего за время $T_{\text{УПО}} = 6t_{\oplus}$ *(см. разд. 4.5.2)*.

При фиксированных значениях ключей K_1, K_2 исходные образующие функции в общем виде трансформируются к трем возможным типам функций:

$$f = f_{aff}(z_1, z_2, z_3) \oplus f_{bent}(z_4, z_5, z_6, z_7, z_8, z_9),$$

$$f' = f'_{aff}(z_1, z_2) \oplus f'_{bent}(z_3, z_4, z_5, z_6, z_7, z_8),$$

$$f'' = f''_{aff}(z_1) \oplus f''_{bent}(z_2, z_3, z_4, z_5, z_6, z_7).$$

Отсюда следует, что и при фиксированных ключах образующие функции сохраняют свою структуру, имеют достаточно высокую нелинейность — соответственно 224, 112, 56 (см. утверждение 4.17) и другие упомянутые выше свойства только при меньшем количестве переменных. Это обстоятельство является еще одним положительным свойством УПО такого типа, поскольку обеспечивает стойкость к криптоанализу при условии, что часть информации о ключах известна.

Приведенный пример практического построения и анализа УПО является иллюстрацией возможностей по проектированию новых криптографических примитивов блочного шифрования, обладающих заданными криптографическими свойствами и регулируемой сложностью реализации. Использование для создаваемых УПО различных функций-прототипов, обладающих широким спектром специальных криптографических свойств, обеспечивает гибкий механизм по проектированию целого ряда новых блочных шифров с контролируемыми параметрами стойкости к современному криптоанализу.

4.6. Заключение

Проведенное в данной главе описание управляемых подстановочных операций как нелинейных криптографических примитивов блочных шифров составляет одно из звеньев всесторонне развивающейся в настоящее время *концепции управляемых операций*, являющейся одним из перспективных направлений в области защиты информации. Главное содержание этой концепции состоит в создании подстановочных и перестановочных элементов блочных шифров, которые обеспечивают высокоскоростные аппаратно и, в ряде случаев, программно реализуемые нелинейные преобразования с большим числом модификаций. Такие преобразования осуществляются сразу над всем блоком данных большого размера (32 и более битов) и динамически управляются преобразуемыми данными и ключами алгоритма. УПО являются альтернативой традиционным **S**-блокам и обычным математическим операциям, которые, как правило, используются при синтезе блочных шифров. В определенном смысле УПО занимают промежуточное положение между **S**-блоками большого размера и парой алгебраических операций — операцией XOR и операцией сложения по модулю 2^n. УПО сочетают в себе положительные качества этих базовых примитивов, но свободны от ряда присущих им недостатков. Предусмотренные для УПО механизмы и способы применения в блочных шифрах обеспечивают высокую стойкость таких алгоритмов к атакам на основе современных методов криптоанализа.

ГЛАВА 5

Проектирование скоростных шифров на основе управляемых операций

В настоящей главе рассматриваются только алгоритмы блочных шифров, разработанные авторами книги. Терминология и обозначения, используемые в данной главе, в целом соответствуют предыдущим двум главам, если не оговорено иное.

5.1. Базовые криптосхемы и ключевая система

Рассмотрим пятерку $\{\mathbf{M}, \mathbf{C}, \mathbf{K}, \mathbf{E}, \mathbf{D}\}$, где $\mathbf{M}$ и $\mathbf{C}$ — множества, соответственно, преобразуемых и преобразованных блоков данных фиксированных длин, $\mathbf{K}$ — множество ключей, а $\mathbf{E}$ и $\mathbf{D}$ — множества прямых и обратных отображений, то есть

$$\mathbf{E}: \mathbf{M} \times \mathbf{K} \to \mathbf{C},$$

$$\mathbf{D}: \mathbf{C} \times \mathbf{K} \to \mathbf{P},$$

причем $\forall K \in \mathbf{K}\ \exists!\ \mathbf{E}_K \in \mathbf{E}$ и $\exists\ \mathbf{D}_K \in \mathbf{D}$ такие, что $\forall X \in \mathbf{M}\ \mathbf{D}_K(\mathbf{E}_K(X)) = X$.

Такая совокупность $\{\mathbf{M}, \mathbf{C}, \mathbf{K}, \mathbf{E}, \mathbf{D}\}$ называется *блочным алгоритмом преобразования данных* или просто *блочным шифром*.

В общем случае мощность множества $\mathbf{C}$ не может быть меньше мощности множества $\mathbf{M}$ [129]. В блочных шифрах множества $\mathbf{M}$ и $\mathbf{C}$, как правило, совпадают с множеством бинарных векторов длины n ($\mathbf{M} = \mathbf{C} = \mathrm{GF}(2)^n$), а множество $\mathbf{K} \subseteq \mathrm{GF}(2)^m$. В этом случае $\mathbf{E}_K$ и $\mathbf{D}_K$ — реализуют взаимно однозначное (биективное) отображение, причем вместо $\mathbf{D}_K$ используют обозначение $\mathbf{E}_K^{-1}$, а блочный шифр обозначают четверкой $\{\mathbf{M}, \mathbf{C}, \mathbf{K}, \mathbf{E}\}$.

Суть блочного преобразования заключается в возможности выбора произвольного отображения $\mathbf{E}_K$ ($\mathbf{E}_K \in \mathbf{E}$), позволяющего каждому, кому известен ключ K, вы-

полнять прямые и обратные преобразования, соответственно, над исходными и преобразованными блоками данных, однако без знания ключа K практически невозможно установить, какое отображение использовалось, если даже имеется несколько значений исходных и преобразованных величин. При этом множество $\mathbf{E}$ должно быть достаточно большим, чтобы нельзя было выполнить процедуру полного (тотального) перебора всех возможных отображений $\mathbf{E}_K$. Поскольку при длине m битовой последовательности ключа можно задать не более 2^m различных отображений $\mathbf{E}_K$, в настоящее время рекомендуется использовать ключи длиной более 80 битов. В частности, организаторами конкурса AES при его проведении было предложено предусмотреть возможность использования трех стандартных длин: 128, 192 или 256 битов, что целиком исключает возможность полного перебора ключей, и при этом позволяет удовлетворить потребности бизнеса в использовании алгоритмов с учетом приоритета требований "скорость, стоимость, стойкость".

Идеальным считается вариант, когда каждому ключу K соответствует свое уникальное отображение $\mathbf{E}_K$. Однако ввиду невозможности размещения и хранения такого множества отображений в средствах вычислительной техники применяют алгоритмический способ их построения.

Как правило, используют "композиционный" принцип, когда суммарное отображение $\mathbf{E}_K$ представляет собой последовательность достаточно простых, зависящих от ключа K, отображений $\mathbf{E}_K^{(i)}$ ($i = 1, 2, \ldots, r$). То есть

$$\mathbf{E}_K = \mathbf{E}_K^{(1)} \circ \mathbf{E}_K^{(2)} \circ \ldots \circ \mathbf{E}_K^{(r)}. \tag{5.1}$$

В теоретическом плане, принцип построения блочных шифров на основе сравнительно простых операций впервые был предложен К. Шенноном [37], где он в качестве простых операций использовал перестановки и подстановки. Построенные по такой схеме блочные алгоритмы, получили название подстановочно-перестановочных сетей (SP-networks, Substitution-Permutation networks) [75]. Первая из перечисленных выше операций выполняет перестановку элементов преобразуемого блока данных, а вторая — замену одних элементов блока, например символов, другими, которые обычно выбираются из одного и того же алфавита. Предполагалось, что выбор операций определяется ключом. В такой интерпретации каждая операция однозначно задавалась значением ключа и не изменялась до очередного ввода ключа.

Операции подстановки и перестановки обеспечивают два важных свойства, которыми должны обладать стойкие блочные шифры: свойство рассеивания и свойство перемешивания. В понятии "стойкий блочный шифр" изначально присутствует некоторая условность, поскольку любой блочный шифр с фиксированной длиной ключа не является абсолютно стойким, так как теоретически с использованием метода полного перебора (тотального опробования), получившего в иностранной литературе название *"грубой силы"* (brute force), ключ может быть

найден. Более того, такой ключ может существовать не обязательно в единственном экземпляре. Дальнейшим развитием теории Шеннона явилось использование преобразуемых данных для управления выбором конкретных преобразований, что и было впервые реализовано в алгоритме DES.

5.1.1. Итеративный (композиционный) блочный шифр

На практике обычно формируется некоторое базовое отображение $\mathbf{E}'$, называемое *раундовым*, или цикловым, которое выполняется заданное число раз. Следует отметить, что в каждом раунде используются, вообще говоря, разные раундовые ключи $K^{(i)}$, формируемые на основе базового ключа K. При этом суммарное преобразование $\mathbf{E}_K$ имеет следующий вид:

$$\mathbf{E}_K = \mathbf{E}'_{K^{(1)}} \circ \mathbf{E}'_{K^{(2)}} \circ \ldots \circ \mathbf{E}'_{K^{(r)}}, \tag{5.2}$$

Такой алгоритм называется *итеративным (композиционным)*, или *итерационным*.

Выбор итеративной схемы шифрования обусловлен не только более высокой технологичностью производства криптосхемы в случае аппаратной реализации, но и необходимостью обоснования как оптимальности использования каждого сформированного отображения в отдельности, так и оптимальности их взаимодействия в плане обеспечения необходимой стойкости шифра. Именно по этой причине итеративные шифры получили широкое распространение. Более того, в таких шифрах сложнее скрыть возможные "потайные ходы" (предусмотренные разработчиком слабости криптосистемы), что повышает доверие пользователей к средствам защиты информации, в которых используются такие алгоритмы.

Объединение раундовых ключей $Q^{(0)} = K^{(1)} \parallel K^{(2)} \parallel \ldots \parallel K^{(r)}$ обычно называют расширенным ключом, причем верхний индекс "0" соответствует режиму "зашифрования" (*прямое* преобразование данных).

Базовый ключ K по своей сути является секретным элементом и поэтому называется секретным ключом. В литературе также используется термин мастер-ключ. Однако, чтобы подчеркнуть его отличие от расширенного ключа, который также является секретным, будем использовать термин "основной ключ".

Для выполнения *обратного* преобразования ("расшифрование") используется отображение $(\mathbf{E}')^{-1}$ и последовательность ключей, выписанная в обратном порядке, а именно:

$$\mathbf{E}_K^{-1} = (\mathbf{E}')^{-1}_{K^{(r)}} \circ \ldots \circ (\mathbf{E}')^{-1}_{K^{(2)}} \circ (\mathbf{E}')^{-1}_{K^{(1)}} \tag{5.3}$$

Соответственно $Q^{(1)} = K^{(r)} \parallel K^{(r-1)} \parallel \ldots \parallel K^{(2)} \parallel K^{(1)}$ — расширенный ключ для обратного преобразования.

5.1.2. Процедура генерации расширенного ключа

Расширенный ключ можно представить в следующем виде:

$$Q^{(e)} = Q^{(e,\,1)} \parallel Q^{(e,\,2)} \parallel \ldots \parallel Q^{(e,\,r)},$$

где e — режим преобразования данных ($e = 0$ — прямое и $e = 1$ — обратное преобразование данных) и $\forall j = 1, 2, \ldots, r$ $Q^{(e,j)} = K^{(j + e(r - 2j + 1))}$. То есть ключ $Q^{(e)}$ является функцией от двух переменных, а именно:

$$Q^{(e)} = \mathbf{H}(K, e), \tag{5.4}$$

где K — основной ключ.

Функцию $\mathbf{H}(K, e)$ обычно называют *процедурой генерации расширенного ключа*. Если при этом $\omega' = \omega(K^{(i)})$ — размерность раундового ключа, тогда размерность расширенного ключа равна $r\omega'$. То есть процедура генерации расширенного ключа предназначена сформировать некоторую битовую последовательность заданной длины.

В общем виде расширенный ключ может использоваться не только для задания раундовых ключей, но и для инициализации (выбора) криптографических примитивов (как, например, в программных шифрах SPECTR-Z [2], Khufu [120], а также Blowfish, SEAL).

В зависимости от того, какими свойствами обладает сформированная последовательность, процедура $\mathbf{H}(K, e)$ может иметь специальное название. Например, *криптографически стойкая (сильная) процедура генерации расширенного ключа* (the strong procedure to extended key) или *расписание ключей* (key schedule). Хотя исторически сложилось так, что в обоих случаях используют один и тот же термин "key schedule", различия между ними весьма существенные.

Так, битовая последовательность расширенного ключа, сформированная с использованием криптографически стойкой процедуры [76], должна удовлетворять следующим требованиям:

- все биты каждого ключевого элемента расширенного ключа должны быть независимы между собой и иметь равновероятное распределение;
- биты каждого ключевого элемента расширенного ключа должны быть статистически независимы от битов нескольких соседних ключевых;
- общее число нулей в расширенном ключе не должно существенно отличаться от числа единиц;
- множество формируемых последовательностей не должно быть существенно меньше множества основных ключей.

В то же время расписание ключей, как правило, подразумевает, что каждый раундовый ключ является обычным подмножеством основного ключа. Классическим примером использования расписания ключей являются DES [33, 120] и

ГОСТ 28147-89 [10], а также SPECTR-H64, SPECTR-128, описание которых приведено в данной главе.

Существует специальная классификация блочных шифров по типу используемой процедуры генерации расширенного ключа, в соответствии с которой каждому алгоритму может быть присвоен двузначный индекс. При этом первый символ имеет всего два значения (1 или 2), а второй — три (A, B, C).

К первой группе относятся блочные шифры, для которых знание раундового ключа позволяет определить все или некоторые биты основного ключа или остальных раундовых ключей. К этой группе относятся все шифры, в которых используется так называемое *расписание ключей*, когда каждый раундовый ключ является подмножеством битов основного ключа.

Ко второй группе относятся блочные шифры, для которых главным критерием является зависимость битов раундовых ключей от всех или не всех битов основного ключа. К этой группе относятся большинство современных блочных шифров, участвовавших в конкурсе AES.

Описание всех шести групп приведено в табл. 5.1 [8].

Таблица 5.1. *Классификация процедур генерации расширенного ключа*

Индекс	Характеристика	Примеры
1A	Знание раундового ключа позволяет однозначно восстановить основной ключ и остальные раундовые ключи	SPECTR-128, NDC
1B	Знание раундового ключа позволяет восстановить некоторые биты основного ключа и/или остальных раундовых ключей	DES, ГОСТ, SPECTR-H64, Magenta
1C	Знание раундового ключа позволяет восстановить некоторые биты основного ключа и/или остальных раундовых ключей после выполнения некоторых сравнительно простых арифметических операций	Rijndael, Crypton, SAFER+, DEAL, IDEA
2A	Не все биты основного ключа используются для формирования раундового ключа, и знание раундового ключа не позволяет восстановить основной или расширенный ключ	DFC, CAST-128
2B	Все биты основного ключа используются для формирования каждого раундового ключа, но знание раундового ключа не позволяет восстановить основной или расширенный ключ	Blowfish, LOKI-97, Serpent, CAST-256, Twofish, RC6, E2, Mars, Frog, HPC
2C	Каждый раундовый ключ формируется независимо от остальных раундовых ключей, и размерность расширенного ключа совпадает с размерностью основного ключа	DESI (DES с независимыми ключами)

Достаточно очевидно, что наиболее безопасными согласно данной классификации являются блочные шифры с индексами 2B и 2C. Вместе с тем, разработка стойкого блочного шифра, соответствующего индексу 2C с ограниченной длиной (например, 256 битов), представляется достаточно сложной задачей, поскольку в этом случае произведение числа раундов на длину раундового ключа равна длине основного ключа и является константой. По этой причине алгоритмов, относящихся к данной группе и используемых на практике, не отмечено.

Использование стойких (сильных) процедур генерации расширенного ключа является элементом, призванным повысить стойкость шифра, или, по крайней мере, упростить доказательную базу этой стойкости. Действительно, наличие такой процедуры позволяет рассматривать раундовые ключи в качестве независимых равновероятно распределенных случайных величин. Соответственно, криптографический анализ схемы, как правило, сводится к анализу одного раунда, включая вычисление необходимых характеристик одного раунда и последующие обобщения для заданного числа раундов.

Авторами книги была осуществлена разработка стойкой процедуры генерации расширенного ключа с использованием управляемых операций [11]. В частности, если применять в качестве управляемых операций блоки управляемых перестановок, то для формирования одного раундового ключа длиной 256 битов необходимо затратить время, сравнимое с последовательным выполнением десяти операций типа XOR. Данную процедуру можно использовать, например, в алгоритме SPECTR-128, что при аппаратной реализации алгоритма повлечет дополнительное увеличение времени работы алгоритма, примерно равное двум раундам.

Современные рекомендации о необходимости использования стойкой процедуры генерации расширенного ключа обоснованы, в первую очередь, многолетними исследованиями ключевой системы алгоритма DES и разработкой специальных методик вскрытия блочных шифров, учитывающих выявленные слабости в формировании или использовании раундовых ключей. В частности, существуют такие понятия, как "слабые ключи", "полуслабые ключи", "криптоэквивалентные ключи". При разработке блочных шифров такие рекомендации учитывать необходимо, особенно в случае, когда размерность раундового ключа сравнительно небольшая, то есть поддается полному перебору.

5.1.3. Практика использования расписания ключей

С другой стороны, использование стойкой процедуры генерации расширенного ключа приводит к снижению скорости преобразования данных, в некоторых случаях весьма существенному, поскольку требуется дополнительное время на генерацию очередного ключа. Данное обстоятельство особенно критично для систем защиты информации, в которых используется режим частой смены ключа, а объем преобразуемой информации сравнительно небольшой. В этом случае для

формирования расширенного ключа предпочтительным является применение расписания ключей.

В любом случае использование расписания ключей оправдано тогда, когда существенным параметром является быстродействие алгоритма, поскольку дополнительная задержка определяется только временем ввода очередного ключа.

К формированию расписания должны предъявляться жесткие требования, учитывающие особенности структуры алгоритма, а также взаимодействие криптографических примитивов как внутри раунда, так и между раундами. Более того, при разработке алгоритма блочного шифра изначально необходимо учитывать тот факт, что будет использоваться расписание ключей.

Одним из приемов, снижающих вероятность возникновения слабых ключей, является увеличение размерности раундового ключа. В этом случае, задача вычисления раундового ключа может быть не менее сложной, чем вычисление основного ключа. Именно данный подход применен при разработке скоростных шифров типа SPECTR. Варианты таких расписаний можно найти в соответствующих разделах описаний конкретных моделей алгоритмов блочных шифров.

Начальное и заключительное преобразования

Возвращаясь к анализу формул (5.2)—(5.4), необходимо отметить, что в большинстве блочных шифров, предложенных разработчиками за последнее десятилетие, использование формул (5.3) и (5.4) в "чистом виде" практически не встречается. То есть процедуры начального $\mathbf{E}_K^{(1)}$ и заключительного преобразований $\mathbf{E}_K^{(r)}$ отличаются от остальных "внутренних" преобразований. Вполне вероятно, что необходимость в таком различии явилась следствием разработки специальной методики линейного криптоанализа алгоритма DES, автором которой является Мацуи. В соответствии с данной методикой для специальных характеристик в первом и последнем раунде используется перебор значений некоторых разрядов раундовых ключей (рассматривается полный перебор части раундового ключа в первом и последнем раунде).

Размерность ключей в начальном и заключительном преобразованиях может отличаться от соответствующей размерности раундовых ключей. Общепризнанно, что использование начального и заключительного преобразований без применения ключевых элементов не может повысить криптографическую стойкость алгоритма, поскольку в этом случае при наличии исходного и преобразованного блоков данных всегда можно определить значения блоков данных после выполнения начального и перед выполнением заключительного преобразования. То есть оценка стойкости данной криптосистемы сводится к решению аналогичной задачи, но с меньшим числом раундов. Начальное и заключительное преобразования с обязательным использованием ключевых элементов в зарубежной литературе получили название *"отбеливание"* (pre-whitening, post-whitening), поскольку основное предназначение данных преобразований заключается в сниже-

нии алгебраических и статистических закономерностей между битами входного и, соответственно, выходного блока данных. То есть преследуется цель, "приблизить" значения входных и выходных блоков данных "к случайным и равновероятно распределенным случайным величинам" и, тем самым, обеспечить условия для применения математических методов для доказательства криптографической стойкости алгоритма.

Таким образом, для современного итеративного блочного шифра формулы зашифрования и расшифрования имеют вид:

$$\mathbf{E}_K = \mathbf{E}^{(IT)}_{K^{(0)}} \circ \mathbf{E}'_{K^{(1)}} \circ \mathbf{E}'_{K^{(2)}} \circ \ldots \circ \mathbf{E}'_{K^{(r)}} \circ \mathbf{E}^{(FT)}_{K^{(r+1)}} \tag{5.5}$$

и

$$\mathbf{E}^{-1}_K = (\mathbf{E}^{(FT)})^{-1}_{K^{(r+1)}} \circ (\mathbf{E}')^{-1}_{K^{(r)}} \circ \ldots \circ (\mathbf{E}')^{-1}_{K^{(2)}} \circ (\mathbf{E}')^{-1}_{K^{(1)}} \circ (\mathbf{E}^{(IT)})^{-1}_{K^{(0)}}. \tag{5.6}$$

Расширенный ключ можно представить в следующем виде:

$$Q^{(e)} = \mathbf{H}(K, e) = Q^{(e,\,0)} \parallel Q^{(e,\,1)} \parallel Q^{(e,\,2)} \parallel \ldots \parallel Q^{(e,\,r)} \parallel Q^{(e,\,r+1)}, \tag{5.7}$$

где e — режим преобразования данных ($e = 0$ — прямое и $e = 1$ — обратное преобразование данных), K — основной ключ и $\forall j = 0, 1, 2, \ldots, r, r + 1$,

$$Q^{(e,\,j)} = K^{(j + e(r + 1 - 2j))}. \tag{5.8}$$

Отсюда зависимость между раундовыми ключами зашифрования и расшифрования имеет вид:

$$Q^{(1,\,j)} = Q^{(0,\,r + 1 - j)}. \tag{5.9}$$

При *программной реализации* выбор раундовых ключей наиболее просто задается в виде двух таблиц (для $e = 0$ и $e = 1$, соответственно), в которых значения элементов расширенного ключа представлены в явном виде, а заполнение таблиц зависит от используемой процедуры $\mathbf{H}(K, e)$.

Покажем теперь вариант эффективной аппаратной реализации выбора раундовых ключей. Пусть с использованием процедуры $\mathbf{H}(K, e)$ сформирована последовательность $Q^{(0)}$, которая размещена в некотором регистре S. А именно:

$$S^{(0)} \parallel S^{(1)} \parallel S^{(2)} \parallel \ldots \parallel S^{(r)} \parallel S^{(r+1)} = Q^{(0,\,0)} \parallel Q^{(0,\,1)} \parallel Q^{(0,\,2)} \parallel \ldots \parallel Q^{(0,\,r)} \parallel Q^{(0,\,r+1)}. \tag{5.10}$$

То есть $S = \mathbf{H}(K, 0) = \mathbf{H}'(K)$. С учетом формулы (5.8) изменение режима шифрования влечет в последовательности S изменение порядка следования раундовых ключей на противоположное, т. е. равноудаленные (от середины) раундовые ключи необходимо поменять местами. Такая процедура эффективно решается с использованием блоков управляемых перестановок $\mathbf{P}_{2/1}$, причем их число должно быть равно половине размерности расширенного ключа. Так, например, если ω — размерность раундового ключа, тогда блок $\mathbf{P}_{2\omega/1}$ позволяет обеспечить перестановку раундовых ключей $S^{(j)}$ и $S^{(r - j + 1)}$, причем в один блок $\mathbf{P}_{2/1}$ должны быть сведены одноименные разряды, подлежащие перестановке.

Рис. 5.1. Пример схемы управления выбором раундовых ключей

На рис. 5.1 представлен пример блока управления выбором раундовых ключей в зависимости от режима преобразования данных (e) для случая $r = 12$, $\omega(S^{(j)}) = 32$, $j = 1, 2, \ldots, 12$ и $\omega(S^{(0)}) = \omega(S^{(13)}) = 16$. Условие одинаковой размерности для всех раундовых ключей не является существенным, а использовано лишь для наглядности. Специальное обозначение $\mathbf{P}_{2\times\omega/1}$ для однослойного блока управляемых перестановок лишь подчеркивает особую структуру блока, предназначенного в зависимости от значения управляющего разряда e поменять или не поменять местами два бинарных вектора длины ω. Для рассматриваемой r-раундовой модели шифра отображение $\mathbf{P}_{2\times\omega/1}$ имеет вид:

$$(O^{(i)} \parallel O^{(r+1-i)}) = \mathbf{P}_{2\times\omega/1}(S^{(i)} \parallel S^{(r+1-i)}, e),$$

где $i = 0, 1, \ldots, r + 1$, причем значения j-х разрядов векторов $O^{(i)}$ и $O^{(r+1-i)}$ вычисляются с использованием следующих формул:

$$o_j^{(i)} = (o_j^{(i)} \oplus o_j^{(r+1-i)})e \oplus o_j^{(i)} \quad \text{и} \quad o_j^{(r+1-i)} = (o_j^{(i)} \oplus o_j^{(r+1-i)})e \oplus o_j^{(r+1-i)}.$$

На схеме последовательность $O = O^{(0)} \parallel O^{(1)} \parallel O^{(2)} \parallel \ldots \parallel O^{(12)} \parallel O^{(13)}$ для $e = 0$ соответствует расширенному ключу $Q^{(0)}$, а для $e = 1$ — расширенному ключу $Q^{(1)}$.

В аппаратной реализации $O^{(j)}$ — это шины данных раундовых ключей.

Очевидно, что выбор раундовых ключей при переходе из режима зашифрования ($e = 0$) в режим расшифрования ($e = 1$) и обратно осуществляется с временем задержки $\tau(\mathbf{P}_{2/1})$. Таким образом, предложенная реализация обеспечивает эффективное использование алгоритма в случае частой смены режима преобразования данных.

Замечание

Если r — нечетное число, тогда для раундового ключа $S^{(j)}$ с номером $(r + 1)/2$ перестановка не требуется, поскольку в этом случае $j = r - j + 1$.

Рассмотрим вариант использования расписания ключей. В этом случае каждый разряд регистра S является значением некоторого разряда регистра, в котором хранится основной ключ K, то есть между этими двумя регистрами организуется некоторая фиксированная коммутация. Но тогда нет необходимости в резервировании специального регистра S, поскольку коммутация между регистром K и однослойными блоками управляемых перестановок может быть выполнена напрямую.

Дальнейшая детализация формул шифрования зависит от типа схемы шифрования, которые рассматриваются ниже. Соответственно будет уточнена и схема выбора раундовых ключей в зависимости от режима шифрования.

5.1.4. Два основных типа схем шифрования блочных шифров

Схемы шифрования современных блочных шифров условно можно разделить на два основных типа:

1. Зашифрование и расшифрование реализовано с использованием одной и той же схемы.
2. Зашифрование и расшифрование реализовано с использованием разных схем.

Главным недостатком схем второго типа является необходимость реализации и анализа двух взаимосвязанных алгоритмов: алгоритма зашифрования и алгоритма расшифрования. Схемы, относящиеся к этому типу, не рассматриваются, поскольку авторы книги разработкой и анализом таких схем практически не занимались.

Замечание

Алгоритм Rijndael, победивший в конкурсе на новый стандарт шифрования (Advanced Encryption Standard, AES), относится именно к этому типу алгоритмов. Справедливости ради необходимо отметить, что алгоритм сконструирован таким образом, что большинство преобразований в алгоритмах зашифрования и расшифрования, а также их последовательность совпадают. В данной книге алгоритм не анализируется, главным образом, по причине отсутствия в нем управляемых операций.

Наибольшее распространение получили схемы первого типа, называемые *универсальными*, обладающие более низкой себестоимостью при аппаратной реализации. В таких схемах при выполнении обратного преобразования, как правило, меняется лишь порядок выбора ключевой информации. В *разд. 5.1.5* рассматривается запатентованная схема блочного шифрования одного раунда (патент РФ № 2140714), совмещающая свойства схемы Фейстеля и итеративной схемы общего вида. В *разд. 5.1.6* описываются общие элементы алгоритмов блочного шифрования, в которых в качестве базовой схемы одного раунда используется усовершенствованная схема патента РФ № 2140714. В *разд. 5.1.7* рассматрива-

ются две универсальные итеративные схемы блочного шифрования, которые не относятся к классу шифров, использующих схему Фейстеля.

5.1.5. Усовершенствованная схема блочного шифрования одного раунда (патент РФ № 2140714)

Наиболее известной универсальной схемой является схема Фейстеля [66], в которой базовое преобразование $\mathbf{E}'$ одного раунда реализовано с использованием некоторой функции $\mathbf{F}$. При этом, как и в случае с преобразованием $\mathbf{E}'$, конкретный вид функции $\mathbf{F}$ зависит не только от ключа, но и от преобразуемых данных. Именно от выбора функции $\mathbf{F}$, а также от количества раундов r во многом зависит стойкость всего алгоритма.

В классической схеме Фейстеля (рис. 5.2, *а*), в i-м раунде правая часть R' преобразованных данных $L' \parallel R'$ является результатом групповой операции XOR ($\oplus$) со значением раундовой функции $\mathbf{F}_{K_i}$:

$$R' = R \oplus \mathbf{F}_{K_i}(L), \tag{5.11}$$

где K_i — раундовый ключ.

В итеративной схеме общего вида, представленной на рис. 5.2, *б*, в каждом раунде правая часть R' является значением раундовой функции $\mathbf{F}_K$:

$$R' = \mathbf{F}_{K_i}(L \parallel R). \tag{5.12}$$

Каждая из представленных схем имеет свои достоинства и недостатки. Поэтому рассмотрим их подробнее.

Существует два основных подхода к обеспечению необходимой стойкости алгоритма:

1. Построение сложных раундовых функций с небольшим числом раундов.
2. Использование простых функций с повышенным числом раундов.

Рис. 5.2. Один раунд преобразований в схеме Фейстеля (*а*) и в итеративной схеме (*б*)

В первом случае специальные параметры p_{DC} и q_{LC} раундовой функции, характеризующие ее криптографическую стойкость, соответственно, по отношению к методам дифференциального [44, 45] и линейного криптоанализа [43, 86], должны приближаться к минимально возможным значениям. В частности, для **S**-блоков размера $n \times n$ нижняя оценка для параметров p_{DC} и q_{LC} равна $2^{-(n-2)}$, т. е. для заданной длины n невозможно обеспечить значения параметров p_{DC} и q_{LC} меньше, чем $2^{-(n-2)}$. Увеличение значения n не только неизбежно влечет падение скоростных параметров, но и связано с техническими трудностями.

Поэтому при создании скоростного алгоритма блочного шифрования второй путь обычно является предпочтительным. Этот же подход к построению раундовых функций изложен в статье [4], однако авторы статьи, ориентируясь на программные алгоритмы, рассматривают только функции, которые уже реализованы в PC-процессорах.

Рассмотрим преобразование $\{X\} \to \{Y\}$, где $X = L \parallel R$ — входной блок, а $Y = L' \parallel R'$ — выходной блок и $\{X\} = \{Y\} = \mathrm{GF}(2)^n$. Для данного преобразования рассмотрим характеристическую функцию δ_{ij} зависимости значения выходного бита y_j от входного x_i, а именно $\delta_{ij} = 1$, если $\exists X$, инверсия бита x_i в котором влечет инверсию бита y_j.

Данная характеристическая функция отражает суть *критерия полноты* [75], определяющего зависимость выходных битов от входных битов. Применительно к одному раунду схемы Фейстеля (см. рис. 5.2, *а*) с раундовой функцией **F**, значения характеристической функции δ_{ij} можно представить в виде (0,1) — матрицы $\|\mathbf{A}\|$ размера $n \times n$, которая имеет следующий вид:

$$\|\mathbf{A}\|_{n\times n} = \begin{pmatrix} \|\mathbf{E}\| & \|\mathbf{B}\| \\ \|\mathbf{0}\| & \|\mathbf{E}\| \end{pmatrix},$$

где $\|\mathbf{B}\|$, $\|\mathbf{E}\|$ и $\|\mathbf{0}\|$ — матрицы размера $(n/2) \times (n/2)$, причем $\|\mathbf{0}\|$ — нулевая матрица, а матрица $\|\mathbf{E}\|$ — реализует тождественное преобразование (только $e_{ii} = 1$). Для схемы Фейстеля максимально возможное число единиц λ_{max} матрицы $\|\mathbf{A}\|$ равно $n^2/4 + n$, причем оно достигается в случае, когда каждый бит подблока L влияет на каждый бит подблока R', т. е. когда в матрице $\|\mathbf{B}\|$ все $b_{ij} = 1$. В частности, в DES матрица $\|\mathbf{B}\|$ содержит $3n$ единиц ($n = 64$), а для матрицы $\|\mathbf{A}\|$ значение $\lambda_{max} = 4n$, т. е. менее 1/4 от максимально возможного значения $n^2/4 + n$. Последующая перестановка местами левого и правого подблоков равносильна умножению матрицы $\|\mathbf{A}\|$ на матрицу $\|\mathbf{U}\|$, имеющую вид:

$$\|\mathbf{U}\|_{n\times n} = \begin{pmatrix} \|\mathbf{0}\| & \|\mathbf{E}\| \\ \|\mathbf{E}\| & \|\mathbf{0}\| \end{pmatrix}.$$

Но тогда в r-раундовой схеме Фейстеля соответствующая матрица является произведением $2r - 1$ матриц $\|\mathbf{A}\|$ и $\|\mathbf{U}\|$, а именно:

$$\|\mathbf{W}\| = (\|\mathbf{A}\| \circ \|\mathbf{U}\|)^{r-1} \circ \|\mathbf{A}\| = \|\mathbf{A}\| \circ \|\mathbf{U}\| \circ \|\mathbf{A}\| \circ \|\mathbf{U}\| \circ \ldots \circ \|\mathbf{U}\| \circ \|\mathbf{A}\|.$$

При этом для заданной функции **F** теоретически вычисляется минимальное число раундов, при котором каждый входной бит x_i влияет на каждый выходной бит y_j.

Для итеративной схемы (см. рис. 5.2, *б*) матрица **A** имеет следующий вид:

$$\|\mathbf{A}'\|_{n\times n}=\begin{pmatrix}\|\mathbf{E}\| & \|\mathbf{B}\| \\ \|\mathbf{0}\| & \|\mathbf{C}\|\end{pmatrix}.$$

Максимально возможное число единиц $\lambda_{max}(\|\mathbf{A}'\|) = n^2/2 + n/2$, причем оно достигается в случае, когда обеспечено влияние каждого бита всего входного блока ($X = L \,\|\, R$) на каждый бит выходного подблока R', т. е. тогда, когда в матрицах $\|\mathbf{B}\|$ и $\|\mathbf{C}\|$ все b_{ij} и c_{ij} равны единице. Соответственно, в матрице $\|\mathbf{W}\|$ для обеспечения влияния каждого входного бита на каждый выходной необходимо меньшее число раундов. В частности, такая схема используется в RC5, хотя схема, приведенная на рис. 5.2, *б*, в оригинале рассматривается как половина раунда. К недостаткам итеративной схемы, представленной на рис. 5.2, *б*, необходимо отнести требование *обратимости* функции **F**, что ограничивает множество возможных функций.

Рассмотрим усовершенствованную итеративную схему блочного преобразования данных (рис. 5.3, *а*), предложенную в патенте РФ № 2140714 [1]. Значение вектора R' для схемы патента вычисляется по формуле:

$$R' = \mathbf{G}_U^{-1}(\mathbf{G}_V(R) \oplus \mathbf{F}_{K_i}(L)), \tag{5.13}$$

Рис. 5.3. Один раунд преобразований в базовой (*а*) и детализированной (*б*) схеме патента РФ № 2140714

где $\mathbf{G}_V$ и $\mathbf{G}_U^{-1}$ — взаимно обратные преобразования, конкретный вид которых определяется управляющими векторами V и U, т. е. последовательно выполняются преобразования $\mathbf{G}_V$, XOR и $\mathbf{G}_U^{-1}$.

В общем случае управляющие векторы V и U являются значениями некоторой процедуры $\mathbf{E}$ от двух переменных (рис. 5.3, *б*): блока данных L и раундового ключа K_V (или K_U), т. е. $V = \mathbf{E}(L, K_V)$ и $U = \mathbf{E}(L, K_U)$. Максимально возможное число единиц $\lambda_{max}(\|\mathbf{A}'\|)$ для данной схемы также равно $n^2/2 + n/2$, однако в этом случае значительно легче достигается независимость между разрядами выхода блока R'.

Наличие двух взаимно обратных преобразований $\mathbf{G}_V$, и $\mathbf{G}_U^{-1}$ обеспечивает возможность применения для прямого и обратного преобразования одной и той же схемы, однако порядок использования ключей в этом случае более сложный.

Действительно, покажем сначала универсальность схемы на примере *одного раунда*.

Раундовый ключ $Q^{(e,\,1)}$ для режима зашифрования ($e = 0$) представим в виде объединения трех раундовых подключей K_V, K_F и K_U, а именно:

$$Q^{(e,\,1)} = K_V \,||\, K_F \,||\, K_U,$$

Тогда раундовый ключ $Q^{(e,\,1)}$ для режима расшифрования ($e = 1$) имеет вид:

$$Q^{(e,\,1)} = K_U \,||\, K_F \,||\, K_V.$$

То есть для обратного преобразования раундовые подключи необходимо выписать в обратной последовательности.

С учетом формул (5.7—5.9), если расширенный ключ $Q^{(0)}$ представить в следующем виде:

$$Q^{(0)} = K^{(0)} \| K_V^{(1)} \| K_F^{(1)} \| K_U^{(1)} \| K_V^{(2)} \| K_F^{(2)} \| K_U^{(2)} \| \ldots \| K_V^{(r)} \| K_F^{(r)} \| K_U^{(r)} \| K^{(r+1)}, \qquad (5.14)$$

тогда расширенный ключ $Q^{(1)}$ для режима расшифрования имеет вид:

$$Q^{(1)} = K^{(r+1)} \| K_U^{(r)} \| K_F^{(r)} \| K_V^{(r)} \| \ldots \| K_U^{(2)} \| K_F^{(2)} \| K_V^{(2)} \| K_U^{(1)} \| K_F^{(1)} \| K_V^{(1)} \| K^{(0)}. \qquad (5.15)$$

То есть достаточно изменить порядок следования раундовых подключей на противоположный. Очевидно, что и в этом случае эффективная аппаратная реализация выбора раундовых подключей в случае изменения режима шифрования также решается с использованием однослойных блоков управляемых перестановок $\mathbf{P}_{2\times\omega/1}$.

На рис. 5.4 приведен пример для случая $\omega(K_F) = 32$ и $\omega(K_V) = \omega(K_U) = 64$.

Рис. 5.4. Пример схемы управления выбором раундовых подключей

В заключение необходимо отметить, что при составлении расписания с учетом особенностей базовой схемы патента РФ № 2140714 (см. рис. 5.3) необходимо обеспечить выполнение следующих требований, несоблюдение которых может ослабить стойкость алгоритма. А именно:

- для исключения вариантов $V^{(i)} = U^{(i)}$, при которых реализуются взаимно обратные модификации $\mathbf{P}_{V^{(i)}}$ и $\mathbf{P}^{-1}_{U^{(i)}}$, т. е. $\mathbf{P}^{-1}_{U^{(i)}}(\mathbf{P}_{V^{(i)}}(R)) = R$, в одном раунде необходимо использовать разные ключи $K_V^{(i)}$ и $K_U^{(i)}$;
- для исключения вариантов $V^{(i+2)} = U^{(i)}$, при которых реализуются взаимно обратные модификации $\mathbf{P}^{-1}_{U^{(i)}}$ и $\mathbf{P}_{V^{(i+2)}}$, т. е. $\mathbf{P}_{V^{(i+2)}}(\mathbf{P}^{-1}_{U^{(i)}}(R)) = R$, необходимо использовать разные ключи $K_V^{(i+2)}$ и $K_U^{(i)}$.

5.1.6. Общие элементы алгоритмов блочного шифрования на основе управляемых операций

Структура базовой модели криптосхемы итеративного *r*-раундового алгоритма блочного шифрования

Для блочных шифров SPECTR-H64, SPECTR-128 и CIKS-128 в качестве базовой модели *r*-раундового алгоритма была выбрана схема, представленная на рис. 5.5.

Функция преобразования данных имеет вид:

$$Y = \mathbf{F}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от секретного ключа $K = (K_1, K_2, \ldots, K_s)$ и режима преобразования данных e ($e = 0$ — прямое преобразование данных, $e = 1$ — обратное преобразование). Расширенный ключ является объединением $r + 2$ рабочих подключей:

$$Q^{(e)} = Q_{IT}^{(e)} \parallel Q_1^{(e)} \parallel Q_2^{(e)} \parallel \ldots \parallel Q_r^{(e)} \parallel Q_{FT}^{(e)} = (Q_{IT}^{(e)}, Q_1^{(e)}, Q_2^{(e)}, \ldots, Q_r^{(e)}, Q_{FT}^{(e)}),$$

где $Q_{IT}^{(e)}, Q_{FT}^{(e)} \in \mathrm{GF}(2)^{n/2}$, а структура и размерность остальных ключей зависят от конкретной модели.

Рис. 5.5. Базовая модель алгоритма итеративного блочного шифра

Функция **F** реализована в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. r раундов (циклов) *преобразований* с использованием процедуры ***Crypt***.
3. Заключительного преобразования ***FT***.

Первоначально выполняется процедура ***IT***: $Y = \boldsymbol{IT}(X, Q_{IT}^{(e)})$.

Блок Y делится на два подблока L_0 и R_0 одинаковой длины, т. е. $(L_0, R_0) = Y$, где $L_0, R_0 \in \mathrm{GF}(2)^{n/2}$. Затем выполняются r раундов преобразований с использованием процедуры ***Crypt*** в соответствии с формулами:

$$L_j = \boldsymbol{Crypt}(R_{j-1}, L_{j-1}, Q_j^{(e)});$$

$$R_j = L_{j-1}, \ (j = 1, 2, \ldots, r).$$

После r-го раунда над блоком $X = (R_r, L_r)$ выполняется заключительное преобразование ***FT*** в соответствии с формулой: $Y = \boldsymbol{FT}(X, Q_{FT}^{(e)})$.

Начальное и заключительное преобразования

Начальное преобразование ***IT*** имеет следующий вид:

$$Y = \boldsymbol{IT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^n$, $A \in \mathrm{GF}(2)^{n/2}$.

На рис. 5.6 приведена схема реализации данного преобразования, при этом каждая пара битов входного блока X с индексами $2j$–1 и $2j$ (j = 1, 2, ..., n/2) либо переставляется ($a_j = 1$), либо не переставляется ($a_j = 0$), после чего каждый четный бит инвертируется. Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j} \oplus 1.$$

Рис. 5.6. Начальное преобразование ($s = n/2$)

Процедура ***FT*** (рис. 5.7) является обратным преобразованием по отношению к процедуре ***IT***. Преобразование имеет следующий вид:

$$Y = \boldsymbol{FT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^n$, $A \in \mathrm{GF}(2)^{n/2}$.

Рис. 5.7. Заключительное преобразование ($s = n/2$)

Первоначально каждый четный бит входного блока инвертируется (рис. 5.7), а затем каждая пара битов входного блока X с индексами $2j - 1$ и $2j$ ($j = 1, 2, \ldots, n/2$) либо переставляется ($a_j = 1$), либо не переставляется ($a_j = 0$).

Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j} \oplus 1.$$

Использование в этой базовой модели начального и заключительного преобразований скорее "дань моде", нежели стремление повысить стойкость алгоритма. В данном случае это лишь "демонстрация знания" современных направлений в разработке алгоритмов блочных шифров.

5.1.7. Универсальные схемы блочного шифрования на основе взаимно обратных управляемых операций

Рассматриваются две универсальные итеративные схемы блочного шифрования, отличающиеся по своей структуре от схемы Фейстеля. Как отмечалось ранее, достоинством широко используемой схемы Фейстеля (см. рис. 5.2, *а*), является возможность выполнять зашифрование и расшифрование с помощью одной и той же последовательности операций, причем изменение режима шифрования достигается изменением очередности использования раундовых подключей. Данное свойство обеспечивает экономичность аппаратной реализации. С другой сторо-

ны, определенным недостатком криптосхемы Фейстеля является тот факт, что в одном раунде преобразованию подвергается только один из двух блоков входных данных (R), причем второй блок данных (L) "лишь принимает участие" в таком преобразовании, но его значение остается неизменным. Более того, преобразование имеет "достаточно простой вид", а именно $R' = R \oplus \mathbf{F}(L, K)$, поскольку раундовая функция **F** зависит только от двух параметров (L и раундового ключа K).

Рассмотрим криптосхемы, в которых преобразование одного из блоков данных имеет наиболее общий вид, а именно:

$$R' = \mathbf{G}(R, L, K). \tag{5.16}$$

Формула (5.16) соответствует рис. 5.2, *б*, причем единственное требование, предъявляемое к такой функции **G**, — это наличие обратной функции по первому параметру, а именно $R = \mathbf{G}^{-1}(R', L, K)$.

Более того, ограничимся рассмотрением криптосхем, обладающих свойством универсальности, в которых в качестве функций **G** и $\mathbf{G}^{-1}$ используются управляемые операции.

В соответствии с общим определением функции второй и третий параметры не являются обязательными, что может быть выражено следующим образом:

$$Y = \mathbf{G}(X <, U> <, V>), \tag{5.17}$$

где запись в угловых скобках означает, что данная запись может отсутствовать. Для записи (5.17) возможны четыре варианта.

- Вариант 1:

  $$Y = \mathbf{G}(X).$$

  Данная запись означает использование унарной операции, к которой, например, относится фиксированная перестановка ($Y = \mathbf{P}_{\pi}(X)$) или подстановка ($Y = \boldsymbol{\sigma}(X)$).

- Вариант 2:

  $$Y = \mathbf{G}(X, U, V).$$

  При рассмотрении функции от трех переменных будем подразумевать, что третий параметр (V) всегда является управляющим вектором, а сама функция реализует управляемую двухместную операцию. То есть размерности первого (X) и второго (U) параметров одинаковые, в то время как размерность третьего параметра (V) может отличаться от размерности первых двух параметров. С учетом записи (5.16) второй и третий параметры можно определить следующим образом:

  $$U = \mathbf{F}(L, K) \quad \text{и} \quad V = \mathbf{E}(L, K). \tag{5.18}$$

Присутствие одного и того же обозначения K для второго параметра не должно вызывать каких-либо вопросов по поводу зависимости переменных, поскольку ключ K всегда можно представить в виде объединения подключей:

$$K = K_1 \| K_2. \tag{5.19}$$

Принципиальное различие между функциями **F** и **E** заключается в том, что функция **E** реализует процедуру расширения вектора L, в то время как функция **F** может иметь произвольный вид.

Примерами функции **F** являются следующие операции:

- поразрядное сложение $L \oplus K$;
- сложение по модулю n ($L + K \bmod 2^n$);
- вычитание по модулю n ($L - K \bmod 2^n$);
- раундовая функция в DES.

❒ Вариант 3:

$$Y = \mathbf{G}(X, U).$$

Данный вариант аналогичен функции $\mathbf{F}(L, K)$ с тем отличием, что, обязательным требованием для функции **G** остается "обратимость" по первому параметру.

❒ Вариант 4:

$$Y = \mathbf{G}(X, V).$$

В данном варианте функция **G** реализует одноместную управляемую операцию. К ним относятся управляемые блоки перестановок ($\mathbf{P}(X, V)$) и управляемые подстановочные операции.

Ниже рассматриваются две универсальные криптосхемы, построенные с использованием управляемых операций, общий вид которых представлен формулой (5.17).

Модель 1

Конструктивной особенностью рассматриваемой схемы является использование в одном раунде (рис. 5.8, *а*) *четного числа* управляемых операций $\mathbf{G}_i$ и $\mathbf{G'}_i$, причем каждый из двух подблоков данных подвергается преобразованию одинаковое число раз.

Для r-раундового итеративного блочного шифра алгоритм шифрования определяется последовательностью следующих действий:

1. Первоначально блок X делится на два подблока X_{lo} и X_{hi} одинаковой длины, т. е. $(X_{lo}, X_{hi}) = X$, и выполняется операция присваивания $(A_0, B_0) = (X_{lo}, X_{hi})$.

2. Затем с использованием процедуры ***Crypt*** одного раунда выполняются r раундов преобразований в соответствии с формулами:

$$(A_j, B_j) := \boldsymbol{Crypt}(A_{j-1}, B_{j-1}, Q_j^{(e)}),$$

где $j = 1, 2, \ldots, r$.

3. Значение $Y = (A_r, B_r)$ является преобразованным блоком данных.

Рис. 5.8. Общая (*а*) и частная (*б*) криптосхемы реализации процедуры ***Crypt*** одного раунда

Если раундовый ключ $Q_j^{(e)}$ представить в виде последовательности раундовых подключей ($Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, \ldots, Q_j^{(2s,e)})$), причем такая последовательность совпадает с очередностью их использования в процедуре ***Crypt***, то тогда соответствие между раундовыми ключами зашифрования и расшифрования имеет следующий вид:

$$Q_{r+1-j}^{(1)} = (Q_{r+1-j}^{(1,1)}, Q_{r+1-j}^{(2,1)}, \ldots, Q_{r+1-j}^{(2s,1)}) = (Q_j^{(2s,0)}, Q_j^{(2s-1,0)}, \ldots, Q_j^{(1,0)}). \qquad (5.20)$$

При такой структуре шифра процедура ***Crypt*** должна обладать свойством обратимости одного раунда:

$$\boldsymbol{Crypt}(\boldsymbol{Crypt}(A_j, B_j, Q_j^{(e)}), Q_j^{(1-e)}) = (A_j, B_j).$$

Кроме того, для заданной структуры одного раунда (рис. 5.8) совокупность управляемых операций $\mathbf{G}_1$, $\mathbf{G}_2$, ..., $\mathbf{G}_s$ и $\mathbf{G}'_1, \mathbf{G}'_2, ..., \mathbf{G}'_s$, для $i = 1, 2, ..., s$ должна обладать следующим свойством взаимного соответствия:

$$\mathbf{G}'_i = \mathbf{G}^{-1}_{s+1-i}, \tag{5.21}$$

При этом должны использоваться одинаковые функции $\mathbf{F}'_i$ и $\mathbf{E}'_i$, а именно:

$$\mathbf{F}'_i(L, K) = \mathbf{F}_{s+1-i}(L, K) \text{ и } \mathbf{E}'_i(L, K) = \mathbf{E}_{s+1-i}(L, K).$$

Фактически для любых L и K должно выполняться следующее равенство:

$$\mathbf{G}'_i(R, \mathbf{F}'_i(L, K), \mathbf{E}'_i(L, K)) = \mathbf{G}^{-1}_{s+1-i}(R, \mathbf{F}_{s+1-i}(L, K), \mathbf{E}_{s+1-i}(L, K)).$$

Алгоритм одного раунда имеет следующий вид:

1. Выполнить операцию присваивания $(A^{(0)}, B^{(0)}) := (A_{j-1}, B_{j-1})$ и установить значение счетчика $i := 1$.
2. Вычислить векторы $U'_i := \mathbf{F}_i(B^{(i-1)}, Q_j^{(2i-1,e)})$ и $V'_i := \mathbf{E}_i(B^{(i-1)}, Q_j^{(2i-1,e)})$.
3. Преобразовать $A^{(i)} := \mathbf{G}_i(A^{(i-1)}, U'_i, V'_i)$.
4. Вычислить векторы $U''_i := \mathbf{F}'_i(A^{(i)}, Q_j^{(2i,e)})$ и $V''_i := \mathbf{E}'_i(A^{(i)}, Q_j^{(2i,e)})$.
5. Преобразовать $B^{(i)} := \mathbf{G}'_i(B^{(i-1)}, U''_i, V''_i)$.
6. Увеличить значение счетчика $i := i + 1$. Если $i > s$, тогда перейти к шагу 2.
7. Выполнить операцию присваивания $(A_j, B_j) := (B^{(r)}, A^{(r)})$.
8. **СТОП**.

Рассмотрим конкретной пример использования данной модели для построения 8-раундового 64-битового блочного шифра с длиной секретного ключа, равной 256 битам.

Алгоритм 5.1

С учетом схемы одного раунда (рис. 5.8, *б*) в j-м раунде используются два 32-битовых подключа $Q_j^{(1,e)}$ и $Q_j^{(2,e)}$, поэтому длина расширенного ключа равна 16- и 32-разрядным словам. Пусть такой расширенный ключ для режима зашиф-

рования сформирован с использованием некоторой процедуры генерации расширенного ключа $\mathbf{H}(K)$, а именно:

$$Q^{(0)} = (Q_1^{(1,0)}, Q_1^{(2,0)}, Q_2^{(1,0)}, Q_2^{(2,0)}, ..., Q_8^{(1,0)}, Q_8^{(2,0)}) = \mathbf{H}(K). \tag{5.22}$$

Для режима расшифрования необходимо использовать следующую формулу:

$$(Q_{9-j}^{(1,1)}, Q_{9-j}^{(2,1)}) = (Q_j^{(2,0)}, Q_j^{(1,0)}),$$

где $j = 1, 2, \ldots, 8$, то есть

$$Q^{(1)} = (Q_8^{(2,0)}, Q_8^{(1,0)}, Q_7^{(2,0)}, Q_7^{(1,0)}, ..., Q_1^{(2,0)}, Q_1^{(1,0)}). \tag{5.23}$$

На схеме (рис. 5.8, *б*) используются следующие операции:

- операции $\mathbf{G}_1$ и $\mathbf{G}_2$ — обратимые по первому параметру управляемые двухместные операции, рассматривавшиеся в работе [16];
- операции $\mathbf{P}$ и $\mathbf{P}^{-1}$ — блоки управляемых перестановок $\mathbf{P}_{32/80}$ и $\mathbf{P}_{32/80}^{-1}$ [14, 68];
- фиксированная перестановка $\mathbf{P}_\pi$ — циклический сдвиг данных на 11 в сторону младших разрядов;
- $\mathbf{E}(B)$ — процедура расширения управляющего вектора, имеющая следующий вид: $V = B \parallel B \parallel B_{hi}$, где $B = B_{lo} \parallel B_{hi}$.

Алгоритм реализации блочного шифра шифрования можно представить в следующем виде:

1. Выбрать режим шифрования ($e = 0$ — зашифрование, $e = 1$ — расшифрование), сформировать расширенный ключ $Q^{(e)}$ в соответствии с формулами (5.22) и (5.23) и выполнить операцию присваивания $(A_0, B_0) := (X_{lo}, X_{hi})$, где (X_{lo}, X_{hi}) — 64-разрядный исходный блок данных.
2. Установить значение счетчика числа раундов $j := 1$.
3. Выполнить операции присваивания $(A^{(0)}, B^{(0)}) := (A_{j-1}, B_{j-1})$ и $(W^{(1)}, W^{(2)}) := (Q_j^{(1,e)}, Q_j^{(2,e)})$.
4. Вычислить вектор $V_1 := B^{(0)} \oplus W^{(2)}$.
5. Преобразовать $A^{(1)} := \mathbf{G}_1(A^{(0)}, W^{(1)}, V_1)$.
6. Вычислить вектор $V_2 := \mathbf{P}_\pi(A^{(1)})$.
7. Преобразовать $B^{(1)} := \mathbf{G}_2^{-1}(B^{(0)}, A^{(1)}, V_2)$.
8. Вычислить вектор $V_3 := \mathbf{E}(B^{(1)})$.
9. Преобразовать $A^{(2)} := \mathbf{P}(A^{(1)}, V_3)$.

10. Вычислить вектор $V_4 := \mathbf{E}(A^{(2)})$.
11. Преобразовать $B^{(2)} := \mathbf{P}(B^{(1)}, V_4)$.
12. Вычислить вектор $V_5 := \mathbf{P}_\pi(B^{(2)})$.
13. Преобразовать $A^{(3)} := \mathbf{G}_2(A^{(2)}, B^{(2)}, V_5)$.
14. Вычислить вектор $V_6 := A^{(3)} \oplus W^{(1)}$.
15. Преобразовать $B^{(3)} := \mathbf{G}_1^{-1}(A^{(3)}, W^{(2)}, V_6)$.
16. Выполнить операцию присваивания $(A_j, B_j) := (B^{(3)}, A^{(3)})$.
17. Увеличить значение счетчика $j := j + 1$. Если $j > r$, тогда перейти к шагу 3.
18. **СТОП**.

Значение $Y = (A_r, B_r)$ является преобразованным блоком данных.

Модель 2

Конструктивной особенностью рассматриваемой схемы является использование в одном раунде (рис. 5.9, *а*) *нечетного числа* управляемых операций $\mathbf{G}_i$ и $\mathbf{G}'_i$, причем правый блок данных подвергается преобразованиям на одну операцию меньше, чем левый.

Для r-раундового итеративного блочного шифра алгоритм шифрования определяется последовательностью следующих действий:

1. Первоначально блок X делится на два подблока X_{lo} и X_{hi} одинаковой длины, т. е. $(X_{lo}, X_{hi}) = X$, и выполняется операция присваивания $(A_0, B_0) := (X_{lo}, X_{hi})$.
2. Затем с использованием процедуры ***Crypt*** одного раунда выполняются r раундов преобразований в соответствии с формулами:

$$(A_j, B_j) := \boldsymbol{Crypt}(B_{j-1}, A_{j-1}, Q_j^{(e)}),$$

   где $j = 1, 2, \ldots, r$.
3. Значение $Y = (B_r, A_r)$ является преобразованным блоком данных.

То есть в отличие от первой модели перестановка значений A_r и B_r в заключительном раунде не выполняется.

Все сказанное в отношении расширенного ключа $Q^{(e)}$ для первой модели справедливо и в данном случае, с той лишь разницей, что общее число подключей в одном раунде нечетно: $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, \ldots, Q_j^{(2s-1,e)})$. То есть:

$$Q_{r+1-j}^{(1)} = (Q_{r+1-j}^{(1,1)}, Q_{r+1-j}^{(2,1)}, \ldots, Q_{r+1-j}^{(2s-1,1)}) = (Q_j^{(2s-1,0)}, Q_j^{(2s-2,0)}, \ldots, Q_j^{(1,0)}).$$

Рис. 5.9. Общая (*а*) и частная (*б*) криптосхемы реализации процедуры ***Crypt*** одного раунда

Как и для первой модели, процедура ***Crypt*** должна обладать свойством обратимости:

$$\boldsymbol{Crypt}(\boldsymbol{Crypt}(A_j, B_j, Q_j^{(e)}), Q_j^{(1-e)}) = (A_j, B_j).$$

Кроме того, для заданной структуры одного раунда (см. рис. 5.8) совокупность управляемых операций $\mathbf{G}_1$, $\mathbf{G}_2$, ..., $\mathbf{G}_s$ и $\mathbf{G}'_1, \mathbf{G}'_2, ..., \mathbf{G}'_{s-1}$ для $i = 1, 2, ..., s$ должна обладать следующим свойством взаимного соответствия:

$$\mathbf{G}_{s+1-i} = \mathbf{G}_i^{-1} \text{ и } \mathbf{G}'_{s-i} = \mathbf{G}'^{-1}_i. \tag{5.24}$$

В частности, операция $\mathbf{G}_j$, индекс которой имеет значение $j = (s + 1) / 2$, является инволюцией. Например, операцией поразрядного сложения по модулю два ($\oplus$).

При этом для взаимно обратных функций, заданных формулами (5.24), должны использоваться одинаковые функции $\mathbf{F}_i$ и $\mathbf{E}_i$, а также $\mathbf{F}'_i$ и $\mathbf{E}'_i$, а именно:

$$\mathbf{F}_i(R, K) = \mathbf{F}_{s+1-i}(R, K) \quad \text{и} \quad \mathbf{E}_i(R, K) = \mathbf{E}_{s+1-i}(R, K)$$

и

$$\mathbf{F}'_i(L, K) = \mathbf{F}'_{s-i}(L, K) \quad \text{и} \quad \mathbf{E}'_i(L, K) = \mathbf{E}'_{s-i}(L, K).$$

Алгоритм одного раунда имеет следующий вид:

1. Выполнить операцию присваивания $(A^{(0)}, B^{(0)}) = (A_{j-1}, B_{j-1})$ и установить значение счетчика $i := 1$.
2. Вычислить векторы $U_i' := \mathbf{F}_i(B^{(i-1)}, Q_j^{(2i-1,e)})$ и $V_i' := \mathbf{E}_i(B^{(i-1)}, Q_j^{(2i-1,e)})$.
3. Преобразовать $A^{(i)} := \mathbf{G}_i(A^{(i-1)}, U_i', V_i')$.
4. Если $i = s$, тогда перейти к шагу 8.
5. Вычислить векторы $U_i'' := \mathbf{F}_i'(A^{(i)}, Q_j^{(2i,e)})$ и $V_i'' := \mathbf{E}_i'(A^{(i)}, Q_j^{(2i,e)})$.
6. Преобразовать $B^{(i)} := \mathbf{G}_i'(B^{(i-1)}, U_i'', V_i'')$.
7. Увеличить значение счетчика $i := i + 1$ и перейти к шагу 2.
8. Выполнить операцию присваивания $(A_j, B_j) := (A^{(r)}, B^{(r)})$.
9. **СТОП**.

Рассмотрим конкретный пример использования данной модели для построения 16-раундового 64-битового блочного шифра с длиной секретного ключа, равной 256 битам.

Алгоритм 5.2

С учетом схемы одного раунда (рис. 5.9, *б*) в j-м раунде используются два 32-битовых подключа $Q_j^{(1,e)}$ и $Q_j^{(2,e)}$, поэтому длина расширенного ключа равна 32-м 32-разрядным словам. Пусть такой расширенный ключ для режима зашифрования сформирован с использованием некоторой процедуры генерации расширенного ключа $\mathbf{H}(K)$, а именно:

$$Q^{(0)} = (Q_1^{(1,0)}, Q_1^{(2,0)}, Q_2^{(1,0)}, Q_2^{(2,0)}, ..., Q_{16}^{(1,0)}, Q_{16}^{(2,0)}) = \mathbf{H}(K). \tag{5.25}$$

Для режима расшифрования необходимо использовать следующую формулу:

$$(Q_{17-j}^{(1,1)}, Q_{17-j}^{(2,1)}) = (Q_j^{(2,0)}, Q_j^{(1,0)}),$$

где $j = 1, 2, \ldots, 16$, то есть

$$Q^{(1)} = (Q_{16}^{(2,0)}, Q_{16}^{(1,0)}, Q_{15}^{(2,0)}, Q_{15}^{(1,0)}, ..., Q_1^{(2,0)}, Q_1^{(1,0)}). \tag{5.26}$$

На схеме (рис. 5.9, *б*) используются следующие операции:

- операция $\mathbf{G}_1$ — обратимая по первому параметру управляемая двухместная операция, рассматривавшаяся в работе [16];
- операции $\mathbf{P}$ и $\mathbf{P}^{-1}$ — блоки управляемых перестановок $\mathbf{P}_{32/80}$ и $\mathbf{P}_{32/80}^{-1}$ [14, 68];

- операция $\mathbf{P}^*$ — управляемая перестановочная инволюция, подробно рассматривавшаяся в [15];
- $\mathbf{E}(B, W)$ — процедура расширения управляющего вектора, имеющая, например, следующий вид: $V = B \parallel B \oplus W \parallel ((B \oplus W)^{>>>11})_{hi}$, где операция $(B \oplus W)^{>>>11}$ означает циклический сдвиг данных на одиннадцать в сторону младших разрядов.

Алгоритм реализации блочного шифра шифрования можно представить в следующем виде:

1. Выбрать режим шифрования ($e = 0$ — зашифрование, $e = 1$ — расшифрование), сформировать расширенный ключ $Q^{(e)}$ в соответствии с формулами (5.25) и (5.26) и выполнить операцию присваивания $(A_0, B_0) := (X_{lo}, X_{hi})$, где (X_{lo}, X_{hi}) — 64-разрядный исходный блок данных.
2. Установить значение счетчика числа раундов $j := 1$.
3. Выполнить операции присваивания $(A^{(0)}, B^{(0)}) := (A_{j-1}, B_{j-1})$ и $(W^{(1)}, W^{(2)}) := (Q_j^{(1,e)}, Q_j^{(2,e)})$.
4. Вычислить вектор $V_1 := B^{(0)} \oplus W^{(2)}$.
5. Преобразовать $A^{(1)} := \mathbf{G}(A^{(0)}, W^{(1)}, V_1)$.
6. Вычислить вектор $V_2 := \mathbf{E}(A^{(1)}, W^{(2)})$.
7. Преобразовать $B^{(1)} := \mathbf{P}(B^{(0)}, V_2)$.
8. Вычислить вектор $V_3 := B^{(1)} \oplus W^{(1)} \oplus W^{(2)}$.
9. Преобразовать $A^{(2)} := \mathbf{P}^*(A^{(1)}, V_3)$.
10. Вычислить вектор $V_4 := \mathbf{E}(A^{(2)}, W^{(1)})$.
11. Преобразовать $B^{(2)} := \mathbf{P}^{-1}(B^{(1)}, V_4)$.
12. Вычислить вектор $V_5 := B^{(2)} \oplus W^{(1)}$.
13. Преобразовать $A^{(3)} := \mathbf{G}^{-1}(A^{(2)}, W^{(2)}, V_5)$.
14. Выполнить операцию присваивания $(A_j, B_j) := (B^{(3)}, A^{(3)})$.
15. Увеличить значение счетчика $j := j + 1$. Если $j < r$, тогда перейти к шагу 3.
16. **СТОП**.

Значение $Y = (B_r, A_r)$ является преобразованным блоком данных.

В заключение отметим, что рассмотренные в данном разделе модели свидетельствуют о многообразии схем построения блочных шифров на основе использования обратимых управляемых операций.

5.2. Блочный шифр SPECTR-H64

Алгоритм является практической реализацией основных идей синтеза скоростного блочного шифра на основе управляемых операций, рассматривавшихся в предыдущих главах, и ориентирован на скоростную обработку информации с 32-разрядной шиной обмена данными. Алгоритм опубликован в работах [14, 68].

5.2.1. Общая схема блочного шифрования

Для алгоритма SPECTR-H64 общая схема шифрования (зашифрование и расшифрование) определяется формулой:

$$Y = \mathbf{F}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от 256-битового секретного ключа K и режима шифрования e ($e = 0$ — зашифрование, $e = 1$ — расшифрование), причем в режиме зашифрования X является исходным блоком двоичных данных (открытым текстом), а в режиме расшифрования — преобразованным блоком двоичных данных (шифртекстом). В режиме зашифрования результирующее значение Y является шифртекстом, а в режиме расшифрования — открытым текстом.

Секретный ключ K представлен в виде объединения восьми 32-разрядных подключей: $K = K_1 \| K_2 \| \ldots \| K_8$, где и $K_1, K_2, \ldots, K_8 \in \mathrm{GF}(2)^{32}$.

Описание алгоритма шифрования (функции **F**) приведено в *разд. 5.2.2*, а процедура формирования рабочего ключа $Q^{(e)} = \mathbf{H}(K, e)$ в *разд. 5.2.3*.

5.2.2. Алгоритм шифрования

Функция **F** (см. рис. 5.5) реализована в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. 12 раундов (циклов) *преобразований* с использованием процедуры ***Crypt***.
3. Заключительного преобразования ***FT***.

Первоначально выполняется процедура ***IT***: $Y = \boldsymbol{IT}(X, Q_{IT}^{(e)})$.

Блок Y делится на два подблока L_0 и R_0 одинаковой длины, т. е. $(L_0, R_0) = Y$, где $L_0, R_0 \in \mathrm{GF}(2)^{32}$. Затем выполняются 12 раундов преобразований с использованием процедуры ***Crypt*** в соответствии с формулами:

$$L_j = \boldsymbol{Crypt}(R_{j-1}, L_{j-1}, Q_j^{(e)});$$

$$R_j = L_{j-1}, \ (j = 1, 2, \ldots, 12).$$

После 12-го раунда над блоком $X = (R_{12}, L_{12})$ выполняется заключительное преобразование ***FT*** в соответствии с формулой:

$$Y = \boldsymbol{FT}(X, Q_{FT}^{(e)}).$$

Начальное преобразование *IT*

Начальное преобразование ***IT*** имеет следующий вид:

$$Y = \boldsymbol{IT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^{64}$, $A \in \mathrm{GF}(2)^{32}$.

Схема реализации данного преобразования приведена на рис. 5.6, при этом каждая пара битов входного блока X с индексами $2j–1$ и $2j$ ($j = 1, 2, ..., 32$) либо переставляется ($a_j = 1$), либо не переставляется ($a_j = 0$), после чего каждый четный бит инвертируется. Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j} \oplus 1.$$

Процедура *Crypt*

Формальная запись процедуры ***Crypt*** имеет вид:

$$R = \boldsymbol{Crypt}(R, L, (A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)}, A^{(5)}, A^{(6)})),$$

где $L, R \in \mathrm{GF}(2)^{32}$ и $A^{(1)}, A^{(2)}, \ldots, A^{(6)} \in \mathrm{GF}(2)^{32}$.

Элементы $A^{(1)}, A^{(2)}, \ldots, A^{(6)}$ являются формальными параметрами ключевой информации, т. е. параметру $A^{(i)}$ соответствует $Q_j^{(i,e)}$ — элемент раундового ключа $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, ..., Q_j^{(6,e)})$. Соответственно, R и L — формальные параметры для левой и правой частей блока входных данных.

Особенностью данной реализации процедуры ***Crypt*** является использование в одном раунде шести подключей длины 32.

В процедуре применяются:

- *циклический сдвиг* ">>>" подблока данных на фиксированное значение в сторону младших разрядов;
- *побитовое сложение* "⊕";
- *нелинейная векторная булева функция* **G**;
- *блоки управляемых перестановок* $\mathbf{P}_{32/80}$ и $\mathbf{P}_{32/80}^{-1}$ равновероятного смещения первого порядка;
- *процедура расширения управляющего вектора* **E** (блок расширения).

Рис. 5.10. Блок управляемых перестановок $\mathbf{P}_{32/80}$

В качестве блоков $\mathbf{P}_{32/80}$ и $\mathbf{P}_{32/80}^{-1}$ используются двухуровневые сети Клоса $\mathbf{C}_{<8,\,4>}$ и $\mathbf{C}_{<4,\,8>}$ соответственно, причем для блока $\mathbf{C}_{<8,\,4>}$ блоками меньшего размера являются блоки $\mathbf{R}_3$ и $\mathbf{R}_2$, а для $\mathbf{C}_{<4,8>} = \mathbf{C}_{<4,8>}^{-1}$ — блоки $\mathbf{R}_2^{-1}$ и $\mathbf{R}_3^{-1}$. По сравнению с классической структурой сети Клоса $\mathbf{C}_{<8,\,4>}$ используется измененный порядок следования блоков $\mathbf{R}_2$ (см. нумерацию блоков на рис. 5.10). Коммутация между третьим и четвертым слоями в блоке $\mathbf{P}_{32/80}$ имеет вид инволюции:

$$\pi = (1)(2,9)(3,17)(4,25)(5)(6,13)(7,21)(8,29)(10)(11,18)(12,26)$$
$$(14)(15,22)(16,30)(19)(20,27)(23)(24,31)(28)(32)$$

Последовательность выполняемых преобразований и их взаимодействие приведены на рис. 5.11.

Временная ось выполнения операций в процедуре ***Crypt*** представлена в табл. 5.2.

Таблица 5.2. *Временная ось выполнения операций в процедуре* ***Crypt***

№№ пп	Временная ось ($\tau_{\oplus}$)	Операции
1	0	$U = L^{>>>11}$ $U' = L^{>>>17}$ $U'' = L^{>>>11}$
2	0 ÷ 1	$V = \mathbf{E}_{A^{(1)}\|\|A^{(2)}}(U)$ $V' = \mathbf{E}_{A^{(3)}\|\|A^{(4)}}(U')$ $V'' = \mathbf{E}_{A^{(5)}\|\|A^{(6)}}(U'')$ $W = L \oplus A^{(3)}$
3	0 ÷ 5	$R = \mathbf{P}_{32/80(V)}(R)$ $X = \mathbf{P}_{32/80(V')}(A^{(4)})$
4	1 ÷ 5	$X = \mathbf{G}_{A^{(3)}\|\|A^{(4)}}(W)$
5	5 ÷ 6	$R = R \oplus X$
6	6 ÷ 7	$R = R \oplus X'$
7	7 ÷ 12	$R = \mathbf{P}_{32/80(V'')}^{-1}(R)$

Как следует из табл. 5.2, большинство операций выполняются параллельно, а общее время, необходимое для выполнения процедуры, равно последовательному выполнению двенадцати операций типа XOR.

Замечание

Этот показатель можно улучшить, если использовать матричную структуру реализации блоков $\mathbf{R}_2$ в блоках $\mathbf{P}_{32/80}$ и $\mathbf{P}^{-1}_{32/80}$. Вопросы матричной реализации рассматривались в *гл. 3*. В этом случае время выполнения процедуры ***Crypt*** не превысит время, равное последовательному выполнению девяти операций типа XOR.

Рис. 5.11. Схема процедуры ***Crypt***

Векторная булева функция G

В качестве векторной булевой функции **G** используется управляемая подстановочная операция, являющаяся одной из реализаций последовательной биективной модели и формируемая в соответствии с правилами, представленными в *разд. 4.3.3*, *4.3.7—4.3.10* и *4.5.6*. Формальная запись функции **G** имеет вид:

$$X' = \mathbf{G}(W, A, B), \text{ где } X', W, A, B \in \mathrm{GF}(2)^{32}.$$

Формальные параметры A и B соответствуют формальным параметрам $A^{(3)}$ и $A^{(4)}$ процедуры ***Crypt***. В векторном виде функция **G** определяется формулой:

$$X' = M^{(0)} \oplus M^{(1)} \oplus (M^{(2)} \otimes A) \oplus (M^{(2)} \otimes M^{(5)} \otimes B) \oplus (M^{(3)} \otimes M^{(5)}) \oplus (M^{(4)} \otimes B).$$

Двоичные векторы $M^{(0)}$, $M^{(1)}$, …, $M^{(5)}$ выражаются рекурсивно через W, а именно:

$$M^{(0)} = (m_1^{(0)}, m_2^{(0)}, m_3^{(0)}, ..., m_{31}^{(0)}, m_{32}^{(0)}) = (w_1, w_2, w_3, ..., w_{31}, w_{32})$$

и $\forall\, j = 1, 2, \ldots, 5$

$$M^{(j)} = (m_1^{(j)}, m_2^{(j)}, m_3^{(j)}, ..., m_{31}^{(j)}, m_{32}^{(j)}) = (1, m_1^{(j-1)}, m_2^{(j-1)}, ..., m_{31}^{(j)}).$$

Учитывая возможность параллельного выполнения некоторых операций, легко установить, что функция **G** выполняется за четыре такта задержки ($4\tau_{\oplus}$) вместо допустимых пяти тактов, т. е. существует резерв для ее усложнения *(см. разд. 4.5.2)*.

Обоснование выбора нелинейной функции G

Определим отображение **G**: $\mathrm{GF}(2)^{32} \times \mathrm{GF}(2)^{32} \times \mathrm{GF}(2)^{32} \to \mathrm{GF}(2)^{32}$ в виде векторной булевой функции $Y = \mathbf{G}(X, A, B)$ следующего вида:

$$(y_1, y_2, \ldots, y_{32}) = (\mathbf{g}_1(X, A, B), \mathbf{g}_2(X, A, B), \ldots, \mathbf{g}_{32}(X, A, B)),$$

где $\mathbf{g}_i$ — некоторые образующие булевы функции.

Если для каждого i выполняется условие $\mathbf{g}_i = \mathbf{g}_i(x_1, x_2, \ldots, x_i, A, B)$, то такое отображение является последовательной моделью управляемой подстановочной операции. Если в качестве образующих функций в указанной модели выбрать функции $\mathbf{g}_i = x_i \oplus \mathbf{g}'_i(x_1, x_2, ..., x_{i-1}, A, B)$, то векторная булева функция **G** реализует биективное отображение по X *(см. разд. 4.3.3)*. В качестве *прототипа образующих функций (см. разд. 4.3.7)* в функции **G** используется единая унифицированная булева функция от восьми переменных, имеющая следующий вид:

$$\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_8) = z_7 \oplus z_8 \oplus \varphi(z_1, z_2, \ldots, z_6),$$

где $\varphi(z_1, z_2, \ldots, z_6) = z_1z_2z_3 \oplus z_1z_4 \oplus z_2z_5 \oplus z_3z_6$ является бент-функцией *(см. разд. 4.2.8)*.

Преобразование Уолша-Адамара *(см. разд. 4.2.3)* функции **g** состоит из 64 элементов, принимающих значение $|\mathbf{U}^*_\alpha(g)| = 32$ для всех векторов $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_6, 0, 0)$, а для остальных 192-х элементов — значение $\mathbf{U}^*_\alpha(g) = 0$. С учетом выбранной структуры и свойств бент-функций имеем следующие характеристики функции **g**:

- нелинейность функции **g** равна $\mathbf{N}(\mathbf{g}) = 2^{m-1} - 2^{((m+2)/2)-1} = 112$ (при $m = 8$), что достаточно близко к максимально возможной нелинейности булевых функций от восьми переменных ($\mathbf{N}_{\max} = 2^{m-1} - 2^{(m/2)-1} = 120$) *(см. разд. 4.2.4)*;
- **g** — корреляционно-иммунная функция на множестве векторов $\{(z_1, z_2, \ldots, z_8)\} \subset \mathrm{GF}(2)^8$, где $z_7 \neq 0$ или $z_8 \neq 0$, что составляет 75% от всего множества $\mathrm{GF}(2)^8$ *(см. разд. 4.2.7)*;

- **g** обладает хорошими автокорреляционными свойствами, т. к. для всех ненулевых лавинных векторов $(\Delta z_1, \Delta z_2, \ldots, \Delta z_8) \in \mathrm{GF}(2)^8$, кроме трех ((0, …, 0, 0, 1), (0, …, 0, 1, 0) и (0, …, 0, 1, 1)), выполняется *критерий распространения 8-го порядка*, т. е. функция $\Delta\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_8) \oplus \mathbf{g}(z_1 \oplus \Delta z_1, z_2 \oplus \Delta z_2, \ldots, z_8 \oplus \Delta z_8)$ является сбалансированной *(см. разд. 4.2.6)*;
- степень алгебраической нормальной формы функции **g** равна трем (**deg**(**g**) = 3) *(см. разд. 4.2.2)*.

Переход от общего вида функции g к конкретным образующим функциям последовательной модели управляемой подстановочной операции осуществляется по следующей подстановке переменных:

$$\begin{pmatrix} z_1 & z_2 & z_3 & z_4 & z_5 & z_6 & z_7 & z_8 \\ x_{i-2} & x_{i-5} & b_i & a_i & x_{i-3} & x_{i-4} & x_{i-1} & x_i \end{pmatrix},$$

где $\forall i \in \{1, 2, \ldots, n\}$ и $n = 32$. *Подробную информацию см. в разд. 4.5.6*.

Таким образом, образующие функции имеют следующий вид:

$$y_i = \mathbf{g}_i(X, A, B) = x_i \oplus x_{i-1} \oplus x_{i-2}a_i \oplus x_{i-2}x_{i-5}\, b_i \oplus x_{i-3}x_{i-5} \oplus x_{i-4}b_i,$$

где x_i, a_i, b_i — компоненты, соответственно, векторов $X, A, B \in \mathrm{GF}(2)^{32}$, а начальные условия соответствуют вектору $(x_{-4}, x_{-3}, x_{-2}, x_{-1}, x_0) = (1, 1, 1, 1, 1)$.

В соответствии со схемой процедуры ***Crypt*** (рис. 5.11) функции **G** предшествует линейное преобразование $X = L \oplus A$, не изменяющее исходные значения нелинейности и степени алгебраической нормальной формы образующих функций (см. утверждение 4.16), однако усложняющее результирующий вид образующих функций. При этом функция $\mathbf{G}' = \mathbf{G}(L \oplus A)$ биективна по L при различных фиксированных значениях параметров A и B.

Процедура расширения управляющего вектора E (блок расширения)

Процедура предназначена для формирования 80-разрядного управляющего вектора из 32-разрядного, формальная запись которого имеет следующий вид:

$$V = (V_1 \parallel V_2 \parallel V_3 \parallel V_4 \parallel V_5) = \mathbf{E}(U, A, B) = \mathbf{E}_{A\|B}(U),$$

где $V \in \mathrm{GF}(2)^{80}$, $V_1, V_2, V_3, V_4, V_5 \in \mathrm{GF}(2)^{16}$, $U, A, B \in \mathrm{GF}(2)^{32}$.

Фактически вектор V вычисляется в соответствии с формулами:

$$V_1 = U_{hi}; \quad V_2 = \mathbf{P}_{\pi(1)}((U \oplus A)_{hi}); \quad V_5 = \mathbf{P}_{\pi(1)}((U \oplus A)_{lo}),$$

$$V_3 = \mathbf{P}_{\pi(5)}((U \oplus B)_{hi}); \quad V_4 = \mathbf{P}_{\pi(5)}((U \oplus B)_{lo}),$$

где преобразование $\mathbf{P}_{\pi(s)}$ имеет вид: $\mathbf{P}_{\pi(s)}(Z) = (Z_{hi})^{>>>s} \parallel (Z_{lo})^{>>>s}$.

Для блока $\mathbf{P}_{32/80}$ в табл. 5.3 указано соответствие номеров разрядов векторов V и U. Так, например, 17 в первой строке означает, что на вход $v_1^{(1)}$ управления блоком $\mathbf{P}_{2/1}$ поступает значение u_{17}, 22_1 соответствует значению $v_{13}^{(2)} = u_{22} \oplus a_{22}$, а 7_2 — значению $v_{10}^{(4)} = u_7 \oplus b_7$. Процедура расширения **E** составлена таким образом, что 16 младших разрядов вектора U участвуют в управлении только блоками $\mathbf{R}_2$, а старшие — только блоками $\mathbf{R}_3$, причем в каждом блоке $\mathbf{R}_2$ и $\mathbf{R}_3$ все номера разрядов вектора U разные.

Таблица 5.3. *Таблица значений управляющих разрядов для блока* $\boldsymbol{P}_{32/80}$

V_1	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	U_{hi}
V_2	26_1	27_1	28_1	29_1	30_1	31_1	32_1	25_1	18_1	19_1	20_1	21_1	22_1	23_1	24_1	17_1	$(U \oplus A)_{hi}$ и $(U \oplus B)_{hi}$
V_3	30_2	31_2	32_2	25_2	26_2	27_2	28_2	29_2	22_2	23_2	24_2	17_2	18_2	19_2	20_2	21_2	
V_4	14_2	15_2	16_2	9_2	10_2	11_2	12_2	13_2	6_2	7_2	8_2	1_2	2_2	3_2	4_2	5_2	$(U \oplus B)_{lo}$ и $(U \oplus A)_{lo}$
V_5	10_1	11_1	12_1	13_1	14_1	15_1	16_1	9_1	2_1	3_1	4_1	5_1	6_1	7_1	8_1	1_1	

Для блока $\mathbf{P}_{32/80}^{-1}$ используется та же самая процедура расширения **E**, однако с учетом особой нумерации разрядов управления блоком $\mathbf{P}_{32/80}^{-1}$ табл. 5.3 преобразуется к виду, представленному в табл. 5.4.

Таблица 5.4. *Таблица значений управляющих разрядов для блока* $\boldsymbol{P}_{32/80}^{-1}$

V_5	10_1	11_1	12_1	13_1	14_1	15_1	16_1	9_1	2_1	3_1	4_1	5_1	6_1	7_1	8_1	1_1	$(U \oplus A)_{lo}$ и $(U \oplus B)_{lo}$
V_4	14_2	15_2	16_2	9_2	10_2	11_2	12_2	13_2	6_2	7_2	8_2	1_2	2_2	3_2	4_2	5_2	
V_3	30_2	31_2	32_2	25_2	26_2	27_2	28_2	29_2	22_2	23_2	24_2	17_2	18_2	19_2	20_2	21_2	$(U \oplus B)_{hi}$ и $(U \oplus A)_{hi}$
V_2	26_1	27_1	28_1	29_1	30_1	31_1	32_1	25_1	18_1	19_1	20_1	21_1	22_1	23_1	24_1	17_1	
V_1	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	U_{hi}

Процедура **E** обеспечивает:

- *во-первых*, равномерность участия всех 32 разрядов вектора U в управлении блоком $\mathbf{P}_{32/80}$;
- *во-вторых*, для произвольно выбранной коммутации одного разряда входа с одним разрядом выхода в управлении участвуют пять (по числу слоев) разных разрядов вектора U, что гарантирует возможность такой коммутации при любых значениях раундовых ключей A и B, и, следовательно, соответствует определению блока первого порядка;

- *в-третьих*, управляющий вектор V_1 является фиксированной перестановкой 16 разрядов блока L, т. е. преобразование в первом слое блока $\mathbf{P}_{32/80}$ выполняется без какой-либо дополнительной временной задержки.

Заключительное преобразование *FT*

Процедура ***FT*** является обратным преобразованием по отношению к процедуре ***IT***. Преобразование имеет следующий вид:

$$Y = \boldsymbol{FT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^{64}$, $A \in \mathrm{GF}(2)^{32}$.

Первоначально каждый четный бит входного блока инвертируется (см. рис. 5.7), а затем каждая пара битов входного блока X с индексами $2j$–1 и $2j$ (j = 1, 2, …, 32) либо переставляется (a_j = 1), либо не переставляется (a_j = 0).

Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j} \oplus 1.$$

5.2.3. Расписание использования раундовых ключей

В 12-раундовом блочном шифре SPECTR-H64 расширенный ключ $Q^{(e)}$ является объединением 14 подключей, а именно:

$$Q^{(e)} = (Q_{IT}^{(e)}, Q_1^{(e)}, Q_2^{(e)}, ..., Q_{12}^{(e)}, Q_{FT}^{(e)}),$$

где $\forall j = 1, 2, \ldots, 12 \quad Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, ..., Q_j^{(6,e)})$,

причем $Q_{IT}^{(e)}, Q_{FT}^{(e)}, Q_j^{(h,e)} \in \mathrm{GF}(2)^{32} \quad \forall h = 1,\ 2,\ ...,\ 6.$

Расширенный ключ $Q^{(e)}$ является последовательностью семидесяти четырех 32-разрядных двоичных векторов, каждый из которых является одним из восьми секретных подключей $K_1, K_2, \ldots, K_8$.

Часть элементов расширенного ключа $Q^{(e)}$ имеют фиксированные значения K_j, а остальные определяются через параметры O_i, которые, в свою очередь, вычисляются по формулам:

$$O_{2i-1} = K_{2i-1+e} \text{ и } O_{2i} = K_{2i-e},$$

где $K = K_1 \parallel K_2 \parallel \ldots \parallel K_8$ и i = 1, 2, 3, 4.

Схема коммутации между O_i и K_j представлена на рис. 5.12.

Соответствие подключей $Q_{IT}^{(e)}, Q_{FT}^{(e)}, Q_j^{(1,e)}, Q_j^{(2,e)}, ..., Q_j^{(6,e)}$ (j = 1, 2, …, 12) задается табл. 5.5.

Рис. 5.12. Схема коммутации подключей

Таблица 5.5. *Расписание использования раундовых ключей*

$Q_{IT}^{(e)}$	j =	1	2	3	4	5	6	7	8	9	10	11	12	$Q_{FT}^{(e)}$
O_1	$Q_j^{(1,e)}$ =	K_1	K_8	K_5	K_4	K_1	K_6	K_7	K_4	K_2	K_6	K_5	K_3	O_2
	$Q_j^{(2,e)}$ =	K_2	K_6	K_7	K_3	K_2	K_8	K_5	K_3	K_1	K_8	K_7	K_4	
	$Q_j^{(3,e)}$ =	O_6	O_1	O_2	O_5	O_7	O_3	O_4	O_8	O_6	O_1	O_2	O_5	
	$Q_j^{(4,e)}$ =	O_7	O_4	O_3	O_8	O_6	O_1	O_2	O_5	O_7	O_4	O_3	O_8	
	$Q_j^{(5,e)}$ =	K_3	K_5	K_6	K_2	K_4	K_7	K_6	K_1	K_4	K_5	K_8	K_1	
	$Q_j^{(6,e)}$ =	K_4	K_7	K_8	K_1	K_3	K_5	K_8	K_2	K_3	K_7	K_6	K_2	

Так, например, для пятого раунда ($j = 5$) формальным параметрам $A^{(1)}, A^{(2)}, \ldots, A^{(6)}$ процедуры ***Crypt*** соответствует последовательность подключей $K_1 \parallel K_2 \parallel O_7 \parallel O_6 \parallel K_4 \parallel K_3$, а именно:

$$Q_5^{(e)} = (Q_5^{(1,e)}, Q_5^{(2,e)}, Q_5^{(3,e)}, Q_5^{(4,e)}, Q_5^{(5,e)}, Q_5^{(6,e)}) = (K_1, K_2, O_7, O_6, K_4, K_3).$$

То есть $Q_5^{(e)} = (K_1, K_2, K_{7+e}, K_{6-e}, K_4, K_3)$.

Для режима зашифрования $Q_5^{(0)} = (K_1, K_2, K_7, K_6, K_4, K_3)$,

а для режима расшифрования $Q_5^{(e)} = (K_1, K_2, K_8, K_5, K_4, K_3)$.

Для начального и заключительного преобразований ключи $Q_{IT}^{(e)}$ и $Q_{FT}^{(e)}$ имеют вид:

$$Q_{IT}^{(e)} = O_1 \text{ и } Q_{FT}^{(e)} = O_2.$$

То есть

$$Q_{IT}^{(e)} = K_{1+e} \text{ и } Q_{FT}^{(e)} = K_{2-e}.$$

Для режима зашифрования

$$Q_{IT}^{(0)} = K_1 \text{ и } Q_{FT}^{(0)} = K_2,$$

а для режима расшифрования

$$Q_{IT}^{(1)} = K_2 \text{ и } Q_{FT}^{(1)} = K_1.$$

5.3. Шифр (алгоритм) SPECTR-128

Алгоритм является практической реализацией основных идей синтеза скоростного блочного шифра на основе управляемых операций, рассматривавшихся в предыдущих главах, и ориентирован на скоростную обработку информации с 64-разрядной шиной обмена данными.

5.3.1. Общая схема блочного шифрования

Для алгоритма SPECTR-128 общая схема шифрования (зашифрование и расшифрование) определяется формулой:

$$Y = \mathbf{F}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от 256-битового секретного ключа K и режима шифрования e ($e = 0$ — зашифрование, $e = 1$ — расшифрование), причем в режиме зашифрования X является исходным блоком двоичных данных (открытым текстом), а в режиме расшифрования — преобразованным блоком двоичных данных (шифртекстом). В режиме зашифрования результирующее значение Y является шифртекстом, а в режиме расшифрования — открытым текстом.

Секретный ключ K представлен в виде объединения четырех 64-разрядных подключей, а именно: $K = K_1 \parallel K_2 \parallel K_3 \parallel K_4$, где и $K_1, K_2, K_3, K_4 \in \mathrm{GF}(2)^{64}$.

Описание алгоритма шифрования (функции **F**) приведено в *разд. 5.3.2*, а процедура формирования рабочего ключа $Q^{(e)} = \mathbf{H}(K, e)$ в *разд. 5.3.3*.

5.3.2. Алгоритм шифрования

Функция **F** (см. рис. 5.5) реализована в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. 12 раундов (циклов) *преобразований* с использованием процедуры ***Crypt***.
3. Заключительного преобразования ***FT***.

Первоначально выполняется процедура ***IT***: $Y = \boldsymbol{IT}(X, Q_{IT}^{(e)})$.

Блок Y делится на два подблока L_0 и R_0 одинаковой длины, т. е. $(L_0, R_0) = Y$, где $L_0, R_0 \in \mathrm{GF}(2)^{64}$. Затем выполняются 12 раундов преобразований с использованием процедуры ***Crypt*** в соответствии с формулами:

$$L_j = \boldsymbol{Crypt}(R_{j-1}, L_{j-1}, Q_j^{(e)});$$

$$R_j = L_{j-1}, \ (j = 1, 2, \ldots, 12).$$

После 12-го раунда над блоком $X = (R_{12}, L_{12})$ выполняется заключительное преобразование ***FT*** в соответствии с формулой:

$$Y = \boldsymbol{FT}(X, Q_{FT}^{(e)}).$$

Начальное преобразование *IT*

Начальное преобразование ***IT*** имеет следующий вид:

$$Y = \boldsymbol{IT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^{128}$, $A \in \mathrm{GF}(2)^{64}$.

Схема реализации данного преобразования приведена на рис. 5.6, при этом каждая пара битов входного блока X с индексами $2j-1$ и $2j$ ($j = 1, 2, ..., 64$) либо переставляется ($a_j = 1$), либо не переставляется ($a_j = 0$), после чего каждый четный бит инвертируется. Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j})a_j \oplus x_{2j} \oplus 1.$$

Процедура *Crypt*

Формальная запись процедуры ***Crypt*** имеет вид:

$$R = \boldsymbol{Crypt}(R, L, (A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)})),$$

где $L, R \in \mathrm{GF}(2)^{64}$ и $A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)} \in \mathrm{GF}(2)^{64}$.

Элементы $A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)}$ являются формальными параметрами, ключевой информации, т. е. параметру $A^{(i)}$, соответствует $Q_j^{(i,e)}$ — элемент раундового ключа $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)})$. Соответственно, R и L — формальные параметры для левой и правой частей блока входных данных.

Особенностью данной реализации процедуры ***Crypt*** является использование в одном раунде четырех подключей длины 64, т. е. участвуют все биты 256-разрядного секретного ключа K. Кроме того, применяется более сложная векторная булева функция **G**.

Рис. 5.13. Блок управляемых перестановок $\mathbf{P}_{64/192}$

В процедуре используются:

- *циклический сдвиг* ">>>" подблока данных на фиксированное значение в сторону младших разрядов;
- *побитовое сложение* "⊕";
- *нелинейная векторная булева функция* **G**;
- *блоки управляемых перестановок* $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ равновероятного смещения первого порядка;
- *процедура расширения управляющего вектора* **E** (блок расширения).

В качестве блоков $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ используется двухуровневая сеть Клоса $\mathbf{C}_{<8,\,8>}$ (рис. 5.13), причем первый уровень составляют блоки $\mathbf{R}_3$, а второй — блоки $\mathbf{R}_3^{-1}$, структура которых приведена в *гл. 3* (см. рис. 3.22). Поскольку структура блоков $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ совпадает, на рис. 5.13 колонка I соответствует управляющему вектору для прямого преобразования, а колонка II — для обратного преобразования.

Последовательность выполняемых преобразований и их взаимодействие представлены на рис. 5.14.

Временная ось выполнения операций в процедуре ***Crypt*** представлена в табл. 5.6.

Как следует из табл. 5.6, большинство операций выполняется параллельно, а общее время, необходимое для выполнения процедуры, равно последовательному выполнению четырнадцати операций типа XOR.

Рис. 5.14. Схема процедуры ***Crypt***

Таблица 5.6. *Временная ось выполнения операций в процедуре* ***Crypt***

№№ пп	Временная ось ($\tau_{\oplus}$)	Операции
1	0	$U = L^{>>>21}$ $U' = L^{>>>43}$ $U'' = L^{>>>21}$
2	0 ÷ 1	$V = \mathbf{E}_{A^{(1)}}(U)$ $V' = \mathbf{E}_{A^{(3)}}(U')$ $V'' = \mathbf{E}_{A^{(2)}}(U'')$
3	0 ÷ 6	$R = \mathbf{P}_{64/192(V)}(R)$ $X = \mathbf{P}_{64/192(V')}(A^{(4)})$
4	0 ÷ 5	$X' = \mathbf{G}_{A^{(3)}\|\|A^{(4)}}(L)$
5	6 ÷ 7	$R = R \oplus X$
6	7 ÷ 8	$R = R \oplus X'$
7	8 ÷ 14	$R = \mathbf{P}^{-1}_{64/192(V'')}(R)$

Векторная булева функция G

В качестве векторной булевой функции **G** используется управляемая подстановочная операция, являющаяся одной из реализаций последовательной биективной модели и формируемая в соответствии с правилами, представленными в *разд. 4.3.3*, *4.3.7—4.3.10* и *4.5.6*. Общая запись функции **G** имеет вид:

$$X' = \mathbf{G}(L, A, B), \text{ где } X', L, A, B \in \mathrm{GF}(2)^{64}.$$

Формальные параметры A и B соответствуют формальным параметрам $A^{(3)}$ и $A^{(4)}$ процедуры ***Crypt***.

В векторном виде функция **G** определяется формулой

$$X' = L^{(0)} \oplus A^{(0)} \oplus (L^{(1)} \otimes B^{(2)}) \oplus (L^{(6)} \otimes L^{(8)}) \oplus (A^{(2)} \otimes L^{(7)}) \oplus$$
$$\oplus (A^{(1)} \otimes B^{(1)}) \oplus (L^{(3)} \otimes L^{(9)}) \oplus (L^{(1)} \otimes L^{(9)} \otimes A^{(2)}) \oplus (L^{(1)} \otimes L^{(6)} \otimes L^{(9)} \otimes B^{(1)}),$$

в которой векторы L_j ($j = 0, 1, 2, \ldots, 9$) выражаются рекурсивно через L, а именно:

$$L^{(0)} = (l_1^{(0)}, l_2^{(0)}, l_3^{(0)}, \ldots, l_{63}^{(0)}, l_{64}^{(0)}) = L = (l_1, l_2, l_3, \ldots, l_{63}, l_{64}) \text{ и } \forall j = 0, 1, \ldots, 5$$

и

$$L^{(j)} = (l_1^{(j)}, l_2^{(j)}, l_3^{(j)}, \ldots, l_{63}^{(j)}, l_{64}^{(j)}) = (1, l_1^{(j-1)}, l_2^{(j-1)}, l_3^{(j-1)}, \ldots, l_{63}^{(j-1)}).$$

Аналогично определяются векторы $A^{(j)}$ и $B^{(j)}$ для $j = 0, 1, 2$.

Учитывая возможность параллельного выполнения некоторых операций, легко установить, что функция **G** выполняется за пять тактов задержки ($5\tau_{\oplus}$) *(см. разд. 4.5.2)*.

Обоснование выбора нелинейной функции G

Определим отображение **G**: $\mathrm{GF}(2)^{64} \times \mathrm{GF}(2)^{64} \times \mathrm{GF}(2)^{64} \to \mathrm{GF}(2)^{64}$ в виде векторной булевой функции $Y = \mathbf{G}(X, A, B)$ следующего вида:

$$(y_1, y_2, \ldots, y_{64}) = (\mathbf{g}_1(X, A, B), \mathbf{g}_2(X, A, B), \ldots, \mathbf{g}_{64}(X, A, B)),$$

где $\mathbf{g}_i$ — некоторые образующие булевы функции. Если для каждого i выполняется условие $\mathbf{g}_i = \mathbf{g}_i(x_1, x_2, \ldots, x_i, A, B)$, то такое отображение является последовательной моделью управляемой подстановочной операции. Если в качестве образующих функций в указанной модели выбрать функции $\mathbf{g}_i = x_i \oplus \mathbf{g}'_i(x_1, x_2, \ldots, x_{i-1}, A, B)$, то векторная булева функция **G** реализует биективное отображение по X *(см. разд. 4.3.3)*. В качестве *прототипа образующих функций (см. разд. 4.3.7)* в функции **G** используется единая унифицированная булева функция от двенадцати переменных, имеющая следующий вид:

$$\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_{12}) = z_{11} \oplus z_{12} \oplus \varphi(z_1, z_2, \ldots, z_{10}),$$

где $\varphi(z_1, z_2, \ldots, z_{10}) = z_1z_2 \oplus z_3z_4 \oplus z_5z_6 \oplus z_7z_8 \oplus z_9z_{10} \oplus z_1z_5z_9 \oplus z_1z_3z_7z_9$ является бент-функцией *(см. разд. 4.2.8)*. Преобразование Уолша-Адамара *(см. разд. 4.2.3)* функции **g** состоит из 1024-х элементов, принимающих значение $|\mathbf{U}_{\alpha}^{\#}(\mathbf{g})| = 128$ для всех векторов $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{10}, 0, 0)$, а для остальных 3072-х элементов — значение $\mathbf{U}_{\alpha}^{*}(g) = 0$. С учетом выбранной структуры и свойств бент-функций, имеем следующие характеристики функции **g**:

- нелинейность функции **g** равна $\mathbf{N}(\mathbf{g}) = 2^{m-1} - 2^{((m+2)/2)-1} = 1984$ (при $m = 12$), что достаточно близко к максимально возможной нелинейности булевой функции от двенадцати переменных ($\mathbf{N}_{\max} = 2^{m-1} - 2^{(m/2)-1} = 2016$) *(см. разд. 4.2.4)*;

- **g** — корреляционно-иммунная булева функция на множестве векторов $\{(z_1, z_2, \ldots, z_{12})\} \subset \mathrm{GF}(2)^{12}$, где $z_{11} \neq 0$ или $z_{12} \neq 0$ *(см. разд. 4.2.7)*, что составляет 75% от всего множества $\mathrm{GF}(2)^{12}$;
- **g** обладает хорошими автокорреляционными свойствами, т. к. для всех ненулевых лавинных векторов $(\Delta z_1, \Delta z_2, \ldots, \Delta z_{12}) \in \mathrm{GF}(2)^{12}$, кроме трех ((0, …, 0, 0, 1), (0, …, 0, 1, 0) и (0, …, 0, 1, 1)), выполняется *критерий распространения 12-го порядка*, т. е. функция $\Delta\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_{12}) \oplus \oplus\, \mathbf{g}(z_1 \oplus \Delta z_1, \ldots, z_{12} \oplus \Delta z_{12})$ является сбалансированной *(см. разд. 4.2.6)*;
- степень алгебраической нормальной формы функции **g** равна четырем ($\mathbf{deg}(\mathbf{g}) = 4$) *(см. разд. 4.2.2)*.

Переход от общего вида функции **g** к конкретным образующим функциям последовательной модели управляемой подстановочной операции осуществляется по следующей подстановке переменных *(подробную информацию см. в разд. 4.5.6)*:

$$\begin{pmatrix} z_1 & z_2 & z_3 & z_4 & z_5 & z_6 & z_7 & z_8 & z_9 & z_{10} & z_{11} & z_{12} \\ l_{i-1} & b_{i-2} & l_{i-6} & l_{i-8} & a_{i-2} & l_{i-7} & b_{i-1} & a_{i-1} & l_{i-9} & l_{i-3} & l_i & a_i \end{pmatrix},$$

где $\forall i \in \{1, \ldots, n\}$ и $n = 64$.

Таким образом, имеем множество образующих функций вида:

$$y_i = \mathbf{g}_i = l_i \oplus a_i \oplus l_{i-1}b_{i-2} \oplus l_{i-6}l_{i-8} \oplus a_{i-2}l_{i-7} \oplus a_{i-1}b_{i-1} \oplus l_{i-3}l_{i-9} \oplus l_{i-3}a_{i-2}l_{i-9} \oplus \\ \oplus\, l_{i-1}b_{i-1}l_{i-6}l_{i-9},$$

где l_j, a_j, b_j — компоненты, соответственно, векторов $L, A, B \in \mathrm{GF}(2)^{64}$, а начальные условия соответствуют вектору $(x_{-4}, x_{-3}, x_{-2}, x_{-1}, x_0) = (1, 1, 1, 1, 1)$.

Процедура расширения управляющего вектора E (блок расширения)

Процедура предназначена для формирования 192-разрядного управляющего вектора из 64-разрядного. Формальная запись процедуры **E** имеет следующий вид:

$$V = (V_1 \parallel V_2 \parallel V_3 \parallel V_4 \parallel V_5 \parallel V_6) = \mathbf{E}(U, A) = \mathbf{E}_A(U),$$

где $V \in \mathrm{GF}(2)^{192}$, $V_1, V_2, V_3, V_4, V_5, V_6 \in \mathrm{GF}(2)^{32}$, $U, A \in \mathrm{GF}(2)^{64}$.

Фактически вектор V вычисляется в соответствии с формулами:

$$V_1 = U_{hi}; \qquad V_2 = \mathbf{P}_{\pi(1)}((U \oplus A)_{hi}); \quad V_3 = \mathbf{P}_{\pi(11)}(U_{hi}) \oplus \mathbf{P}_{\pi(8)}(A_{hi});$$
$$V_6 = U_{lo} \oplus (A_{lo})^{>>>28}; \quad V_5 = \mathbf{P}_{\pi(1)}((U \oplus A)_{lo}); \quad V_4 = \mathbf{P}_{\pi(11)}(U_{lo}) \oplus \mathbf{P}_{\pi(8)}(A_{lo});$$

где преобразование $\mathbf{P}_{\pi(s)}$ имеет вид: $\mathbf{P}_{\pi(s)}(Z) = (Z_{hi})^{>>>s} \parallel (Z_{lo})^{>>>s}$.

Для блока $\mathbf{P}_{64/192}$ в табл. 5.7 указано соответствие номеров разрядов векторов V и U. Так, например, 19 в четвертой строке, соответствующей вектору V_4, означает,

что на вход $v_{10}^{(4)}$ управления блоком $\mathbf{P}_{2/1}$ поступает значение u_{19}, 22 соответствует значению u_{22}, а 7 — значению u_7.

Таблица 5.7. *Таблица соответствия между разрядами векторов U и V для блока* $\mathbf{P}_{64/192}$

V_1	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
V_2	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	49	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	33
V_3	58	59	60	61	62	63	64	49	50	51	52	53	54	55	56	57	42	43	44	45	46	47	48	33	34	35	36	37	38	39	40	41
V_4	26	27	28	29	30	31	32	17	18	19	20	21	22	23	24	25	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8	9
V_5	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	17	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1
V_6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Аналогично, в табл. 5.8 указано соответствие номеров разрядов векторов V и A, причем из анализа таблицы следует, что V_1 не зависит от A.

Таблица 5.8. *Таблица соответствия между разрядами векторов A и V для блока* $\mathbf{P}_{64/192}$

V_1																																
V_2	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	49	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	33
V_3	57	58	59	60	61	62	63	64	49	50	51	52	53	54	55	56	41	42	43	44	45	46	47	48	33	34	35	36	37	38	39	40
V_4	25	26	27	28	29	30	31	32	17	18	19	20	21	22	23	24	9	10	11	12	13	14	15	16	1	2	3	4	5	6	7	8
V_5	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	17	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	1
V_6	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32

Расширение составлено таким образом, что 32 старших разряда вектора A участвуют в управлении только блоками $\mathbf{R}_3$, а младшие — только блоками $\mathbf{R}_3^{-1}$, причем в каждом блоке $\mathbf{R}_3$ и $\mathbf{R}_3^{-1}$ все номера разрядов векторов U и A разные. Более того, по построению не существует двух блоков $\mathbf{P}_{2/1}$, для которых управляющие разряды имели бы одинаковое выражение типа $u_i \oplus a_j$.

Процедура расширения управляющего вектора обеспечивает:

- равномерность участия всех 64 разрядов вектора U в управлении блоком $\mathbf{P}_{64/128}$;
- для произвольно выбранной коммутации одного разряда входа с одним разрядом выхода в управлении участвуют шесть различных разрядов вектора U и

пять различных разрядов ключа A, что гарантирует возможность реализации всех вариантов коммутации;

- управляющий вектор V_1 является фиксированной перестановкой 32 разрядов вектора U, т. е. преобразование в первом слое блока $\mathbf{P}_{64/192}$ выполняется без какой-либо дополнительной временной задержки.

Заключительное преобразование *FT*

Процедура ***FT*** является обратным преобразованием по отношению к процедуре ***IT***. Преобразование имеет следующий вид:

$$Y = \boldsymbol{FT}(X, A),$$

где $X, Y \in \mathrm{GF}(2)^{128}$, $A \in \mathrm{GF}(2)^{64}$.

Первоначально каждый четный бит входного блока инвертируется (см. рис. 5.7), а затем каждая пара битов входного блока X с индексами $2j$–1 и $2j$ (j = 1, 2, …, 64) либо переставляется ($a_j = 1$), либо не переставляется ($a_j = 0$). Значения разрядов вектора Y вычисляются с использованием следующих формул:

$$y_{2j-1} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j-1} \quad \text{и} \quad y_{2j} = (x_{2j-1} \oplus x_{2j} \oplus 1)a_j \oplus x_{2j} \oplus 1.$$

5.3.3. Расписание использования раундовых ключей

В 12-раундовом блочном шифре SPECTR-128 расширенный ключ $Q^{(e)}$ является объединением 14 подключей, а именно:

$$Q^{(e)} = (Q_{IT}^{(e)}, Q_1^{(e)}, Q_2^{(e)}, ..., Q_{12}^{(e)}, Q_{FT}^{(e)}),$$

где $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)})$ $\forall j = 1, 2, \ldots, 12$,

причем $Q_{IT}^{(e)}, Q_{FT}^{(e)}, Q_j^{(h,e)} \in \mathrm{GF}(2)^{64}$ $\forall h = 1, 2, 3, 4$.

Расширенный ключ $Q^{(e)}$ является последовательностью пятидесяти 64-разрядных двоичных векторов, каждый из которых является одним из четырех секретных подключей K_1, K_2, K_3, K_4.

Элементы расширенного ключа $Q^{(e)}$ определяются через параметры O_i, которые, в свою очередь, вычисляются по формулам:

$$O_{2i-1} = K_{2i-1+e} \text{ и } O_{2i} = K_{2i-e},$$

где $K = K_1 \parallel K_2 \parallel K_3 \parallel K_4$ и $i = 1, 2$.

Схема коммутации между O_i и K_j представлена на рис. 5.15.

Рис. 5.15. Схема коммутации подключей

Соответствие подключей $Q_{IT}^{(e)}, Q_{FT}^{(e)}, Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)}$ ($\forall j = 1, 2, \ldots, 12$) задается табл. 5.9.

Таблица 5.9. *Расписание использования раундовых ключей*

$Q_{IT}^{(e)}$	j =	1	2	3	4	5	6	7	8	9	10	11	12	$Q_{FT}^{(e)}$
O_4	$Q_j^{(1,e)}$ =	O_1	O_3	O_2	O_4	O_3	O_1	O_4	O_2	O_1	O_3	O_2	O_4	O_3
	$Q_j^{(2,e)}$ =	O_3	O_1	O_4	O_2	O_1	O_3	O_2	O_4	O_3	O_1	O_4	O_2	
	$Q_j^{(3,e)}$ =	O_4	O_2	O_1	O_3	O_2	O_4	O_3	O_1	O_4	O_2	O_1	O_3	
	$Q_j^{(4,e)}$ =	O_2	O_4	O_3	O_1	O_4	O_2	O_1	O_3	O_2	O_4	O_3	O_1	

Так, например, для пятого раунда ($j = 5$) формальным параметрам $A^{(1)}$, $A^{(2)}$, $A^{(3)}$, $A^{(4)}$ процедуры ***Crypt*** соответствует последовательность подключей O_3, O_1, O_2, O_4, а именно:

$$Q_5^{(e)} = (Q_5^{(1,e)}, Q_5^{(2,e)}, Q_5^{(3,e)}, Q_5^{(4,e)}) = (O_3, O_1, O_2, O_4).$$

То есть

$$Q_5^{(e)} = (K_{3+e}, K_{1+e}, K_{2-e}, K_{4-e}).$$

Для режима зашифрования $Q_5^{(0)} = (K_3, K_1, K_2, K_4)$,

а для режима расшифрования $Q_5^{(1)} = (K_4, K_2, K_1, K_3)$.

Для начального и заключительного преобразований ключи $Q_{IT}^{(e)}$ и $Q_{FT}^{(e)}$ имеют вид:

$$Q_{IT}^{(e)} = O_4 \text{ и } Q_{FT}^{(e)} = O_3.$$

То есть

$$Q_{IT}^{(e)} = K_{4-e} \text{ и } Q_{FT}^{(e)} = K_{3+e}.$$

Для режима зашифрования

$$Q_{IT}^{(0)} = K_4 \text{ и } Q_{FT}^{(0)} = K_3,$$

а для режима расшифрования

$$Q_{IT}^{(1)} = K_3 \text{ и } Q_{FT}^{(1)} = K_4.$$

5.4. Шифр (алгоритм) CIKS-128

Алгоритм CIKS-128 является разновидностью итеративного блочного шифра с использованием схемы патента РФ № 2140714 в качестве базовой схемы одного раунда. По сравнению с алгоритмом SPECTR-128 изменена структура раундовой функции. В частности, в раундовой функции вместо блока управляемых перестановок $\mathbf{P}_{64/192}$ и нелинейной векторной булевой функции **G** используются две одинаковые нелинейные векторные булевы функции **G**.

Как и алгоритм SPECTR-128, алгоритм CIKS-128 является практической реализацией основных идей синтеза скоростного блочного шифра на основе управляемых операций, рассматривавшихся в предыдущих главах, и ориентирован на скоростную обработку информации с 64-разрядной шиной обмена данными.

5.4.1. Общая схема блочного шифрования

Для алгоритма CIKS-128 общая схема шифрования (зашифрование и расшифрование) определяется формулой:

$$Y = \mathbf{F}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от 256-битового секретного ключа K и режима шифрования e ($e = 0$ — зашифрование, $e = 1$ — расшифрование), причем в режиме зашифрования X является исходным блоком двоичных данных (открытым текстом), а в режиме расшифрования — преобразованным блоком двоичных данных (шифртекстом). В режиме зашифрования результирующее значение Y является шифртекстом, а в режиме расшифрования — открытым текстом.

Секретный ключ K представлен в виде объединения четырех 64-разрядных подключей: $K = K_1 \| K_2 \| K_3 \| K_4$, где и $K_1, K_2, K_3, K_4 \in \mathrm{GF}(2)^{64}$.

Описание алгоритма шифрования (функции **F**) приведено в *разд. 5.4.2*, а процедура формирования рабочего ключа $Q^{(e)} = \mathbf{H}(K, e)$ в *разд. 5.4.3*.

5.4.2. Алгоритм шифрования

Функция **F** (см. рис. 5.5) реализована в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. 12 раундов (циклов) *преобразований* с использованием процедуры ***Crypt***.
3. Заключительного преобразования ***FT***.

Первоначально выполняется процедура ***IT***: $Y = \boldsymbol{IT}(X, Q_{IT}^{(e)})$.

Блок Y делится на два подблока L_0 и R_0 одинаковой длины, т. е. $(L_0, R_0) = Y$, где $L_0, R_0 \in \mathrm{GF}(2)^{64}$. Затем выполняются 12 раундов преобразований с использованием процедуры ***Crypt*** в соответствии с формулами:

$$L_j = \boldsymbol{Crypt}(R_{j-1}, L_{j-1}, Q_j^{(e)}),$$

$$R_j = L_{j-1}, \ (j = 1, 2, \ldots, 12).$$

После 12-го раунда над блоком $X = (R_{12}, L_{12})$ выполняется заключительное преобразование ***FT*** в соответствии с формулой:

$$Y = \boldsymbol{FT}(X, Q_{FT}^{(e)}).$$

Начальное *(IT)* и заключительное *(FT)* преобразования

Используется простейшая операция поразрядного сложения с ключом, а именно:

$$Y = \boldsymbol{IT}(X, A) = X \oplus A$$

и

$$Y = \boldsymbol{FT}(X, B) = X \oplus B,$$

где $X, Y, A, B \in \mathrm{GF}(2)^{128}$.

Формальный параметр A соответствует ключу $Q_{IT}^{(e)}$, а параметр B — ключу $Q_{FT}^{(e)}$.

Процедура *Crypt*

Формальная запись процедуры ***Crypt*** имеет вид:

$$R = \boldsymbol{Crypt}(R, L, (A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)})),$$

где $L, R \in \mathrm{GF}(2)^{64}$ и $A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)} \in \mathrm{GF}(2)^{64}$.

Элементы $A^{(1)}, A^{(2)}, A^{(3)}, A^{(4)}$ являются формальными параметрами, ключевой информации, т. е. параметру $A^{(i)}$, соответствует $Q_j^{(i,e)}$ — элемент раундового ключа

$Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)})$. Соответственно, R и L — формальные параметры для левой и правой частей блока входных данных.

Особенностью данной реализации процедуры ***Crypt*** является использование в одном раунде двух векторных булевых функций **G**, четырех подключей длины 64 (участвуют все биты 256-разрядного секретного ключа K), а также инволюции между блоками $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ при преобразовании правой части входных данных.

В процедуре используются:

- *циклический сдвиг* ">>>" подблока данных на фиксированное значение в сторону младших разрядов;
- *побитовое сложение* "$\oplus$";
- две *нелинейные векторные булевы функции* **G** с одинаковой структурой;
- *блоки управляемых перестановок* $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ равновероятного смещения первого порядка;
- *однослойный блок управляемых перестановок* $\mathbf{P}_{2\times 64/1}$ (для обеспечения универсальности схемы с некоммутативной структурой преобразования информации);
- *фиксированные 64-разрядные коммутаторы*, заданные перестановкой $\boldsymbol{\pi}$ и инволюцией **I**;
- *процедура расширения управляющего вектора* **E** (блок расширения).

В качестве блоков $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ используется двухуровневая сеть Клоса $\mathbf{C}_{<8,\,8>}$ (см. рис. 5.13), причем первый уровень составляют блоки $\mathbf{R}_3$, а второй — блоки $\mathbf{R}_3^{-1}$, структура которых приведена в *гл. 3* (см. рис. 3.22). Поскольку структура блоков $\mathbf{P}_{64/192}$ и $\mathbf{P}_{64/192}^{-1}$ совпадает, на рис. 5.13 колонка I соответствует управляющему вектору для прямого преобразования, а колонка II — для обратного преобразования.

Последовательность выполняемых преобразований и их взаимодействие представлены на рис. 5.16.

Временная ось выполнения операций в процедуре ***Crypt*** представлена в табл. 5.10.

Как следует из табл. 5.10, большинство операций выполняется параллельно, а общее время, необходимое для выполнения процедуры, равно последовательному выполнению четырнадцати операций типа XOR.

Рис. 5.16. Схема процедуры ***Crypt***

Таблица 5.10. *Временная ось выполнения операций в процедуре* ***Crypt***

№№ пп	Временная ось ($\tau_{\oplus}$)	Операции
1	0	$L' = \mathbf{P}_{\pi}(L)$
2	0 — 1	$V = \mathbf{E}_{A^{(1)}\|\|A^{(4)}}(L) \quad (W \,\|\|\, W') = \mathbf{P}_{2\times 64/1(e)}(L \,\|\|\, L') \quad V' = \mathbf{E}_{A^{(3)}\|\|A^{(2)}}(L)$
3	0 — 6	$R = \mathbf{P}_{64/192(V)}(R)$
4	1 — 6	$X = \mathbf{G}_{A^{(1)}\|\|A^{(4)}}(W) \quad X' = \mathbf{G}_{A^{(3)}\|\|A^{(2)}}(W')$
5	6 — 7	$R = \mathbf{I}(R \oplus X)$
6	7 — 8	$R = R \oplus X'$
7	8 — 14	$R = \mathbf{P}^{-1}_{64/192(V')}(R)$

Фиксированные коммутаторы P_{π} и I

Основное предназначение перестановки π состоит в изменении взаимного расположения битов блока данных, поступающего одновременно на вход двух нелинейных векторных булевых функций **G**, имеющих идентичный вид, но с использованием разных раундовых ключей.

Перестановка состоит из четырех циклов длины 16 и имеет следующий вид:

$$\pi = (1,50,9,42,17,34,25,26,33,18,41,10,49,2,57,57)$$
$$(3,64,43,24,19,48,59,8,35,32,11,56,51,16,27,40)$$
$$(4,7,28,47,52,23,12,63,36,39,60,15,20,55,44,31)$$
$$(5,14,13,6,21,62,29,54,37,46,45,38,53,30,61,22)$$

Соответственно, подстановочное преобразование $\mathbf{P}_\pi$, реализуемое данной перестановкой, задается формулой $Y = \mathbf{P}_\pi(X)$, где $X, Y \in \mathrm{GF}(2)^{64}$.

А именно:

$$y_{\pi(i)} = x_i.\ \forall i = 1, 2, \ldots, 64.$$

Например, $y_{50} = x_1$, $y_9 = x_{50}$, $y_1 = x_{57}$, и т. д.

Инволюция $\mathbf{I}$ имеет следующий вид:

$$Y = \mathbf{I}(X) = (X_6^{>>>4} \| X_5^{>>>4} \| X_4^{>>>4} \| X_3^{>>>4} \| X_2^{>>>4} \| X_1^{>>>4} \| X_8^{>>>4} \| X_7^{>>>4}),$$

где $X = (X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8)$ и $X_1, X_2, \ldots, X_8 \in \mathrm{GF}(2)^8$. То есть выполняется перестановка байтов и циклический сдвиг каждого байта на 4.

Чтобы убедиться, что действительно задана инволюция, достаточно выполнить преобразование $\mathbf{I}^2$. А именно:

$$\mathbf{I}^2(X) = \mathbf{I}((X_6)^{>>>4} \| (X_5)^{>>>4} \| (X_4)^{>>>4} \| (X_3)^{>>>4} \| (X_2)^{>>>4} \| (X_1)^{>>>4} \| (X_8)^{>>>4} \| (X_7)^{>>>4}) =$$
$$= \mathbf{I}(((X_1)^{>>>4})^{>>>4} \| ((X_2)^{>>>4})^{>>>4} \| ((X_3)^{>>>4})^{>>>4} \| ((X_4)^{>>>4})^{>>>4} \|$$
$$\| ((X_5)^{>>>4})^{>>>4} \| ((X_6)^{>>>4})^{>>>4} \| ((X_7)^{>>>4})^{>>>4} \| ((X_8)^{>>>4})^{>>>4}) =$$
$$= (X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8) = X.$$

Однослойный блок управляемых перестановок $P_{2\times 64/1}$

Отображение, реализуемое блоком $\mathrm{P}_{2\times\omega/1}$, имеет следующий вид:

$$(W \| W') = \mathbf{P}_{2\times 64/1}(L \| L', e),$$

причем значения j-х разрядов векторов W и W' ($j = 1, 2, \ldots, 64$) вычисляются с использованием следующих формул:

$$w_j = (l_j \oplus l'_j)e \oplus l_j \quad \text{и} \quad w'_j = (l_j \oplus l'_j)e \oplus l'_j.$$

Векторная булева функция G

В качестве векторной булевой функции $\mathbf{G}$ используется управляемая подстановочная операция, являющаяся одной из реализаций последовательной модели и формируемая в соответствии с правилами, представленными в *разд. 4.3.3*, *4.3.7—4.3.10* и *4.5.6*.

Формальная запись функции **G** имеет вид:

$$X = \mathbf{G}(W, A, B), \text{ где } X, W, A, B \in \mathrm{GF}(2)^{64}.$$

В векторном виде функция **G** определяется формулой:

$$X = U^{(0)} \oplus A^{(0)} \oplus (U^{(1)} \otimes B^{(0)}) \oplus (U^{(2)} \otimes U^{(5)}) \oplus (U^{(6)} \otimes A^{(1)}) \oplus (A^{(2)} \otimes B^{(1)}) \oplus$$
$$\oplus (U^{(3)} \otimes U^{(4)}) \oplus (U^{(1)} \otimes U^{(4)} \otimes U^{(6)}) \oplus (U^{(2)} \otimes U^{(6)} \otimes B^{(1)}) \oplus$$
$$\oplus (U^{(1)} \otimes U^{(2)} \otimes U^{(4)} \otimes B^{(1)}),$$

в которой векторы U_j (j = 0, 1, 2, …, 6) выражаются рекурсивно через W, а именно:

$$U^{(0)} = (u_1^{(0)}, u_2^{(0)}, u_3^{(0)}, ..., u_{63}^{(0)}, u_{64}^{(0)}) = W = (w_1, w_2, w_3, ..., w_{63}, w_{64})$$

и

$$U^{(j)} = (u_1^{(j)}, u_2^{(j)}, u_3^{(j)}, ..., u_{63}^{(j)}, u_{64}^{(j)}) = (1, u_1^{(j-1)}, u_2^{(j-1)}, u_3^{(j-1)}, ..., u_{63}^{(j-1)}).$$

Аналогично определяются векторы $A^{(j)}$ и $B^{(j)}$ для j = 0, 1, 2.

Учитывая возможность параллельного выполнения некоторых операций, легко установить, что функция **G** выполняется за пять тактов задержки ($5\tau_{\oplus}$).

Обоснование выбора нелинейной функции G

Определим отображение **G**: $\mathrm{GF}(2)^{64} \times \mathrm{GF}(2)^{64} \times \mathrm{GF}(2)^{64} \to \mathrm{GF}(2)^{64}$ в виде векторной булевой функции $Y = \mathbf{G}(X, A, B)$ следующего вида:

$$(y_1, y_2, \ldots, y_{64}) = (\mathbf{g}_1(X, A, B), \mathbf{g}_2(X, A, B), \ldots, \mathbf{g}_{64}(X, A, B)),$$

где $\mathbf{g}_i$ — некоторые образующие булевы функции. В качестве *прототипа образующих функций (см. разд. 4.3.7)* будем рассматривать единую унифицированную булеву функцию от двенадцати переменных, имеющую следующий вид:

$$\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_{12}) = z_{11} \oplus z_{12} \oplus \varphi(z_1, z_2, \ldots, z_{10}),$$

где $\varphi(z_1, z_2, \ldots, z_{10}) = z_1z_2 \oplus z_3z_4 \oplus z_5z_6 \oplus z_7z_8 \oplus z_9z_{10} \oplus z_1z_5z_9 \oplus z_3z_5z_7 \oplus z_1z_3z_7z_9$ является бент-функцией *(см. разд. 4.2.8)*. Преобразование Уолша-Адамара *(см. разд. 4.2.3)* функции **g** состоит из 1024-х элементов, принимающих значение $|\mathbf{U}_{\alpha}^{*}(g)| = 128$ для всех векторов $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_{10}, 0, 0)$, а для остальных 3072-х элементов — значение $\mathbf{U}_{\alpha}^{*}(g) = 0$.

С учетом выбранной структуры и свойств бент-функций имеем следующие характеристики функции **g**:

- нелинейность функции **g** равна $\mathbf{N}(\mathbf{g}) = 2^{m-1} - 2^{((m+2)/2)-1} = 1984$ (при $m = 12$), что достаточно близко к максимально возможной нелинейности булевой функции от двенадцати переменных ($\mathbf{N}_{\max} = 2^{m-1} - 2^{(m/2)-1} = 2016$) *(см. разд. 4.2.4)*;
- **g** — корреляционно-иммунная функция на множестве векторов $Z^* = \{(z_1, z_2, \ldots, z_{12})\} \subset \mathrm{GF}(2)^{12}$, где $z_{11} \neq 0$ или $z_{12} \neq 0$, что составляет 75% от всего множества $\mathrm{GF}(2)^{12}$ *(см. разд. 4.2.7)*;

- **g** обладает хорошими автокорреляционными свойствами, т. к. для всех ненулевых лавинных векторов $(\Delta z_1, \Delta z_2, \ldots, \Delta z_{12}) \in \mathrm{GF}(2)^{12}$, кроме трех ((0, …, 0, 0, 1), (0, …, 0, 1, 0) и (0, …, 0, 1, 1)), выполняется *критерий распространения 12-го порядка*, т. е. функция $\Delta\mathbf{g} = \mathbf{g}(z_1, z_2, \ldots, z_{12}) \oplus \oplus\, \mathbf{g}(z_1 \oplus \Delta z_1, z_2 \oplus \Delta z_2, \ldots, z_{12} \oplus \Delta z_{12})$ является сбалансированной *(см. разд. 4.2.6)*;
- степень алгебраической нормальной формы функции **g** равна четырем ($\mathbf{deg}(\mathbf{g}) = 4$) *(см. разд. 4.2.2)*.

Переход от общего вида функции **g** к конкретным образующим функциям последовательной модели управляемой подстановочной операции осуществляется по следующей подстановке переменных *(подробную информацию см. в разд. 4.5.6)*:

$$\begin{pmatrix} z_1 & z_2 & z_3 & z_4 & z_5 & z_6 & z_7 & z_8 & z_9 & z_{10} & z_{11} & z_{12} \\ u_{i-1} & b_i & u_{i-2} & u_{i-5} & u_{i-6} & a_{i-1} & b_{i-1} & a_{i-2} & u_{i-4} & u_{i-3} & u_i & a_i \end{pmatrix},$$

где $\forall i \in \{1, \ldots, n\}$ и $n = 64$.

Таким образом, имеем множество образующих функций вида:

$$y_i = g_i = u_i \oplus a_i \oplus u_{i-1}b_i \oplus u_{i-2}u_{i-5} \oplus a_{i-1}u_{i-6} \oplus b_{i-1}a_{i-2} \oplus$$
$$\oplus\, u_{i-4}u_{i-3} \oplus u_{i-1}u_{i-4}u_{i-6} \oplus u_{i-2}b_{i-1}u_{i-6} \oplus u_{i-1}u_{i-2}b_{i-1}u_{i-4},$$

где l_j, a_j, b_j — компоненты, соответственно, векторов $L, A, B \in \mathrm{GF}(2)^{64}$, а начальные условия соответствуют вектору $(x_{-4}, x_{-3}, x_{-2}, x_{-1}, x_0) = (1, 1, 1, 1, 1)$.

Процедура расширения управляющего вектора E (блок расширения)

Процедура предназначена для формирования 192-разрядного управляющего вектора из 64-разрядного. Формальная запись процедуры **E** имеет следующий вид:

$$V = (V_1 \parallel V_2 \parallel V_3 \parallel V_4 \parallel V_5 \parallel V_6) = \mathbf{E}(U, A, B) = \mathbf{E}_{A\|B}(U),$$

где $V \in \mathrm{GF}(2)^{192}$, $V_1, V_2, V_3, V_4, V_5, V_6 \in \mathrm{GF}(2)^{32}$, $U, A, B \in \mathrm{GF}(2)^{64}$.

Фактически вектор V вычисляется в соответствии с формулами:

$$V_1 = \mathbf{P}_{\pi_1}(U_{lo}); \quad V_2 = \mathbf{P}_{\pi_2}((U \oplus A)_{lo}); \quad V_3 = \mathbf{P}_{\pi_3}((U \oplus B)_{lo});$$
$$V_4 = \mathbf{P}_{\pi_4}(U_{hi}) = U_{hi}; \quad V_5 = \mathbf{P}_{\pi_5}((U \oplus A)_{hi}); \quad V_6 = \mathbf{P}_{\pi_6}((U \oplus A)_{hi}),$$

где для блока $\mathbf{P}_{64/192}$ преобразования $\mathbf{P}_{\pi_1}, \mathbf{P}_{\pi_2}, \ldots, \mathbf{P}_{\pi_6}$ реализуют фиксированные перестановки битов, заданные табл. 5.11.

Так, например, 32 в первой строке, соответствующей вектору V_1, означает, что на вход $v_2^{(1)}$ управления блоком $\mathbf{P}_{2/1}$ поступает значение u_{32}, во второй строке 22

соответствует значению $v_{32}^{(2)} = u_{22} \oplus a_{22}$, а 53 в шестой строке — значению $v_9^{(6)} = u_{53} \oplus b_{53}$.

Таблица 5.11. *Таблица значений управляющих разрядов для блока* $\mathbf{P}_{64/192}$

V_1	31	32	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	1	2
V_2	10	24	25	26	29	13	27	16	1	2	31	32	3	4	19	6	7	8	9	23	11	12	28	15	14	30	17	18	5	20	21	22
V_3	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	1	2	3	4	5	6	7	8	12	10	11	9
V_4	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
V_5	55	56	57	58	59	60	61	62	63	64	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54
V_6	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	33	34	35	36	37	38	39	40	41	42	43	44

Расширение составлено таким образом, что 32 старших разряда вектора U участвуют в управлении только блоками $\mathbf{R}_3^{-1}$, а младшие — только блоками $\mathbf{R}_3$, причем в каждом блоке $\mathbf{R}_3$ и $\mathbf{R}_3^{-1}$ все номера разрядов векторов U, A и B разные.

Процедура расширения управляющего вектора обеспечивает:

- равномерность участия всех 64 разрядов векторов U, A и B в управлении блоком $\mathbf{P}_{64/128}$;
- для произвольно выбранной коммутации одного разряда входа с одним разрядом выхода в управлении участвуют шесть различных разрядов вектора U и по два различных разряда ключей A и B, что гарантирует возможность реализации всех вариантов коммутации;
- управляющий вектор V_1 является фиксированной перестановкой 32-х разрядов вектора U, т. е. преобразование в первом слое блока $\mathbf{P}_{64/192}$ выполняется без какой-либо дополнительной временной задержки.

5.4.3. Расписание использования раундовых ключей

Как и в 12-раундовом блочном шифре SPECTR-128, в блочном шифре CIKS-128 расширенный ключ $Q^{(e)}$ является объединением 14 подключей, а именно:

$$Q^{(e)} = (Q_{IT}^{(e)}, Q_1^{(e)}, Q_2^{(e)}, ..., Q_{12}^{(e)}, Q_{FT}^{(e)}),$$

где $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)})$ $\forall j = 1, 2, \ldots, 12$,

причем $Q_{IT}^{(e)}, Q_{FT}^{(e)} \in \mathrm{GF}(2)^{128}$, а $Q_j^{(h,e)} \in \mathrm{GF}(2)^{64}$ $\forall h = 1, 2, 3, 4$.

Расширенный ключ $Q^{(e)}$ является последовательностью пятидесяти двух 64-разрядых двоичных векторов, каждый из которых является одним из четырех секретных подключей K_1, K_2, K_3, K_4. Элементы расширенного ключа $Q^{(e)}$ определяются через параметры O_i, которые, в свою очередь, вычисляются по формулам:

$$O_{2i-1} = K_{2i-1+e} \text{ и } O_{2i} = K_{2i-e},$$

где $K = K_1 \| K_2 \| K_3 \| K_4$ и $i = 1, 2$.

Схема коммутации между O_i и K_j представлена на рис. 5.17.

Рис. 5.17. Схема коммутации подключей

Соответствие подключей $Q_{IT}^{(e)}, Q_{FT}^{(e)}, Q_j^{(1,e)}, Q_j^{(2,e)}, Q_j^{(3,e)}, Q_j^{(4,e)}$ ($\forall j = 1, 2, \ldots, 12$) задается табл. 5.12.

Таблица 5.12. *Расписание использования раундовых ключей*

$Q_{IT}^{(e)}$	j =	1	2	3	4	5	6	7	8	9	10	11	12	$Q_{FT}^{(e)}$
$O_1 \| O_3$	$Q_j^{(1,e)}$ =	O_2	O_3	O_1	O_4	O_2	O_1	O_2	O_1	O_3	O_2	O_4	O_1	$O_2 \| O_4$
	$Q_j^{(2,e)}$ =	O_4	O_1	O_3	O_2	O_4	O_3	O_4	O_3	O_1	O_4	O_2	O_3	
	$Q_j^{(3,e)}$ =	O_1	O_4	O_2	O_3	O_1	O_2	O_1	O_2	O_4	O_1	O_3	O_2	
	$Q_j^{(4,e)}$ =	O_3	O_2	O_4	O_1	O_3	O_4	O_3	O_4	O_2	O_3	O_1	O_4	

Например, для пятого раунда ($j = 5$) формальным параметрам $A^{(1)}$, $A^{(2)}$, $A^{(3)}$, $A^{(4)}$ процедуры ***Crypt*** соответствует последовательность подключей O_2, O_4, O_1, O_3, а именно:

$$Q_5^{(e)} = (Q_5^{(1,e)}, Q_5^{(2,e)}, Q_5^{(3,e)}, Q_5^{(4,e)}) = (O_2, O_4, O_1, O_3).$$

То есть

$$Q_5^{(e)} = (K_{2-e}, K_{4-e}, K_{1+e}, K_{3+e}).$$

Для режима зашифрования $Q_5^{(0)} = (K_2, K_4, K_1, K_3)$,

а для режима расшифрования $Q_5^{(1)} = (K_1, K_3, K_2, K_4)$.

Для начального и заключительного преобразований ключи $Q_{IT}^{(e)}$ и $Q_{FT}^{(e)}$ имеют вид:

$$Q_{IT}^{(e)} = (O_1 \| O_3) \text{ и } Q_{FT}^{(e)} = (O_2 \| O_4).$$

То есть

$$Q_{IT}^{(e)} = (K_{1+e} \| K_{3+e}) \text{ и } Q_{FT}^{(e)} = (K_{2-e} \| K_{4-e}).$$

Для режима зашифрования

$$Q_{IT}^{(0)} = (K_1 \| K_3) \text{ и } Q_{FT}^{(0)} = (K_2 \| K_4),$$

а для режима расшифрования

$$Q_{IT}^{(1)} = (K_2 \| K_4) \text{ и } Q_{FT}^{(1)} = (K_1 \| K_3).$$

Универсальность схемы

Для доказательства универсальности схемы достаточно показать универсальность (обратимость) схемы одного раунда, а точнее, универсальность преобразования, выделенного на рис. 5.16 серым цветом. Выпишем формулы для $e = 0$ и $e = 1$.

$$\boldsymbol{e} = 0 \quad \Rightarrow \quad R' = \mathbf{I}(R) \oplus \mathbf{I}(\mathbf{G}(L, A^{(1)} \| A^{(4)})) \oplus \mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)}).$$
$$\boldsymbol{e} = 1 \quad \Rightarrow \quad U' = \mathbf{I}(U) \oplus \mathbf{I}(\mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)})) \oplus \mathbf{G}(L, A^{(1)} \| A^{(4)}).$$

Выполняя последовательно зашифрование и расшифрование ($U = R'$), получаем:

$$U' = \mathbf{I}(U) \oplus \mathbf{I}(\mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)})) \oplus \mathbf{G}(L, A^{(1)} \| A^{(4)}) =$$
$$= \mathbf{I}^2(R) \oplus \mathbf{I}^2(\mathbf{G}(L, A^{(1)} \| A^{(4)})) \oplus \mathbf{I}(\mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)})) \oplus$$
$$\oplus \mathbf{I}(\mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)})) \oplus \mathbf{G}(L, A^{(1)} \| A^{(4)}) = R.$$

Выполняя последовательно расшифрование и зашифрование ($R = U'$), получаем:

$$R' = \mathbf{I}(R) \oplus \mathbf{I}(\mathbf{G}(L, A^{(1)} \| A^{(4)})) \oplus \mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)}) = \mathbf{I}^2(U) \oplus \mathbf{I}^2(\mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)})) \oplus$$
$$\oplus \mathbf{I}(\mathbf{G}(L, A^{(1)} \| A^{(4)})) \oplus \mathbf{I}(\mathbf{G}(L, A^{(1)} \| A^{(4)})) \oplus \mathbf{G}(\mathbf{P}_\pi(L), A^{(3)} \| A^{(2)}) = U.$$

То есть действительно реализована универсальная схема.

5.5. Перспективные программные шифры на основе управляемых перестановок

В *гл. 2* рассматривались скоростные программные шифры, основанные на выборке подключей в зависимости от преобразуемых данных. Для процессоров типа Celeron 500 МГц такие шифры обеспечивают преобразование данных со скоростью более 200 Мбит/с.

Поскольку оптоволоконные каналы связи обеспечивают возможность передачи информации с существенно более высокой скоростью, производителями процессоров рассматривается вопрос о встраивании в процессоры (на аппаратном уровне) криптографических средств преобразования информации, реализующих один из алгоритмов, получивших всеобщее признание и удовлетворяющих современным требованиям стойкости и скорости. Недостатком данного направления является, фактически, "навязывание" потребителю определенного алгоритма преобразования информации, независимо от степени доверия пользователя, и ограничение свободы выбора.

Альтернативным направлением в решении данного вопроса является расширение списка команд, реализующих преобразования криптографической направленности, обладающих высоким быстродействием и выполненных в виде специальных инструкций. В частности, представляет интерес реализация *операции управляемых битовых перестановок*, эффективность которой при построении аппаратных шифров обсуждалась в *гл. 3*. Размерность блоков управляемых перестановок, реализующих такие операции, должна соответствовать разрядности используемого процессора.

Так, например, для процессоров, оперирующих с 32-разрядными двоичными векторами (числами, словами), выполнение битовых перестановок эффективно

Рис. 5.18. Блок управляемых перестановок $\mathbf{P}_{32/96}$

решается с использованием блока управляемых перестановок второго порядка $\mathbf{P}_{32/96}$ (рис. 5.18).

Описание такой команды, а также ее эффективное использование в программных блочных шифрах и рассматривается в данном разделе.

5.5.1. Описание гипотетической команды *DDP32*

Для новой команды введем условное обозначение $\mathbf{P}_{32/32}^{(U,e)}$ и идентификатор ***DDP32*** (Data Dependent Permutation 32 bit, зависящая от данных 32-разрядная управляемая перестановка).

Входными параметрами такой команды являются:

- X — 32-разрядный преобразуемый бинарный вектор;
- U — 32-разрядный управляющий вектор, формирующий модификацию перестановки;
- e — одноразрядный управляющий вектор, значение которого определяет режим преобразования данных ($e = 0$ — "прямая" перестановка, $e = 1$ — "обратная" перестановка).

Результатом выполнения команды $\mathbf{P}_{32/32}^{(U,e)}$ является перестановка 32-разрядного бинарного вектора X. Другими словами, команда ***DDP32*** реализует отображение $\mathrm{GF}(2)^{32}\times \mathrm{GF}(2)^{32}\times \mathrm{GF}(2)\rightarrow \mathrm{GF}(2)^{32}$, а именно:

$$Y = \mathbf{P}_{32/32}(X, U, e).$$

Поскольку в качестве прототипа операции используется блок второго порядка $\mathbf{P}_{32/96(V)}$, в котором разрядность управляющего вектора равна 96, в новой команде помимо блока $\mathbf{P}_{32/96(V)}$ необходимо реализовать процедуру расширения управляющего вектора $V = \mathbf{E}(U)$, а именно:

$$\mathbf{E}\colon \mathrm{GF}(2)^{32} \rightarrow \mathrm{GF}(2)^{96}),$$

обеспечивающую коммутацию каждого бита двоичного вектора U с тремя различными битами 96-битового двоичного вектора V.

В связи с тем, что для выполнения обратного преобразования $\mathbf{P}_{32/96(V)}^{-1}$ в блоке $\mathbf{P}_{32/96}$ достаточно изменить порядок следования векторов $V_1, V_2, ..., V_6$ на обратный, т. е. $\mathbf{P}_{32/96(V)}^{-1} = \mathbf{P}_{32/96(V')}$, где $V = (V_1, V_2, ..., V_6)$ и $V' = (V_6, V_5, ..., V_1)$, изменение такого порядка следования эффективно решается с использованием однослойного блока $\mathbf{P}_{96/1}^{(e)}$, представленного на рис. 5.19.

Таким образом, операционный блок, реализующий команду $\mathbf{P}_{32/32}^{(U,e)}$, имеет вид, представленный на рис. 5.20.

Рис. 5.19. Блок $\mathbf{P}^{(e)}_{96/1}$ управления режимом преобразования данных в команде $\mathbf{P}^{(U,e)}_{32/32}$

Рис. 5.20. Блок управляемых перестановок $\mathbf{P}^{(U,e)}_{32/32}$

Благодаря такой структуре, блок $\mathbf{P}^{(U,e)}_{32/32}$ выполняет либо прямую управляемую перестановочную операцию ($e = 0$), либо обратную перестановочную операцию ($e = 1$).

При разработке программных шифров на основе управляемых перестановок предполагается, что значение вектора $U = (u_1, u_2, ..., u_{32})$ является блоком преобразуемых данных, причем каждый бит вектора U определяет три различных бита двоичного вектора V. Необходимо отметить, что блок $\mathbf{P}_{32/96}$ является блоком второго порядка, поэтому существует ровно два варианта коммутации между произвольным входным и произвольным выходным разрядами, причем в каждой такой коммутации участвуют шесть элементарных переключателей $\mathbf{P}_{2/1}$, управляемых вектором V. Следовательно, для обеспечения возможности любой такой коммутации вектором U при формировании соответствия между битами векторов U и V в процедуре $\mathbf{E}$ целесообразно руководствоваться следующим критерием:

Замечание

В управлении шестью блоками $\mathbf{P}_{2/1}$, реализующими коммутацию между произвольно выбранными разрядами входа и выхода, должны участвовать шесть различных разрядов вектора U.

В частности, в табл. 5.13 приведен пример соответствия между векторами U и V, который удовлетворяет такому критерию и может быть использован при формировании процедуры расширения **E**.

Так, например, в первой строке, соответствующей вектору V_1, 10 означает, что на вход $v_2^{(1)}$ управления блоком $\mathbf{P}_{2/1}$ поступает значение u_{10}, во второй строке 10 соответствует значению $v_{15}^{(2)} = u_{10}$, а 26 в шестой строке — значению $v_7^{(6)} = u_{26}$.

Таблица 5.13. *Таблица соответствия между разрядами векторов U и V для блока* $\mathbf{P}_{32/96}$

V_1	9	10	11	12	1	2	3	4	5	6	7	8	13	14	15	16
V_2	16	15	14	13	6	5	8	7	1	2	3	4	12	11	10	9
V_3	4	3	2	1	11	12	15	16	14	13	9	10	8	7	6	5
V_4	27	28	19	20	31	32	23	24	17	18	21	22	25	26	29	30
V_5	31	32	17	18	29	30	21	22	25	26	27	28	23	24	19	20
V_6	24	23	22	21	28	27	26	25	32	31	30	29	20	19	18	17

При аппаратной реализации элементарный переключатель реализуется с использованием не более 12 транзисторов. Соответственно, на реализацию операционного блока $\mathbf{P}_{32/32}^{(U,e)}$ потребуется не более 1800 транзисторов. Поэтому включение команды ***DDP32*** в набор стандартных операций современных универсальных процессоров массового назначения не приведет к существенному усложнению схемотехнической реализации таких процессоров. Наличие же этой команды обеспечит возможность увеличить в несколько раз (до 1 Гбит/с и более) быстродействие программных шифров, использующих данную операцию, что позволит решить ряд актуальных проблем при разработке систем компьютерной безопасности, обеспечивающих защиту информации в реальном масштабе времени в современных высокопроизводительных автоматизированных системах.

5.5.2. Программный шифр SPECTR-SZ

Рассмотрим программный шифр SPECTR-SZ, в котором используется новая команда управляемых перестановок ***DDP32***. В статье [5] данный шифр имеет название DDP-S, однако в книге название изменено по следующим обстоятельствам:

- по своей структуре рассматриваемый шифр является модернизированным вариантом программного шифра SPECTR-Z, в котором операция циклического сдвига заменена новой командой ***DDP32***;

- внесены изменения в описание алгоритма, обусловленные не только необходимостью корректного использования команды ***DDP32***, но и упрощающие понимание функционирования алгоритма;
- в этой же главе рассматриваются два блочных шифра (DDP-S64 и DDP-S128), которые по своей структуре существенно отличаются от программного шифра SPECTR-Z, и в такой ситуации использование названия DDP-S для рассматриваемого шифра не совсем корректно.

Обозначения и входные данные

При описании шифра SPECTR-SZ использованы нижеприведенные дополнительные обозначения.

- Байты (бинарные векторы длины 8) будем обозначать строчными латинскими буквами без выделения курсивом. Например:

$$\mathrm{u} = (u_0, u_1, \ldots, u_7), \text{ где} \forall i = 0, 1, \ldots, 7 \;\; u_i \in \{0,1\}.$$

- Обозначение **B** будем использовать для множества всех байтов:

$$\mathbf{B} \equiv \{0,1\}^8 \text{ и } \mathrm{u} \in \mathbf{B}.$$

- Под термином "слово" будем понимать 32-разрядные бинарные векторы, обозначаемые прописными латинскими буквами, выделенными курсивом. Например, $U = (u_0, u_1, \ldots, u_{31})$, где $\forall i = 0, 1, \ldots, 31 \;\; u_i \in \{0,1\}$. Очевидно, что $U = (\mathrm{u}_0, \mathrm{u}_1, \mathrm{u}_2, \mathrm{u}_3)$, где $\forall j = 0, 1 \ldots, 3 \; \mathrm{u}_j \in \mathbf{B}$ и $U \in \mathbf{B}^4$.
- Для обозначения последовательностей байтов, превышающих длину четыре, будем использовать прописные латинские буквы, выделенные курсивом с полужирным начертанием. Например, $\boldsymbol{Q} \in \mathbf{B}^{2051}$, где $\boldsymbol{Q} = (q_0, q_1, \ldots, q_{2050})$.
- Операции "$+_m$" и "$-_m$" обозначают, соответственно, "сложение по модулю 2^m" и "вычитание по модулю 2^m".
- Операция "$W \leftrightarrow V$" обозначает операцию обмена значениями для слов W и V.

Замечание

В дискретной математике, как правило, рассматриваются числа, представленные бинарными векторами заданного размера (например, m). При этом, например, обычное сложение с переносом двух чисел X и Y ($X \geq 2^{m-1}$ и $Y \geq {}^{m-1}$) теряет смысл, поскольку результирующее значение невозможно представить одним бинарным вектором размерности m. Поэтому обычно рассматривают операцию "сложения по модулю 2^m" и используют запись $(X + Y) \bmod 2^m$. При написании сложных формул такая запись влечет резкое увеличение числа скобок и, как следствие, усложняется восприятие этих формул. В статьях часто оговаривается заранее, что запись $X + Y$ означает "сложение по модулю 2^m" и, в дальнейшем, обозначение "mod 2^m" не используется. Однако такую запись применяют в случае фиксированного значения m. Если выполняются операции, в которых используются несколько различных значений ($m_1, m_1, \ldots, m_k$), возникают сложности в наглядном представлении формул и

выражений. Поэтому, авторами книги вместо общепринятой записи $(X + Y) \bmod 2^m$ предлагается использовать запись $X +_m Y$. То есть

$$X +_m Y \overset{def}{=} (X + Y) \bmod 2^m.$$

Обозначения "$+_m$" и "$-_m$" в настоящее время еще не стали общепринятыми, однако, по мнению авторов, использование таких обозначений является естественным развитием индексной символики для множеств, переменных и т. д.

Безусловно, более общими являются обозначения "$+_{2^m}$" и "$-_{2^m}$ ", поскольку позволяют рассматривать произвольный модуль p, а не только степень числа два 2 (например, "$+_p$" и "$-_p$"). Однако данное различие является менее значимым по сравнению с представлением выражений $(X + Y) \bmod 2^m$ и $X +_m Y$.

В алгоритме используются следующие шестнадцатеричные константы α, β, γ, δ:

- $\alpha = 0D_x$, $\alpha \in \mathbf{B}$;
- $\beta = FFFF07FF_x$, $\beta \in \mathbf{B}^4$;
- $\gamma = B25D28A7\ 1A62D775_x$, $\gamma \in \mathbf{B}^8$;
- $\delta = 98915E7E\ C8265EDF\ CDA31E88\ F24809DD$
  $B064BDC7\ 285DD50D\ 7289F0AC\ 6F49DD2D_x$, $\delta \in \mathbf{B}^{32}$.

Индекс "$_x$" означает, что используется шестнадцатеричное представление числа, причем младший разряд расположен справа. Например, байт $\alpha = 0D_x$ означает:

$$(\alpha_7, \alpha_6, \ldots, \alpha_0) = 0000\ 1101, \text{ т. е. } \alpha = (\alpha_0, \alpha_1, \ldots, \alpha_7) = 1011\ 0000.$$

Данная особенность в представлении чисел является следствием извечного спора между "математиками" и "программистами" о том, "с какой стороны (справа или слева) находится младший значащий бит".

При интерпретации нескольких последовательных байтов $\{u_0, u_1, \ldots, u_{s-1}\}$ двоичным числом во всех случаях байт u_{s-1} относится к *старшим* разрядам числа. Если, например, 64-разрядную константу $\gamma = B25D28A7\ 1A62D775_x$ представить в виде последовательности байтов $u_0, u_1, \ldots, u_7$, тогда $u_0 = 75_x$, $u_1 = D7_x$, …, $u_7 = B2_x$.

Шифр SPECTR-SZ ориентирован на использование в компьютерах с 32-разрядным процессором и по своей структуре построения является *блочно-поточным* шифром. Данная особенность структуры шифра заключается в том, что в процедурах преобразования данных ***Encrypt*** и ***Decrypt*** первый и последний раунды реализуют схему поточного шифра, а остальные (внутренние) раунды — итеративную схему блочного шифра.

Структура шифра

Программный шифр SPECTR-SZ обычно реализуется в виде двух модулей (см. рис. 5.21):

- *модуля инициализации* (предназначен для настройки программы и формирования всех необходимых параметров, включая и расширенный ключ ***Q***);

Рис. 5.21. Схема программного шифра SPECTR-Z

- *резидентного модуля*, обслуживающего запросы других программ на зашифрование и расшифрование данных.

Расширенный ключ $\boldsymbol{Q}$ ($\boldsymbol{Q} \in \mathbf{B}^{2051}$), вырабатываемый по секретному ключу $\boldsymbol{K}$, формируется модулем инициализации в двух случаях:

- на этапе инициализации программы;
- при смене секретного ключа.

Резидентный модуль состоит из двух подпрограмм ***Encrypt*** и ***Decrypt***, предназначенных, соответственно, для зашифрования и расшифрования информации, представленной в виде блоков данных по $\boldsymbol{m}$ 32-разрядных слов.

Процедура генерации расширенного ключа

Первоначально последовательность $\boldsymbol{Q'} = (q_0', q_1', \ldots, q'_{2050})$ ($\boldsymbol{Q'} \in \mathbf{B}^{2051}$) формируется повторением необходимого числа раз секретного ключа $\boldsymbol{K} = (k_0, k_1, \ldots, k_s)$, где $8 \le s \le 2050$.

Замечание

Такая "странная" длина последовательности $\boldsymbol{Q'}$ обусловлена необходимостью оперирования с 2048-ю 32-разрядными словами. Действительно, последовательность байтов $u_0, u_1, \ldots, u_{n-1}$ можно использовать для формирования 32-битовых слов $U_j = \{u_j, u_{j+1}, u_{j+2}, u_{j+3}\}$, где $j = 0, 1, \ldots, n-4$. Поэтому в случае использования 2048 (2^{11}) 32-разрядных слов, формируемых данным способом, длина исходной последовательности должна равняться 2051 байту ($n = 2051$).

Следующий шаг — использование процедуры ***Table_H***($\boldsymbol{Q'}$) для формирования вспомогательного ключа $\boldsymbol{H}$ ($\boldsymbol{H} \in \mathbf{B}^{2051}$).

Процедура *Table_H*

Алгоритм выполнения процедуры ***Table_H***($\boldsymbol{Q'}$) имеет следующий вид:

1. Установить значение счетчика $i := 0$.
2. Вычислить 32-байтовое число $\boldsymbol{H}_i' := (\alpha^{23+i} \bmod \gamma)^{17} \bmod \delta$, ($\boldsymbol{H}_i' \in \mathbf{B}^{32}$).

3. Увеличить значение счетчика $i := i + 1$. Если $i \neq 64$, тогда перейти к шагу 2.
4. Сформировать последовательность $\boldsymbol{H} := \| \boldsymbol{H}_0' \| \boldsymbol{H}_1' \| \ldots \| \boldsymbol{H}_{63}' \| h_0' \| h_1' \| h_2'$, где $(h_0' \| h_1' \| h_2')$ — первые три байта числа $\boldsymbol{H}_0' = (h_0', h_1', h_2', \ldots, h_{63}')$.
5. Выполнить преобразование $\boldsymbol{H} := \boldsymbol{Q}' \oplus \boldsymbol{H}$.
6. Представить $\boldsymbol{H}$ в виде последовательности байтов $\boldsymbol{H} = (h_0, h_1, \ldots, h_{2050})$.
7. **СТОП**.

Окончательно, расширенный ключ $\boldsymbol{Q}$ формируется с использованием процедуры ***Form_Key***, а именно $\boldsymbol{Q} = \boldsymbol{Form_Key}(\boldsymbol{Q}', \boldsymbol{H})$, причем последовательность $\boldsymbol{Q}'$ интерпретируется в виде объединения четырех 512-байтовых последовательностей (блоков данных) $\boldsymbol{Q}'^{(1)}$, $\boldsymbol{Q}'^{(2)}$, $\boldsymbol{Q}'^{(3)}$, $\boldsymbol{Q}'^{(4)}$ и трех байтов, а именно:

$$\boldsymbol{Q}' = \boldsymbol{Q}'^{(1)} \| \boldsymbol{Q}'^{(2)} \| \boldsymbol{Q}'^{(3)} \| \boldsymbol{Q}'^{(4)} \| q'_{2048} \| q'_{2049} \| q'_{2050}.$$

В процедуре ***FormKey*** используется одна из двух программ резидентного модуля — процедура ***Encrypt***. Процедура ***Encrypt*** зависит от трех параметров, первый из которых определяет длину преобразуемого блока данных (выраженную в 32-разрядных словах), второй — собственно преобразуемый блок данных, а третий параметр — расширенный ключ.

Процедура *FormKey*

Алгоритм выполнения процедуры ***FormKey*** имеет следующий вид:

1. Для процедуры ***Encrypt*** установить значение параметра $m := 128$, а в качестве ключа шифрования использовать последовательность $\boldsymbol{H}$.
2. Преобразовать $\boldsymbol{Q}'^{(1)}$: $\boldsymbol{Q}'^{(1)} := \boldsymbol{Encrypt}(m, \boldsymbol{Q}'^{(1)}, \boldsymbol{H})$.
3. Преобразовать $\boldsymbol{Q}'^{(2)}$: $\boldsymbol{Q}'^{(2)} := \boldsymbol{Encrypt}(m, \boldsymbol{Q}'^{(2)} \oplus \boldsymbol{Q}'^{(1)}, \boldsymbol{H})$.
4. Преобразовать $\boldsymbol{Q}'^{(3)}$: $\boldsymbol{Q}'^{(3)} := \boldsymbol{Encrypt}(m, \boldsymbol{Q}'^{(3)} \oplus \boldsymbol{Q}'^{(2)}, \boldsymbol{H})$.
5. Преобразовать $\boldsymbol{Q}'^{(4)}$: $\boldsymbol{Q}'^{(4)} := \boldsymbol{Encrypt}(m, \boldsymbol{Q}'^{(4)} \oplus \boldsymbol{Q}'^{(3)}, \boldsymbol{H})$.
6. Сформировать расширенный ключ:

   $\boldsymbol{Q} := \{\boldsymbol{Q}'^{(1)}, \boldsymbol{Q}'^{(2)}, \boldsymbol{Q}'^{(3)}, \boldsymbol{Q}'^{(4)}, q'_{2048}, q'_{2049}, q'_{2050}\}$.
7. **СТОП**.

Таким образом, расширенный ключ шифрования $\boldsymbol{Q}$ представляет собой последовательность из 2051 байта: $\boldsymbol{Q} = \{q_0, q_1, q_2, \ldots, q_{2047}, q_{2048}, q_{2049}, q_{2050}\}$.

Процедура зашифрования *Encrypt*

Процедура $\boldsymbol{Encrypt}(m, T, Q)$ является одной из двух программ резидентного модуля, предназначенной для преобразования (зашифрования) данных. Процедура ***Encrypt*** зависит от трех параметров, первый из которых определяет длину преобразуемого блока данных (выраженную в 32-разрядных словах), второй — собственно преобразуемый блок данных, а третий параметр — расширенный ключ.

Итак, длина блока данных, подлежащих преобразованию, определяется параметром m ($m \geq 4$). Значение числа m выбирается в зависимости от области применения. Например, данное значение может определяться размером сектора на жестком диске или размером буфера обмена информации. В частности, для $m = 128$ длина блока преобразуемых данных (в байтах) равна $4m$, т. е. равна 512 байтам.

При зашифровании используются 32-разрядные слова (подключи) следующего вида:

$$Q_j = q_j \parallel q_{j+1} \parallel q_{j+2} \parallel q_{j+3},$$

где $j = 0, 1, \ldots, 2047$.

Рис. 5.22. Очередность преобразования 32-разрядных слов в шифре SPECTR-SZ

Процедура ***Encrypt*** включает два полных и четыре сокращенных раунда преобразований (рис. 5.22). В случае $m = 4$ полный и сокращенный раунды идентичны.

Для заданного m преобразуемый блок данных является последовательностью m 32-разрядных слов, а именно:

$$\boldsymbol{T} = \{T_0, T_1, T_2, \ldots, T_{m-1}\},$$

где $m \geq 4$ и $\boldsymbol{T} \in \mathbf{B}^{4m}$.

В каждом раунде выполняется последовательное преобразование слов T_0, T_1, …, T_{m-1}. После каждого раунда, исключая последний раунд, выполняются две операции: $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.

В раундовых преобразованиях используются две следующие типовые процедуры:

Процедура ***Initialize***: {Установить $i := 0$; $R := Q_9$; $V := Q_{17}$; $N := Q_{31}$}.

Процедура ***ChangeNVR***: $\{ \quad N := \mathbf{P}_{32/32}^{(R,0)}(N) \oplus R\,;$

$n := N +_{11} 0;$

$V := \mathbf{P}_{32/32}^{(N,1)}(V) +_{32} Q_n \quad \}.$

Первый (полный) раунд зашифрования

Алгоритм выполнения первого (полного) раунда зашифрования имеет следующий вид.

1. Выполнить процедуру ***Initialize***.
2. Выполнить процедуру ***ChangeNVR***.
3. Выполнить преобразование $T_i := \mathbf{P}_{32/32}^{(R,0)}(T_i -_{32} V)$.
4. Преобразовать переменную R: $R := \mathbf{P}_{32/32}^{(V,1)}(R +_{32} T_i)$.
5. Выполнить преобразование $T_i := T_i +_{32} N$.
6. Увеличить значение счетчика $i := i + 1$.

   Если $i \neq m$, тогда перейти к шагу 2.
7. Если $m > 4$, тогда поочередно преобразовать слова T_2 и T_3:

   $T_2 := T_2 \oplus T_{m-2}$; $T_3 := T_3 \oplus T_{m-1}$.
8. Выполнить операции $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. **СТОП**.

Четыре сокращенных раунда

Алгоритм выполнения четырех сокращенных раундов имеет следующий вид:

1. Установить начальное значение для счетчика числа сокращенных раундов $j := 0$.
2. Выполнить процедуру ***Initialize***.
3. Выполнить процедуру ***ChangeNVR***.
4. Выполнить преобразование $T_i := \mathbf{P}_{32/32}^{(R,0)}(T_i -_{32} V)$.
5. Преобразовать переменную R: $R := \mathbf{P}_{32/32}^{(V,1)}(R +_{32} T_i)$.

6. Выполнить преобразование $T_i := T_i +_{32} N$.
7. Увеличить значение счетчика $i := i + 1$.

   Если $i \neq 4$, тогда перейти к шагу 3.
8. Выполнить операции $T_0 \leftrightarrow T_3$ и $T_1 \leftrightarrow T_2$.
9. Увеличить значение счетчика $j := j + 1$.

   Если $j \neq 4$, тогда перейти к шагу 2.
10. **СТОП**.

Шестой (полный) раунд зашифрования

Алгоритм выполнения шестого (полного) раунда зашифрования имеет следующий вид:

1. Выполнить процедуру ***Initialize***.
2. Если $m > 4$, тогда поочередно преобразовать слова T_3 и T_2:

   $T_3 := T_3 \oplus T_{m-1}$ и $T_2 := T_2 \oplus T_{m-2}$.
3. Выполнить процедуру ***ChangeNVR***.
4. Выполнить преобразование $T_i := \mathbf{P}_{32/32}^{(R,0)}(T_i -_{32} V)$.
5. Преобразовать переменную R: $R := \mathbf{P}_{32/32}^{(V,1)}(R +_{32} T_i)$.
6. Преобразовать слово T_i $T_i := T_i +_{32} N$.
7. Увеличить значение счетчика $i := i + 1$.

   Если $i \neq m$, тогда перейти к шагу 2.
8. **СТОП.**

Процедура расшифрования *Decrypt*

Процедура ***Decrypt***(m, T, Q) является второй программой резидентного модуля, предназначенной для восстановления (расшифрования) преобразованных данных. Процедура по отношению к процедуре ***Encrypt*** выполняет обратное преобразование. Описание процедуры не приводится, однако заинтересованный читатель может самостоятельно сформировать такую процедуру, учитывая, что очередность преобразования слов такая же, что и в случае зашифрования (см. рис. 5.22).

Скоростные параметры и криптографическая стойкость шифра

Поскольку в общем случае только часть входного блока данных преобразуется в соответствии с двумя полными раундами, скорость шифрования зависит от размера входного блока. При $m = 4$ она минимальна и составляет около 200 Мбит/с (для процессора типа Celeron 500 МГц в предположении, что в нем в виде специ-

альной инструкции реализована команда ***DDP32***). С увеличением размера блока скорость увеличивается до 600 Мбит/с при $m \geq 32$.

По сравнению с программными алгоритмами, рассмотренными в *гл. 2*, в шифре SPECTR-SZ более высокая скорость шифрования обеспечивается не только за счет сокращения общего числа выполняемых операций, но и меньшим количеством обращений к памяти (с пяти до трех).

Необходимая практическая стойкость алгоритма достигается благодаря следующим особенностям алгоритма:

1. Большая длина расширенного ключа $\boldsymbol{Q}$ ($\boldsymbol{Q} \in \mathbf{B}^{2051}$).
2. Использование на каждом этапе преобразования 32-разрядных слов T_i трех 32-разрядных переменных N, V и R, каждая из которых зависит как от расширенного ключа $\boldsymbol{Q}$, так и от уже преобразованных входных данных.
3. Обеспечение выполнения критерия строгого лавинного эффекта как операциями выборки подключей Q_j, так и использованием входных (преобразуемых) данных для формирования управляющих векторов для новой команды ***DDP32***.

Поясним второй пункт. Процедура ***Encrypt*** составлена таким образом, что для двух блоков данных $\boldsymbol{T}$ и $\boldsymbol{T'}$, различающихся в i-м слове ($T_i \neq T_i'$), уже в первом раунде при преобразовании слов T_i и T_i' значения переменных R и R' различны. На следующем шаге различие между переменными R и R' порождает различие, соответственно, между переменными N и N' и между V и V', а следовательно, и различие между T_{i+1} и T'_{i+1}. Соответственно, все последующие значения 32-разрядных слов T_j и T_j', вообще говоря, также различны, причем при переходе к последующим преобразуемым словам изменения имеют лавинообразный характер. Преобразования последующих раундов обеспечивают распространение различий на каждое 32-разрядное слово преобразуемых данных.

Благодаря большому числу шагов преобразования, выполняемых в одном раунде, сильный лавинный эффект достигается осуществлением всего двух полных раундов шифрования.

Сокращенные раунды шифрования предназначены обеспечить реализацию сильного лавинного эффекта при появлении различий только в последнем слове T_{m-1}.

Операция ***DDP32***, управляющий вектор которой зависит от преобразуемого блока данных, вносит существенный вклад в распространение лавинного эффекта при преобразовании слов T_i и переменной V.

В соответствии с оценками, полученными для комбинаторно-вероятностной модели, рассмотренной в *гл. 2*, стойкость шифра SPECTR-SZ составляет не менее 2^{190} (2^{95}) операций, при условии наличия у атакующего специально подобранного входного текста объемом не менее 2^{100} (2^{50}) байтов.

Возможность параметрического задания размера преобразуемого блока данных расширяет диапазон практического использования блочного шифра SPECTR-SZ и свидетельствует о его универсальности.

Говоря о перспективности новой команды ***DDP32***, необходимо отметить, что ее можно использовать и для создания скоростных программных хэш-функций. При этом, по сравнению с быстродействием шифра SPECTR-SZ, легко достигается более высокая скорость хэширования, поскольку алгоритмы хэширования могут быть составлены таким образом, чтобы осуществлялось только одно обращение к памяти. С использованием операции ***DDP32*** можно обеспечить скорость хэширования 1—2 Гбит/с. Для сравнения — в SPECTR-SZ используются три операции обращения к памяти в расчете на одно 32-разрядное преобразуемое слово, а именно:

- считывание текущего преобразуемого слова;
- считывание одного подключа;
- запись преобразованного слова.

Поскольку операция ***DDP32*** ориентирована на применения в криптографических приложениях, существуют объективные условия создания скоростных алгоритмов шифрования и хэширования, обладающих необходимой практической стойкостью.

5.5.3. Блочные шифры COBRA-F64a и COBRA-F64b

Рассмотрим варианты построения микропрограммных блочных шифров с малым размером входного блока данных. Такие шифры могут быть применены, например, в микроконтроллерах и интеллектуальных электронных карточках, для которых необходимо максимизировать отношение производительности шифрования к количеству используемых активных элементов электронных схем.

Ниже приведено описание двух итеративных блочных шифров COBRA-F64a и COBRA-F64b с 64-разрядным входом и 128-разрядным секретным ключом, ориентированных на микропрограммную (firmware) реализацию. Описание алгоритмов приведено в соответствии с обозначениями, принятыми для блочных шифров.

Общая схема шифрования

В блочных шифрах COBRA-F64a и COBRA-F64b используется схема шифрования, которая не является универсальной в принятой ранее терминологии. Напомним, что универсальной считается схема, в которой изменение режима шифрования (зашифрование и расшифрование) влечет лишь изменение очередности использования раундовых ключей. Однако схему шифрования алгоритмов COBRA-F64a и COBRA-F64b можно назвать полууниверсальной, поскольку все отличие заключается в использовании новой команды $\mathbf{P}_{32/32}^{(U,e)}$, зависящей от параметра e. Используя верхний индекс "$^{(e)}$", общая схема шифрования может быть задана следующей формулой:

$$Y = \mathbf{F}^{(e)}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от 128-разрядного ключа K и режима шифрования e ($\mathbf{F}^{(0)}$ — зашифрование, $\mathbf{F}^{(1)}$ — расшифрование), причем в режиме зашифрования X ($X \in \mathrm{GF}(2)^{32}$) является исходным блоком двоичных данных (открытым текстом), а в режиме расшифрования — преобразованным блоком двоичных данных (шифртекстом). В режиме зашифрования результирующее значение Y ($Y \in \mathrm{GF}(2)^{32}$) является шифртекстом, а в режиме расшифрования — открытым текстом.

С учетом современных принципов разработки блочных шифров расширенный ключ $Q^{(e)} = \mathbf{H}(K, e)$ необходимо формировать с использованием некоторой криптографически стойкой процедуры генерации расширенного ключа. Фактически достаточно сформировать некоторую псевдослучайную последовательность $S = Q^{(0)} = \mathbf{H}(K, 0)$, длина которой зависит от числа раундов и суммарной длины ключа одного раунда.

Расширенный ключ $Q^{(1)}$ является перестановкой элементов расширенного ключа $Q^{(0)}$, причем такая перестановка определяется конструктивными особенностями схемы, включая начальное и заключительное преобразования.

Обобщенная схема преобразований представлена на рис. 5.23, а схема одного раунда для алгоритмов COBRA-F64a и COBRA-F64b — на рис. 5.24.

Рис. 5.23. Общая схема шифрования в алгоритмах COBRA-F64a, b

Рис. 5.24. Процедура шифрования $\boldsymbol{Crypt}^{(e)}$ в алгоритмах:
а — COBRA-F64a и *б* — COBRA-F64b

Особенностью общей схемы шифрования является отсутствие начального преобразования. Однако зашифрование и расшифрование выполняются с использованием одного и того же алгоритма, поскольку структура каждого раунда такова, что заключительное преобразование является составной его частью.

Расписание использования раундовых ключей

В алгоритмах COBRA-F64a и COBRA-F64b, как в одном раунде, так и в заключительном преобразовании, используются два 32-разрядных подключа, а именно:

$$Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}),$$

где $1 \le j \le r + 1$ и r — число раундов шифрования.

С учетом целевого назначения алгоритмов — применение в микроконтроллерах и интеллектуальных электронных карточках, для которых объем занимаемой памяти является критическим параметром, использование стойкой процедуры генерации расширенного ключа вряд ли оправдано. Более экономичным вариантом является использование расписания ключей.

Пусть, например, секретный ключ K представлен в виде объединения четырех 32-разрядных слов, а именно: $K = K_0 \| K_1 \| K_2 \| K_3$, где и $K_0, K_1, K_2, K_3 \in \mathrm{GF}(2)^{32}$, а расширенный ключ $Q^{(0)} = S$ — табл. 5.14.

Тогда значения раундовых ключей $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)})$ для режимов зашифрования и расшифрования определяются табл. 5.14 и следующей формулой:

$$(Q_j^{(i,e)}, Q_{r+1-j}^{(i,e)}) = \mathbf{P}_{2\times 32/1}^{(e)}(S_j^{(1)}, S_{r+1-j}^{(2)}),$$

где $i = 1,2$ и $j = 1, 2, \ldots, r$.

Таблица 5.14. *Расписание использования раундовых ключей*

$j =$	1	2	3	4	5	6	7	8	9	10	11	12
$S_j^{(1)} =$	K_1	K_2	K_3	K_4	K_2	K_1	K_4	K_3	K_1	K_2	K_4	K_3
$S_j^{(2)} =$	K_4	K_3	K_1	K_2	K_3	K_2	K_1	K_4	K_2	K_3	K_1	K_2

$j =$	13	14	15	16	17	18	19	20	21	22	23	24
$S_j^{(1)} =$	K_1	K_4	K_2	K_3	K_1	K_4	K_3	K_1	K_2			
$S_j^{(2)} =$	K_3	K_1	K_3	K_4	K_2	K_1	K_4	K_2	K_3			

С учетом криптографических свойств используемых операций, а также особенностей конструкции одного раунда, в алгоритме COBRA-F64a оптимальным является число раундов шифрования, равное 16, а для алгоритма COBRA-F64b — равное 20. Тем не менее, при необходимости повысить нижнюю границу криптографической стойкости шифра, можно увеличить число раундов шифрования.

Алгоритм шифрования

Функция **F** (см. рис. 5.5) реализована в виде последовательности следующих процедур:

1. r раундов (циклов) *преобразований* с использованием процедуры $\boldsymbol{Crypt}^{(e)}$.
2. Заключительного преобразования ***FT***.

Первоначально блок X делится на два подблока L_0 и R_0 одинаковой длины, т. е. $(L_0, R_0) = X$, где $L_0, R_0 \in \mathrm{GF}(2)^{32}$. Затем выполняются r раундов преобразований с использованием процедуры $\boldsymbol{Crypt}^{(e)}$ в соответствии с формулами:

$$L_j = \boldsymbol{Crypt}(R_{j-1}, L_{j-1}, Q_j^{(e)});$$

$$R_j = L_{j-1}, \qquad (j = 1, 2, \ldots, r).$$

После r-го раунда над блоком $X = (R_r, L_r)$ выполняется заключительное преобразование ***FT*** в соответствии с формулой:

$$Y = \boldsymbol{FT}(X, Q_{FT}^{(e)}).$$

Заключительное преобразование ***FT*** для алгоритма COBRA-F64a имеет вид:

$$Y = (L_{r+1}, R_{r+1}) = (R_r -_{32} Q_{r+1}^{(1,e)}, L_r +_{32} Q_{r+1}^{(2,e)}).$$

Соответственно, для алгоритма COBRA-F64b преобразование ***FT*** имеет вид:

$$Y = (L_{r+1}, R_{r+1}) = (R_r \oplus Q_{r+1}^{(1,e)}, L_r \oplus Q_{r+1}^{(2,e)}).$$

Скоростные параметры и криптографическая стойкость шифров

Достоинством алгоритмов COBRA-F64a,b является малая длина двоичного кода при их программной реализации, что существенно важно, например, при проектировании криптосистем, встроенных в процедуру защищенной начальной загрузки ЭВМ. Микропрограммная реализация этих алгоритмов с использованием микроконтроллеров, работающих на тактовой частоте 33 МГц, обеспечивает производительность более 20 Мбит/с.

При программной реализации алгоритмов COBRA-F64a и COBRA-F64b (с использованием процессора типа Celeron 500 МГц в предположении, что в нем в виде специальной инструкции реализована команда ***DDP32***) предполагаемая скорость шифрования составит примерно 400 и 300 Мбит/с, соответственно.

Рассмотренные примеры свидетельствуют, что программные шифры с использованием управляемых перестановок, выполняемых в зависимости от данных, по сравнению с наиболее быстрыми из известных в настоящее время шифров, потенциально обеспечивают увеличение скорости в несколько раз.

Проведение исследований по оценке криптографической стойкости шифров ко всем известным методам криптоанализа является сложной и дорогостоящей процедурой. Поэтом такие исследования выполняются по отношению к наиболее эффективным методам криптоанализа. Метод дифференциального криптоанализа — один из них.

В методе дифференциального криптоанализа *(более подробно данный вопрос рассматривается в гл. 6)* исследуются разности пар преобразуемых блоков данных X и X', а именно $\Delta(X, X')$, и соответствующие разности преобразованных блоков $\delta(Y, Y')$, причем обычно выбирают случай, когда $\Delta(X, X') = X \oplus X' = \Delta = \text{const}$. Значение Δ называется *входной разностью*. Ненулевые разряды рассматриваемой разности Δ называются *активными битами*. Задачей дифференциального криптоанализа является нахождение таких значений Δ и δ, для которых выполняется неравенство $p(\delta/\Delta) > 2^{-n}$, где n — длина входного блока в битах, а $p(\delta/\Delta) = \Pr(\delta/\Delta)$ — условная вероятность. Обычно "значимые вероятности" $p(\delta/\Delta)$, т. е. максимально отличающиеся от средних значений, для рассматриваемого нами дифференциального криптоанализа относятся к разностям с малым числом активных битов. В *гл. 6* дифференциальный криптоанализ шифров COBRA-F64a, COBRA-F64b, DDP-S64 и DDP-S128 рассматривается более подробно, а в данной главе для каждого шифра приводятся лишь краткие результаты.

В результате исследования криптографической стойкости шифров COBRA-F64a,b к методу дифференциального криптоанализа было установлено, что наиболее значимые вероятности соответствуют дифференциальным характеристикам с разностями, включающими только один "активный бит". Напомним, что термин "активный бит" используется в том смысле, что вес Хэмминга разности двух рассматриваемых векторов равен единице. Так, для алгоритма COBRA-F64a лучшей среди найденных характеристик является трехраундовая характеристика, для которой однобитовой входной разности Δ_1 после трех раундов преобразований соответствует выходная однобитовая разность δ_1 с вероятностью $p_a(3) = 2^{-21}$. Соответственно, для алгоритма COBRA-F64b лучшей среди найденных характеристик является двухраундовая характеристика, для которой входной разности Δ_1 с одним активным битом в правом (или левом) блоке данных после двух раундов преобразований соответствует выходная однобитовая разность δ_1 с вероятностью $p_b(2) = 2^{-12}$. Однако итерационное применение данных характеристик в первом случае на 15 раундов, а во втором случае на 20 раундов не позволяет сформировать эффективные 15- и 20-раундовые характеристики, поскольку значения вероятностей $p_a(15) = 2^{-105}$ и $p_b(20) = 2^{-120}$ не превышают значения соответствующей вероятности в случайном 64-разрядном блочном шифре. Разработка дифференциальных характеристик с большим числом активных битов желаемых результатов получения эффективной характеристики также не дала, что позволяет говорить о высокой криптографической стойкости данных шифров к дифференциальному криптоанализу.

Высокие показатели скорости и криптографической стойкости предложенных шифров, в которых используется команда ***DDP32***, позволяют надеяться, что производителями микропроцессоров данная команда будет включена в набор стандартных операций универсальных процессоров.

О достоинствах использования программных шифров уже говорилось в предыдущих главах. Поэтому отметим лишь гибкость процедуры создания программного шифра. Действительно, встраивание на схемном уровне в процессоры некоторого, пусть даже общепризнанного алгоритма шифрования, фактически приводит к отсутствию выбора у потребителей данного вида продукции. Такая унификация дополнительно стимулирует разработку злоумышленниками конкретных методов вскрытия шифра, поскольку в случае успеха открываются широкие возможности несанкционированного доступа к информации в компьютерных системах, использующих данный шифр для защиты информации. Экономически выгодно анализировать один шифр, но не десять и, тем более, не сто.

Программная реализация новой команды (криптографического примитива) требует лишь минимальных схемотехнических затрат и практически не приводит к удорожанию модернизированных процессоров. Однако при этом расширяются возможности создания целой серии скоростных алгоритмов. Тогда, в случае компрометации отдельного алгоритма, его легко заменить другим, что невозможно осуществить для аппаратной реализации.

Как уже отмечалось выше, управляемые битовые перестановки общего типа могут быть использованы и для решения других актуальных задач защиты информации, таких как разработка скоростных алгоритмов целостности данных и хэш-функций. Другим перспективным направлением применения команды управляемых битовых перестановок является ее включение на микропрограммном уровне, что обеспечивает минимизацию используемых схемных ресурсов при обеспечении высокой скорости криптографических преобразований, осуществляемых как для контроля целостности данных, так и для защиты информации (хэширования и шифрования).

5.5.4. Алгоритмы DDP-S64 и DDP-S128

Рассмотренные выше шифры COBRA-F64a и COBRA-F64b ориентированы как на программную, так и на микропрограммную реализацию, что достигнуто использованием простой структуры преобразования. При ориентации только на программную реализацию можно применить несколько более сложные преобразования.

Рассмотрим 64- и 128-битовый программные шифры DDP-S64 и DDP-S128, в которых также используется команда управляемой перестановки ***DDP32***.

Общая схема шифрования

Как и в блочных шифрах COBRA-F64a, b, в программных шифрах DDP-S64 и DDP-S128 используется полууниверсальная схема шифрования, причем ее отличие от универсальной схемы также обусловлено использованием новой команды $\mathbf{P}_{32/32}^{(U,e)}$.

Общая схема шифрования может быть задана следующей формулой:

$$Y = \mathbf{F}^{(e)}(X, Q^{(e)}),$$

где $Q^{(e)} = \mathbf{H}(K, e)$ — *расширенный ключ*, являющийся функцией от 256-разрядного ключа K и режима шифрования e ($\mathbf{F}^{(0)}$ — зашифрование, $\mathbf{F}^{(1)}$ — расшифрование), причем в режиме зашифрования X является исходным блоком двоичных данных (открытым текстом), а в режиме расшифрования — преобразованным блоком двоичных данных (шифртекстом). В режиме зашифрования результирующее значение Y является шифртекстом, а в режиме расшифрования — открытым текстом.

Все ранее сказанное в отношении необходимости использовать некоторую криптографически стойкую процедуру генерации расширенного ключа применимо и для шифров, рассматриваемых в данном разделе. Так, достаточно сформировать некоторую псевдослучайную последовательность $S = Q^{(0)} = \mathbf{H}(K, 0)$, длина которой зависит от числа раундов и суммарной длины ключа одного раунда, а расширенный ключ $Q^{(1)}$ получается перестановкой элементов расширенного ключа $Q^{(0)}$, причем такая перестановка однозначно

причем такая перестановка однозначно определяется конструктивными особенностями схемы, включая начальное и заключительное преобразования. В качестве процедуры генерации расширенного ключа можно использовать, например, построенные на основе управляемых перестановок алгоритмы, приведенные в статье [11].

При описании алгоритмов рассматривается случай использования расписания ключей, что особенно эффективно для режима частой смены ключа.

64-битовый шифр DDP-S64

Обобщенная схема шифрования представлена на рис. 5.25.

Рис. 5.25. Общая схема шифрования (*б*) и один раунд преобразований (*а*) в алгоритме DDP-S64

Функция $\mathbf{F}^{(e)}$ (рис. 5.25, *б*) реализована в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. r раундов (циклов) *преобразований* с использованием процедуры $\boldsymbol{Crypt}^{(e)}$.
3. Заключительного преобразования ***FT***.

Первоначально блок X делится на два подблока X_{lo} и X_{hi} одинаковой длины, т. е. $(X_{lo}, X_{hi}) = X$, где $X_{lo}, X_{hi} \in \mathrm{GF}(2)^{32}$ и выполняется процедура ***IT***:

$$(A_0, B_0) = \boldsymbol{IT}(X, Q_0^{(e)}) = (X_{lo} \oplus Q_0^{(1,e)}, X_{hi} \oplus Q_0^{(2,e)}).$$

Затем выполняются $r = 10$ раундов преобразований с использованием процедуры $\boldsymbol{Crypt}^{(e)}$ в соответствии с формулами:

$$A_j = \boldsymbol{Crypt}^{(e)}(B_{j-1}, A_{j-1}, Q_j^{(e)});$$

$$B_j = A_{j-1}, \qquad (j = 1, 2, \ldots, r).$$

После r-го раунда над блоком $X = (B_r, A_r)$ выполняется заключительное преобразование $\boldsymbol{FT}$ в соответствии с формулой:

$$Y = \boldsymbol{FT}(X, Q_{r+1}^{(e)}) = (B_r \oplus Q_{r+1}^{(1,e)}, A_r \oplus Q_{r+1}^{(2,e)}).$$

Один раунд преобразований представлен на рис. 5.25, *а*.

В процедуре $\boldsymbol{Crypt}^{(e)}$ операция управляемой перестановки $\mathbf{P}_{32/32}^{(U,e)}$ обозначена как $\mathbf{P}^{(U,e)}$, а сам алгоритм имеет следующий вид:

1. $U := B +_{32} Q_i^{(2,e)}$.
2. $A := A +_{32} \mathbf{P}^{(U,0)}(B)$.
3. $V := \mathbf{P}^{(U,e)}(A)$.
4. $B := \mathbf{P}^{(V,e)}(U)$.
5. $A := V -_{32} Q_i^{(1,e)}$.
6. $B := B -_{32} \mathbf{P}^{(V,0)}(A)$.
7. $(A, B) := (B, A)$.
8. **СТОП**.

Расписание использования подключей в шифре DDP-S64 приведено в табл. 5.15

Таблица 5.15. *Расписание использования подключей в шифре DDP-S64*

$i =$	0	1	2	3	4	5	6	7	8	9	10	11
$Q_i^{(1,1)} =$	K_6	K_1	K_2	K_3	K_4	K_8	K_7	K_5	K_6	K_7	K_8	K_7
$Q_i^{(2,1)} =$	K_4	K_5	K_6	K_7	K_8	K_3	K_5	K_2	K_1	K_4	K_3	K_5
$Q_i^{(1,0)} =$	K_7	K_3	K_4	K_1	K_2	K_5	K_3	K_8	K_7	K_6	K_5	K_6
$Q_i^{(2,0)} =$	K_5	K_8	K_7	K_6	K_5	K_7	K_8	K_4	K_3	K_2	K_1	K_4

Оценка стойкости шифра DDP-S64

В методе дифференциального криптоанализа для данной модели наиболее эффективными являются характеристики, связанные с рассмотрением входных разностей Δ_1 (с одним активным битом в левом подблоке данных) через два раунда. Такие разности после двух раундов трансформируются в разность δ_1 (в которой активный бит также принадлежит левому подблоку) с вероятностью 2^{-23}. При этом на выходе первого раунда формируется разность δ_2' (причем каждый из подблоков содержит по одному активному биту), которая при прохождении второго раунда превращается в однобитовую разность δ_1. Рассматривая структуру раундового преобразования для DDP-S64, оценим вероятность для первого раунда $p_1(\delta_2'/\Delta_1) \approx 2^{-9}$, а для второго раунда — $p_2(\delta_1/\delta_2') \approx 2^{-14}$. Соответственно, для двух раундов вероятность — $p(2) = p(\delta_1/\Delta_1) = p_1 p_2 \approx 2^{-23}$. Однако последующее формирование характеристик шести и восьми раундов не может привести к желаемому результату, поскольку в этом случае вероятности $p(6) = p(\delta_1/\Delta_1) \approx 2^{-69}$ и $p(8) = p(\delta_1/\Delta_1) \approx 2^{-92}$ не превышают значения соответствующей вероятности в случайном 64-разрядном блочном шифре. Разработка дифференциальных характеристик с большим числом активных битов желаемых результатов получения эффективной характеристики также не дала, что позволяет говорить о высокой криптографической стойкости блочного шифра DDP-S64 к дифференциальному криптоанализу.

128-битовый шифр DDP-S128

Обобщенная схема шифрования представлена на рис. 5.26, *б*.

В 128-битовом шифре DDP-S128 предусмотрено12 раундов шифрования ($r = 12$), причем, с учетом ориентации на программную реализацию, входной 128-битовый блок данных разбивается на четыре 32-битовых подблока A, B, C и D , которые и подвергаются дальнейшему преобразованию.

Увеличение (по сравнению с DDP-S64) числа раундов шифрования обусловлено тем фактом, что в одном раунде шифрования два из четырех подблоков данных остаются неизменными. Отличительной чертой алгоритма DDP-S128 является то, что в процедуре $\boldsymbol{Crypt}^{(e)}$ (рис. 5.26, *а*) над каждым из подблоков A и D осуществляются две управляемые перестановки, одна из которых зависит от подблока B, а другая от подблока C. Такая структура обеспечивает реализацию большого числа (около 2^{64}) перестановок, выполняемых над подблоками A и D в пределах одного раунда. При этом суперпозиция управляемых перестановок $\mathbf{P}_{32/32}^{(B,e)}$ и $\mathbf{P}_{32/32}^{(C,e)}$ является блоком максимального порядка.

Алгоритм блочного шифра DDP-S128 (рис. 5.26, *б*) реализован в виде последовательности следующих процедур:

1. Начального преобразования ***IT***.
2. r раундов (циклов) *преобразований* с использованием процедуры $\boldsymbol{Crypt}^{(e)}$.
3. Заключительного преобразования ***FT***.

Рис. 5.26. Общая схема шифрования (*б*) и один раунд преобразований (*а*) в алгоритме DDP-S128

В процедуре $\boldsymbol{Crypt}^{(e)}$ операция управляемой перестановки $\mathbf{P}_{32/32}^{(U,e)}$ обозначена как $\mathbf{P}^{(U,e)}$. Первоначально блок X делится на четыре подблока A, B, C, D одинаковой длины, т. е. $(A, B, C, D) = X$, где $A, B, C, D \in \mathrm{GF}(2)^{32}$, и выполняется процедура $\boldsymbol{IT}$:

$$(A_0, B_0, C_0, D_0) = \boldsymbol{IT}(A \parallel B \parallel C \parallel D, Q_{IT}^{(e)}) =$$
$$= (A \oplus Q_1^{(1,e)}, B \oplus Q_2^{(1,e)}, C \oplus Q_3^{(1,e)}, D \oplus Q_4^{(1,e)}).$$

Затем выполняются $r = 12$ раундов преобразований с использованием процедуры $\boldsymbol{Crypt}^{(e)}$ в соответствии с формулами:

$$(A, B, C, D) = \boldsymbol{Crypt}^{(e)}(A_{j-1}, B_{j-1}, C_{j-1}, D_{j-1}, Q_j^{(1,e)}, Q_j^{(2,e)});$$
$$(A_j, B_j, C_j, D_j) = (B, A, D, C) \quad (j = 1, 2, \ldots, r).$$

После r-го раунда над блоком $X = (B_r, A_r)$ выполняется заключительное преобразование $\boldsymbol{FT}$ в соответствии с формулой:

$$Y = \boldsymbol{FT}(B_r, A_r, D_r, C_r, Q_{FT}^{(e)}) =$$
$$= (B_r \oplus Q_r^{(2,e)}, A_r \oplus Q_{r-1}^{(2,e)}, D_r \oplus Q_{r-2}^{(2,e)}, C_r \oplus Q_{r-3}^{(2,e)}).$$

Процедура $\boldsymbol{Crypt}^{(e)}$ представлена на рис. 5.26, *а* и имеет следующий вид:

1. $V := B$.
2. $A := \mathbf{P}^{(B,0)}(A)$; $B := \mathbf{P}^{(C,1)}(B)$; $D := \mathbf{P}^{(C,0)}(D)$.
3. $C := C -_{32} Q_i^{(2,e)}$.
4. $A := A \oplus (B +_{32} C)$.
5. $B := B +_{32} Q_i^{(1,e)}$.
6. $C := \mathbf{P}^{(B,1)}(C)$.
7. $D := D \oplus (V +_{32} C)$.
8. $A := \mathbf{P}^{(C,0)}(A)$; $D := \mathbf{P}^{(C,1)}(D)$.
9. $(B, C) := (C, B)$.
10. **СТОП**.

Как и в предыдущих алгоритмах, применяется расписание ключей, причем в j-м раунде j (где $1 \le j \le r$) используются два 32-разрядных подключа, а именно:

$$Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)}),$$

расширенный ключ $Q^{(0)} = S$ представлен табл. 5.16, где 32-разрядные ключи $K_0, K_1, \ldots, K_7$ являются разбиением секретного ключа K ($K = K_0 \| K_1 \| \ldots \| K_7$), а значения раундовых ключей $Q_j^{(e)} = (Q_j^{(1,e)}, Q_j^{(2,e)})$ для режимов зашифрования и расшифрования определяются следующей формулой:

$$(Q_j^{(1+e,e)}, Q_{r+1-j}^{(2-e,e)}) = \mathbf{P}_{2\times 32/1}^{(e)}(S_j^{(1)}, S_{r+1-j}^{(2)}).$$

Таблица 5.16. *Расписание использования подключей в шифре DDP-S128*

$j =$	1	2	3	4	5	6	7	8	9	10	11	12
$S_j^{(1)} =$	K_0	K_1	K_2	K_3	K_4	K_5	K_6	K_7	K_0	K_2	K_4	K_6
$S_j^{(2)} =$	K_7	K_5	K_6	K_4	K_2	K_3	K_0	K_1	K_5	K_7	K_3	K_1

В начальном и заключительном преобразованиях алгоритма DDP-S128 используются по четыре 32-разрядных подключа, а именно:

$$Q_{IT}^{(e)} = (Q_1^{(1,e)}, Q_2^{(1,e)}, Q_3^{(1,e)}, Q_4^{(1,e)}),$$

$$Q_{FT}^{(e)} = (Q_r^{(2,e)}, Q_{r-1}^{(2,e)}, Q_{r-2}^{(2,e)}, Q_{r-3}^{(2,e)}),$$

где r — число раундов шифрования.

Необходимо отметить, что $\boldsymbol{Crypt}^{(e)}$ характеризуется следующими особенностями:

- управляемые перестановки над каждым из подблоков A и D выполняются в зависимости от подблоков B и C;
- подблок B преобразуется с использованием операций, являющихся взаимно обратными по сравнению с операциями, выполняемыми над подблоком C;
- суммы значений подблоков B и C, используемые при поразрядном сложении с подблоками A и D, различны;
- операция обмена значениями для подблоков B и C является элементом процедуры $\boldsymbol{Crypt}^{(e)}$.

Структура раундового преобразования шифра DDP-S128 удачно сочетает в себе идеи построения итеративных криптосхем, рассматривавшихся в *разд. 5.1*.

Оценка стойкости шифра DDP-S128

Для шифра DDP-S128 лучшими разностями дифференциального криптоанализа являются разности Δ_2 с двумя активными битами, проходящими поочередно по крайним и средним ветвям криптосхемы. По крайним ветвям единичные разности проходят без размножения с вероятностью $p_1 = 1$. Две единичные разности также могут пройти по средним ветвям без появления разности в крайних ветвях в случае, если в момент вычисления каждой из сумм $(B +_{32} C)$ (см. алгоритм процедуры $\boldsymbol{Crypt}^{(e)}$) единичные разности окажутся в одинаковых разрядах блоков B и C. В этом случае при вычислении каждой из сумм с вероятностью 0,5 разности для блоков B и C взаимно уничтожаются. Если обозначить вероятность этого события через p_2, тогда легко показать справедливость соотношения $p_2 \approx 2^{-32}$. Указанная выше разность проходит два раунда с вероятностью $p(2) = p_1 p_2 \approx 2^{-32}$, а восемь раундов — с вероятностью $p(8) = p_1 p_2 \approx 2^{-128}$.

Для десяти и двенадцати раундов такие вероятности имеют значение $p(10) \approx 2^{-160}$ и $p(12) \approx 2^{-192}$, соответственно. Таким образом, при числе раундов $r \geq 8$ можно утверждать, что блочный шифр DDP-S128 является стойким к дифференциальному криптоанализу.

5.6. Статистические свойства алгоритмов

5.6.1. Критерии оценки свойств "лавинного эффекта"

С целью проверки рассеивающих свойств блочных алгоритмов их тестирование целесообразно проводить по методике, предложенной членами Нового Европейского Проекта по созданию базовых примитивов, имея в виду возможную будущую стандартизацию (NESSIE, New European Schemes for Signature, Integrity and Encryption) [111].

В частности, для каждого разработанного алгоритма необходимо анализировать результаты статистической обработки по следующим критериям:

1. Среднее число битов выхода, изменяющихся при изменении одного бита входного вектора.
2. Степень полноты преобразования.
3. Степень лавинного эффекта.
4. Степень соответствия строгому лавинному критерию.

Пусть $U^{(i)} = U \oplus E_i$, т. е. бинарный вектор, полученный инвертированием i-го бита в векторе U. Тогда бинарный вектор $Y^{(i)} = \mathbf{F}(U^{(i)}) \oplus \mathbf{F}(U))$ называется *лавинным вектором* по компоненту i. Для блочного шифра $U = X \| K$. Пусть для рассматриваемых критериев размерность вектора U равна n, а вектора Y — равна m.

В критериях 2 и 4 используется матрица зависимостей $\|a_{ij}\|_{n\times m}$, а именно:

$$a_{ij} = \#\{Y^{(i)}, y_j^{(i)} = 1\}.$$

Матрица $\|a_{ij}\|_{n\times m}$ отражает зависимость j-го разряда выходного вектора от i-го разряда входного вектора. При этом *степень полноты преобразования* (критерий № 2) оценивается по формуле:

$$d_c = 1 - \frac{\#\{(i,j) \mid a_{ij} = 0\}}{nm}.$$

а *степень соответствия строгому лавинному критерию* (критерий № 4) — по формуле:

$$d_{sa} = 1 - \frac{\sum_{i=1}^{n} \sum_{j=1}^{m} \left| \frac{2a_{ij}}{N} - 1 \right|}{nm},$$

где $N = \#\mathbf{U} = \#\{U\}$.

Для получения точной оценки необходим полный перебор всех значений вектора U, однако, учитывая интегральный характер данных оценок, для получения приближенных значений достаточно использовать метод Монте-Карло, т. е. некоторую сравнительно небольшую выборку входных значений.

В критериях 1 и 3 используется матрица расстояний $\|b_{ij}\|_{n\times m}$, а именно:

$$b_{ij} = \#\{Y^{(i)} \mid w(y^{(i)}) = j\},$$

являющаяся маркировкой веса Хэмминга лавинных векторов.

Среднее число битов выхода, изменяющихся при изменении одного бита входа (критерий № 1), оценивается по формуле:

$$d_1 = \frac{1}{n} \sum_{i=1}^{n} \frac{\sum_{j=1}^{m} j b_{ij}}{N},$$

а *степень лавинного эффекта* (критерий № 3) — по формуле:

$$d_a = 1 - \frac{\sum_{i=1}^{n} \left| \frac{1}{N} \sum_{j=1}^{m} 2jb_{ij} - m \right|}{nm}.$$

В следующих двух разделах приведены результаты тестирования блочного шифра SPECTR-128. Результаты тестирования остальных алгоритмов получаются аналогичным образом.

5.6.2. Оценка влияния битов исходного текста на преобразованный текст

Критерии оценки влияния битов исходного текста на преобразованный текст предназначены для выявления возможных слабостей в алгоритме, которые могут быть использованы с применением методов криптографического анализа на основе выбранного исходного текста или методом дифференциального криптоанализа. Для приведенных в *разд. 5.6.1* критериев рассматривается случай, когда *лавинный вектор* $Y^{(i)}$ формируется по входным векторам $U = X \| K$ и $U = X^{(i)} \| K$, где $X^{(i)} = X \oplus E_i$.

В качестве параметров задаются значения: q — для ключей, t — для исходных текстов. $N = qt$. Выбор значений q и t зависит от значений n и m и имеющихся вычислительных ресурсов. *Матрица зависимостей* $\|a_{ij}\|_{n\times m}$ и *матрица расстояний* $\|b_{ij}\|_{n\times m}$ имеют вид:

$$a_{ij} = \#\{X \in \mathbf{X}, K \in \mathbf{K} \mid (\mathbf{F}(X^{(i)}, K))_j \neq (\mathbf{F}(X, K))_j\},$$

$$b_{ij} = \#\{X \in \mathbf{X}, K \in \mathbf{K} \mid w(\mathbf{F}(X^{(i)}, K) \oplus \mathbf{F}(X, K)) = j\}.$$

Ключи и исходные тексты необходимо формировать с использованием датчика случайных чисел.

Для алгоритма SPECTR-128 были получены интегральные оценки как с заявленным числом раундов, так и для сокращенного числа раундов. По аналогии с приведенными в [111] результатами тестирования пяти кандидатов — финалистов конкурса на новый американский стандарт блочного шифрования, проводившегося Национальным институтом Стандартов и Технологий США, был реализован вариант "один ключ и 10 000 текстов", а также вариант "100 ключей и 100 текстов" (табл. 5.17).

Результаты тестирования алгоритма SPECTR-128 соответствуют результатам тестирования финалистов конкурса и свидетельствуют, что предложенный алгоритм преобразования данных обладает хорошими рассеивающими свойствами даже при небольшом числе раундов и может рассматриваться как хороший генератор псевдослучайных подстановок даже при небольшом числе раундов. В ча-

стности, *критерий полноты*, согласно которому "каждый входной бит должен влиять на каждый выходной бит", выполняется уже после двух раундов шифрования. Для сравнения, в алгоритмах DES и ГОСТ 28147-89 данный критерий выполняется не менее чем через четыре раунда шифрования данных, что обусловлено исключительно используемой схемой Фейстеля.

Таблица 5.17. *Таблица значений критериев 1—4 влияния исходного текста на преобразованный текст (для различного числа раундов)*

Число раундов	#*K*=1, #*X*=10000;				#*K*=100, #*X*=100.			
	(1)	(2)=d_c	(3)=d_a	(4)=d_{sa}	(1)	(2)=d_c	(3)=d_a	(4)=d_{sa}
12	64.00145	1.00000	0.99922	0.99208	64.00385	1.00000	0.99937	0.99198
11	63.99543	1.00000	0.99922	0.99206	64.00783	1.00000	0.99929	0.99214
10	64.00466	1.00000	0.99933	0.99204	64.00295	1.00000	0.99930	0.99197
9	63.98921	1.00000	0.99932	0.99205	64.00405	1.00000	0.99921	0.99205
8	63.99262	1.00000	0.99935	0.99200	63.99932	1.00000	0.99928	0.99194
7	64.00420	1.00000	0.99927	0.99206	64.00502	1.00000	0.99926	0.99203
6	64.00168	1.00000	0.99929	0.99205	63.99895	1.00000	0.99937	0.99203
5	63.99244	1.00000	0.99931	0.99208	63.99628	1.00000	0.99930	0.99206
4	63.64805	1.00000	0.99421	0.98838	63.62012	1.00000	0.99369	0.98805
3	52.35455	1.00000	0.81804	0.81594	52.45568	1.00000	0.81962	0.81759
2	26.45824	0.98535	0.41341	0.41330	26.55646	1.00000	0.41494	0.41490
1	6.24359	0.48852	0.09755	0.08970	6.22903	0.50781	0.09732	0.09663

5.6.3. Оценка влияния битов ключа на преобразованный текст

В данном случае *лавинный вектор* $Y^{(i)}$ формируется по входным векторам $U = X \| K$ и $U = X \| K^{(i)}$, где $K^{(i)} = K \oplus E_i$. В критериях матрица зависимостей $\|a_{ij}\|_{n\times m}$ и матрица расстояний $\|b_{ij}\|_{n\times m}$ имеют следующий вид:

$$a_{ij} = \#\{X \in \mathbf{X}, K \in \mathbf{K} \mid (\mathbf{F}(X, K^{(i)}))_j \neq (\mathbf{F}(X, K))_j\},$$

$$b_{ij} = \#\{X \in \mathbf{X}, K \in \mathbf{K} \mid w(\mathbf{F}(X, K^{(i)}) \oplus \mathbf{F}(X, K)) = j\}.$$

Данные тестирования алгоритма SPECTR-128 свидетельствуют (табл. 5.18), что даже без использования стойкой процедуры генерации расширенного секретного ключа обеспечивается достаточно сильное рассеивающее влияние каждого бита ключа на все биты преобразованного текста.

Необходимо отметить, что рассматриваемые критерии являются эффективным инструментом для выявления слабостей как в разрабатываемых отдельных процедурах и преобразованиях, так и при составлении расписания использования раундовых ключей, а также при выборе оптимального числа раундов.

Таблица 5.18. *Таблица значений критериев 1—4 влияния битов ключа на преобразованный текст (для различного числа раундов)*

Число раундов	#*K*=400, #*X*=25;				#*K*=100, #*X*=100.			
	(1)	(2)=d_c	(3)=d_a	(4)=d_{sa}	(1)	(2)=d_c	(3)=d_a	(4)=d_{sa}
12	63.99562	1.00000	0.99931	0.99202	64.00289	1.00000	0.99931	0.99206
11	63.99997	1.00000	0.99923	0.99202	64.00450	1.00000	0.99928	0.99203
10	64.00301	1.00000	0.99932	0.99204	63.99632	1.00000	0.99930	0.99202
9	63.99707	1.00000	0.99930	0.99204	63.99466	1.00000	0.99923	0.99202
8	64.00112	1.00000	0.99930	0.99200	64.00076	1.00000	0.99926	0.99195
7	63.99715	1.00000	0.99932	0.99200	63.99873	1.00000	0.99932	0.99201
6	64.00123	1.00000	0.99928	0.99200	64.00549	1.00000	0.99927	0.99201
5	63.94757	1.00000	0.99885	0.99195	63.95256	1.00000	0.99891	0.99202
4	63.94757	1.00000	0.99070	0.98590	63.41481	1.00000	0.99072	0.98592
3	54.33583	1.00000	0.84899	0.84733	54.36625	1.00000	0.84947	0.84783
2	27.21150	1.00000	0.42518	0.42510	27.22301	1.00000	0.42536	0.42528
1	4.59536	0.50390	0.07180	0.07173	4.59477	0.50390	0.07179	0.07171

С учетом существующих современных рекомендаций число раундов в алгоритмах блочного шифрования должно быть на два больше минимальной безопасной границы, что и реализовано, в частности, в алгоритмах SPECTR-H64 и SPECTR-128.

5.6.4. Использование критерия χ^2 для уточнения интегральных оценок

В данном разделе рассматривается лишь один из семейства статистических критериев, позволяющих становить соответствие между теоретической и практической моделями.

Как и в случае рассмотрения критериев № 1 и № 3, оценки критериев № 2 и № 4 также являются интегральными. Однако по матрице $\|b_{ij}\|_{n\times m}$ можно дифференцированно оценить влияние каждого бита входного вектора. Для этого необходимо отдельно рассмотреть каждую строку матрицы $\|b_{ij}\|_{n\times m}$. Если блочный шифр, реа-

лизует псевдослучайную перестановку, т. е. обладает хорошими рассеивающими свойствами, распределение встречаемости веса Хэмминга лавинных векторов для каждой строки должно соответствовать теоретическому распределению расстояния Хэмминга между двумя случайно выбранными векторами A и B той же длины. Если $Y \in \mathrm{GF}(2)^m$, тогда вероятность того, что расстояние Хэмминга между векторами A и B равно j, вычисляется по формуле

$$p_j = \binom{m}{j} 2^{-m},$$

где $0 \le j \le m$.

Для оценки соответствия выборки теоретическому распределению можно использовать χ^2 *как критерий согласия* для двух простых гипотез.

Пусть для фиксированного i, $\Psi_i^{(N)}$ — выборка значений b_{ij}, полученная в результате проведения N испытаний. Гипотеза H_0 состоит в том, что распределение $\Psi_i^{(N)}$ соответствует теоретическому распределению χ^2 с $v - 1$ степенями свободы ($v = m + 1$). При этом для каждого i значение χ_i^2 вычисляется по формуле:

$$\chi_i^2 = \sum\nolimits_{j=0}^{m} (b_{ij})^2 / \mathrm{N}\, p_j - N\,.$$

Для заданного уровня значимости χ_p^2 гипотеза H_0 принимается, если $\chi_i^2 < \chi_p^2$, и отвергается в противном случае. Таким образом, в результате применения данного критерия можно сформировать бинарный вектор $M\chi_p^2$, причем значение i-го разряда равно единице, если принимается гипотеза H_0, и ноль — в противном случае. Результатом данного критерия является бинарный вектор $M\chi_p^2$ длины n (разрядность входа), в котором не должно быть нулей.

При реализации критерия нужно учитывать условие его применимости. Так, если $Np_\mathrm{j} \le 10$, то такие значения j необходимо объединять в группы. При этом число степеней свободы всегда меньше числа групп на единицу. Тестирование по данному критерию осуществлялось, в частности, для алгоритма SPECTR-H64, однако в книге результаты такого тестирования не приводятся.

Основные результаты

К числу основных результатов проектирования скоростных шифров на основе управляемых операций, рассмотренных в данной главе, можно отнести следующие:

- Представлены основные схемы конструирования универсальных итеративных блочных шифров на основе управляемых операций.
- Предложены конкретные модели блочных шифров для аппаратной и программной реализации.

- Показана перспективность управляемых подстановочных и перестановочных операций для их использования в качестве криптографических примитивов. В частности, для производителей процессоров предложена новая команда, реализующая управляемую операцию перестановки, которую, в случае ее реализации в виде специальной инструкции, можно эффективно использовать при разработке программных шифров.
- Предложены конкретные схемы и конструкции применения однослойных блоков управляемых перестановок для управления не только выбором раундовых ключей, но и при формировании управляющих векторов для реализации обратного преобразования в блоках управляемых перестановок специального вида.
- Поскольку блочные шифры могут обладать лишь практической стойкостью (но не теоретической), при их разработке необходимо неукоснительно следовать современным принципам разработки таких шифров. Приведенные в данной главе статистические тесты являются испытанным инструментом, позволяющим выявить имеющиеся очевидные слабости алгоритма.

ГЛАВА 6

Элементы криптоанализа шифров на базе управляемых операций

6.1. Об оценке гибких шифров

Как правило, криптоанализ базируется на использовании алгебраических и/или статистических закономерностей шифров. В случае выявления некоторых закономерностей, позволяющих отличить детерминированное преобразование, реализуемое шифром под управлением секретного ключа, от случайного, появляются реальные предпосылки для раскрытия шифра. Поэтому достаточно общей задачей криптоанализа является выявление таких закономерностей, а также разработка методов использования выявленных закономерностей для раскрытия шифра или некоторых его элементов.

Наиболее известным универсальным статистическим методом криптоанализа является *частотный криптоанализ*. Частотный криптоанализ базируется на использовании неравномерности встречаемости разных знаков алфавита или их комбинаций в исходном тексте, которая в результате применения шифрующего преобразования нивелируется, но не полностью. Так, например, если используется шифр простой замены (каждому символу алфавита исходного текста соответствует уникальный символ алфавита шифртекста), то в преобразованном тексте (шифртексте) полностью сохраняется структура частоты встречаемости символов исходного текста. При наличии шифртекста объема, достаточного для выявления существенных различий в частотах встречаемости различных его знаков, данная закономерность позволяет раскрыть шифр, поскольку имеется возможность установления соответствия между символами шифртекста и исходного текста.

Как известно, один из основных режимов использования блочных шифров — это режим простой замены (режим электронной кодовой книги). Однако современные блочные шифры практически неуязвимы к методу частотного криптоанализа ввиду больших значений мощности алфавитов исходного и преобразованного текстов (2^{64} или 2^{128}). То есть практически неосуществима задача получения

реальной частоты встречаемости двоичных блоков исходного текста и, тем более, значимой выборки для преобразованных текстов. Поэтому разрабатываемые методы криптоанализа современных блочных шифров учитывают закономерности, для выявления которых необходимо использовать не только преобразованные, но и исходные тексты, в том числе и специально подобранные тексты. Эти варианты атак позволяют построить более разнообразные статистические методы анализа.

Существуют две основных задачи криптоанализа шифра или шифрсистемы.

Первая задача — вычисление секретного ключа или нахождение способа бесключевого чтения шифртекстов решается злоумышленником (дешифровальщиком, криптоаналитиком) с целью доступа к информации. Это так называемая задача дешифрования. Заметим, что в процессе решения первой задачи криптоаналитик с целью определения перспективности разработки шифра должен получить верхнюю оценку стойкости шифра для каждого из известных ему методов криптоанализа с учетом условий их практической применимости и выбрать наиболее эффективный метод вскрытия шифрсистемы. То есть для выбранного метода криптоаналитик должен определить, например, максимальное число элементарных операций (если такая оценка определяется числом элементарных операций), которое необходимо выполнить, чтобы решить задачу криптоанализа.

Криптоаналитик получает эту оценку с учетом трудоемкости отличия алгоритма шифрования от случайного преобразования и трудоемкости вычисления ключа с использованием обнаруженных статистических закономерностей. При оценке стойкости, кроме трудоемкости решения задачи криптоанализа, часто определяется необходимая для реализации предлагаемого метода атаки память. Как правило, при использовании большой памяти количество необходимых операций существенно уменьшается. Общим требованием к современным шифрам является высокая вычислительная сложность криптоанализа (например, 10^{40} операций) при использовании большого объема памяти (например, 10^{20} байт).

Вторая задача — оценка стойкости шифра, выполняемая обычно разработчиком. Как правило, условия решения первой задачи являются менее благоприятными по сравнению с условиями решения второй. Однако вторая задача не является менее сложной, поскольку разработчику шифра необходимо получить нижнюю оценку стойкости разрабатываемого шифра для каждого из известных методов криптоанализа, в том числе и к методам, которые на момент разработки шифра невозможно реализовать на практике, но которые имеют принципиальное теоретическое обоснование. Разработчик криптосистемы старается получить оценку, соответствующую минимальному числу операций, требующихся для решения задачи криптоанализа при использовании лучшего алгоритма криптоанализа. Поскольку не всегда возможно указать лучший алгоритм криптоанализа, минимальная оценка дается для лучшего из известных алгоритмов криптоанализа. Разработчик также может оценить стойкость шифра как трудоемкость определения его отличия от случайного шифра.

Таким образом, оценки, полученные в ходе решения первой задачи, должны быть выше оценок, полученных при решении второй задачи. Однако на практике это

выполняется не всегда, поскольку многое зависит от субъективных условий решения каждой из задач.

Обычно при выполнении криптоанализа предполагается, что целостность шифратора обеспечена в достаточной степени, когда алгоритм при осуществлении шифрования не модифицируется, а атакующий не имеет возможности считывать информацию о ключе через какие-либо каналы утечки (наводки по цепи питания, побочные электромагнитные излучения, измерение времени вычислений, формирование аппаратных ошибок и др.). Однако в последнее время все чаще рассматриваются варианты атак, ориентированные на нарушение целостности шифратора. Это связано с все расширяющимися технологическими применениями шифрования, когда шифраторы используются для решения самых разнообразных задач в различных условиях эксплуатации. Заметим, что криптографический анализ шифров специального применения всегда включал рассмотрение стойкости в условиях нарушения целостности шифратора в той или иной степени. Активное рассмотрение таких вариантов атак началось на стадии массового применения шифрования как технологического элемента системы защиты информации в автоматизированных системах.

Особый интерес представляет вопрос оценки стойкости *гибких* шифров, т. е. шифров, в которых использование конкретного криптоалгоритма определяется секретным ключом. Отметим, что в данном типе шифров принцип Керкхоффа, согласно которому единственным неизвестным элементом является секретный ключ, можно обеспечить путем использования предвычислений, которые включают процедуру формирования алгоритма шифрования в зависимости от секретного ключа пользователя. В этом случае конкретный алгоритм шифрования является легко сменяемым элементом. Он автоматически изменяется при смене секретного ключа. При большом числе возможных модификаций криптоалгоритма (например, 10^{20}) достаточно трудно проанализировать каждую из них. Поэтому при построении гибких шифров разрабатывается некоторый базовый механизм, в котором зафиксированы определенные принципы, определяющие общие свойства всех модификаций. Как правило, разработчик вычисляет оценку стойкости для ограниченной совокупности алгоритмов, предполагая, что конкретная модификация алгоритма шифрования известна криптоаналитику. С целью уточнения нижней оценки стойкости шифра полезно также получить такую оценку для наиболее "слабой" модификации. Другой важный вариант анализа гибких алгоритмов предполагает, что ключ шифрования известен, а неизвестна только модификация криптоалгоритма. Разумно применять только такие гибкие шифры, которые обладают высокой стойкостью ко всем перечисленным вариантам криптоанализа.

Секретность модификации криптоалгоритма в гибких шифрах должна рассматриваться не как главный фактор обеспечения высокой стойкости, а как механизм, определяющий дополнительный запас стойкости. Действительно, если криптоаналитик не знает модификации криптоалгоритма, с помощью которой выпол-

няются криптографические процедуры, то он сталкивается со значительно более сложной криптоаналитической задачей.

Универсальными методами криптоанализа гибких шифров являются метод бесключевого чтения и перебор по пространству возможных ключей. Эти методы не требуют знания конкретной модификации криптоалгоритма, однако первый из них эффективен только при раскрытии достаточно слабых шифров, а второй — в случае ключей малого размера.

Для гибких программных шифров, представленных в *гл. 2*, использование данных универсальных методов не позволяет осуществить практическое раскрытие криптосистемы. Использование более эффективных методов, основанных на наличии статистических особенностей в выборках, полученных при исследовании отдельных преобразований, имеет существенно более высокую трудоемкость в случае неизвестной модификации криптоалгоритма по сравнению со случаем полностью известных процедур преобразования.

Для получения минимальной оценки стойкости шифров, основанных на выборке ключей шифрования в зависимости от преобразуемых данных, была предложена *комбинаторно-вероятностная модель*. Описание и методы использования этой модели приведены в *гл. 2* применительно к оценке стойкости некоторых программных 512-байтовых алгоритмов. Основанием применения данной модели для получения минимальных оценок стойкости является то обстоятельство, что один раунд шифрования включает большое число операций ($128k$, где $k = 2, 4, 6$), из которых, по крайней мере, половина осуществляется в сложной зависимости от каждого бита входных данных и каждого бита секретного ключа. При этом выбираемые подключи непосредственно при преобразовании текущих подблоков данных не используются, а являются элементами процедуры формирования аккумулирующих ключевых переменных.

Эти переменные преобразуются в пределах одного раунда в режиме сцепления, что определяет влияние текущего (включая начальное) значения каждой из этих переменных на все последующие ее значения. Способ сцепления обуславливает вовлечение в преобразование каждой аккумулирующей переменной, а следовательно, и в преобразование каждого подблока данных, длинных цепочек подключей, выбираемых псевдослучайным образом. Попытка вычислить такие цепочки на основе входных и выходных сообщений с использованием алгебраических соотношений приводит к формированию системы нелинейных булевых уравнений с большим числом неизвестных, решение которой относится к трудно решаемым задачам.

Составление уравнений, которые могли бы быть решены с использованием разумных вычислительных ресурсов, приводит к необходимости включения в такие уравнения некоторых псевдослучайных значений аккумулирующих переменных. Допущение в соотношениях, используемых для криптоанализа, некоторых псевдослучайных параметров является общим для многих статистических типов атак, включая линейный и дифференциальный методы, которые признаны одними из

наиболее общих и эффективных методов при анализе блочных криптосистем. В настоящее время кроме комбинаторно-вероятностной модели в научной литературе не было предложено других способов оценок стойкости программных 512-байтовых алгоритмов, хотя последние обсуждаются в различных работах около 10 лет.

6.2. Дифференциальные характеристики блоков управляемых перестановок

Одним из универсальных методов криптоанализа блочных шифров, используемых с целью оценки их стойкости, является дифференциальный криптоанализ (ДКА). Основы базового метода ДКА представлены в *разд. 4.4*.

Еще раз поясним понятие дифференциального (разностного) криптоанализа.

Пусть дано некоторое достаточно сложное и зависящее от ключа взаимно однозначное отображение $Y = \mathbf{F}(X)$, где мощность множества X является весьма большим числом (например, 2^{64}). Как правило, для случайно выбранного X предсказать значение Y без выполнения данного отображения не представляется возможным. Пусть уже имеется некоторое достаточно большое число неповторяющихся пар (X_1, Y_1), (X_2, Y_2), ..., (X_r, Y_r). Если для любого X, не принадлежащего множеству $\{X_1, X_2, ..., X_r\}$, оказывается невозможным предсказать Y с вероятностью большей, чем $(2^n - r)^{-1}$, т. е. все еще не встречавшиеся значения Y, равновероятны, то говорят о псевдослучайном характере данного отображения. В противном случае отображение не является псевдослучайным.

Теперь рассмотрим одновременно два вектора X_1 и X_2. Используя некоторую бинарную операцию, каждой паре двоичных векторов можно поставить в соответствие третий вектор, который будем называть разностью. Обычно в качестве такой бинарной операции используется операция XOR ($X_1 \oplus X_2$), а разность принято обозначать символом Δ^X. Таким образом, входная разность Δ^X имеет вид: $\Delta^X = X_1 \oplus X_2$. Соответственно, разность значений функции $\mathbf{F}$, именуемая выходной разностью, имеет следующий вид: $\Delta^Y = Y_1 \oplus Y_2$. Обычно при определении входной и выходной разности используют одну и ту же бинарную операцию, хотя такое совпадение не является обязательным. В рассматриваемом примере даже при отсутствии статистической зависимости между выборками значений преобразуемых и преобразованных текстов может существовать статистическая зависимость между выборками *разностей* преобразуемых и преобразованных текстов. То есть в результате преобразования множества пар двоичных векторов, связанных между собой определенной разностью, может существовать неравномерность значений выходной разности Δ^Y.

Идея дифференциального криптоанализа как раз и состоит в том, чтобы найти такую входную разность, которая после прохождения через все шифрующее преобразование порождает некоторую разность с частотой встречаемости, сущест-

венно отличающейся от среднего значения. Нахождение таких входных и выходных разностей является основной задачей ДКА. Если такие разности найдены, то это означает, что найдены условия, при которых шифрующее преобразование распознается как неслучайное преобразование. После этого находится способ использования этого обстоятельства для вычисления секретного ключа. Поскольку в ДКА рассматривается шифрование специально выбранных пар текстов, то этот вид криптоанализа относится к случаю атаки на основе специально выбранных (подобранных) текстов.

Таким образом, в методе ДКА рассматривается прохождение разностей определенного вида через отдельные операции, раунд или все шифрующее преобразование, а при его реализации требуется шифрование большого числа пар блоков данных, разность между которыми равна постоянной величине.

Другими словами, в методе ДКА осуществляется поиск эффективных дифференциальных характеристик, использование которых позволяет вскрыть шифр. Напомним, что дифференциальной характеристикой называется тройка значений $(\Delta^X, \Delta^Y, p(\Delta^X \to \Delta^Y))$, где $p(\Delta^X \to \Delta^Y) = p(\Delta^Y / \Delta^X)$ — вероятность появления разности Δ^Y при условии, что на входе имеется разность Δ^X. Эффективной называется такая характеристика, для которой значение вероятности $p(\Delta^X \to \Delta^Y)$ для полного числа раундов шифрования существенно превышает 2^{-n}, где n — длина векторов X и Y.

Пусть найдена некоторая эффективная характеристика $(\Delta^X, \Delta^Y, p(\Delta^X \to \Delta^Y))$, причем $2^{-n+20} \approx p(\Delta^Y / \Delta^X) >> 2^{-n}$, однако для вычисления ключа требуется выполнить шифрование очень большого числа пар блоков данных. Как правило, для такого числа N имеет место неравенство $N \geq 1/p^2(\Delta^Y / \Delta^X)$ *(см. разд. 4.4.2)*. В рассматриваемом примере, чтобы показать, что шифрующее преобразование не является псевдослучайным, т. е. неотличимо от случайного преобразования, достаточно зашифровать около 2^{n-20} пар блоков данных. Однако для вычисления ключа потребуется выполнить 2^{2n-40} пар входных блоков данных, что при $n = 64$ является проблематичным, а при $n = 128$ вычислительно нереализуемым в настоящее время.

Таким образом, для некоторых блочных шифров можно показать, что они реализуют шифрующее преобразование, отличное от случайного, однако вычисление ключа является вычислительно сложной задачей. Если такая ситуация обнаруживается на этапе проектирования шифра, то разработчик должен модернизировать текущую версию шифра таким образом, чтобы сделать его псевдослучайной подстановкой (напомним, что любой блочный шифр является подстановкой). При этом, выполняемый разработчиком ДКА может не включать рассмотрение этапа вычисления ключа и трудоемкости реализации его различных вариантов. Достаточно построить шифр, неотличимый от случайного преобразования, для всех возможных дифференциальных характеристик.

Важным случаем дифференциальных характеристик являются итеративные характеристики, у которых входная и выходная разности совпадают. Для шифров,

основанных на перестановках, зависящих от преобразуемых данных, также важны дифференциальные характеристики, у которых входные и выходные разности принадлежат одному и тому же классу и имеют одинаковый и, как правило, небольшой вес Хэмминга.

Операции управляемых перестановок, на первый взгляд, представляются сложными для дифференциального криптоанализа. Однако при постепенном переходе от анализа дифференциальных характеристик элементарного переключателя к более сложным блокам управляемых перестановок можно получить достаточно простые общие зависимости, относящиеся к наиболее важным случаям дифференциального криптоанализа. Эти случаи связаны с определением вероятностей прохождения разностей с малым числом активных битов. Именно такие разности имеют наибольшие вероятности для криптосхем, основанных на использовании блоков управляемых перестановок. Действительно, как было показано при описании конкретных шифров в *гл. 5*, в одном раунде шифрования обычно используются две операции перестановок, зависящих от преобразуемых данных, причем их суперпозиция образует перестановку максимального порядка. Тогда активные биты попадают в произвольные разряды. Это является причиной того, что увеличение числа активных битов приводит к существенному уменьшению вероятности получения ожидаемой разности на выходе одного раунда. Далее это будет показано при конкретном рассмотрении дифференциального криптоанализа шифров, представленных в *гл. 5*.

Достаточно легко установить, что все дифференциальные характеристики элементарного блока $\mathbf{P}_{2/1}$ имеют ненулевые вероятности. Эти характеристики показаны на рис. 6.1, где верхний индекс разности Δ указывает на ее соответствие входу (x), выходу (y) или управляющему входу (v), а нижний индекс указывает на число единичных битов в разности. Используя данные свойства элементарного переключателя, являющегося базовым элементом более сложных блоков управляемых перестановок, и учитывая распределение битов управляющего подблока данных, легко вычислить вероятности дифференциальных характеристик наиболее важных типов блоков управляемых перестановок.

Рассмотрим случай использования блока расширения **E**, обеспечивающего влияние каждого бита управляющего подблока данных L на q ($q = 2, 3$) различных битов на выходе **E** и строго однократное влияние каждого бита L на все входные биты блока управляемых перестановок.

В общей схеме на рис. 6.2 в качестве блока $\mathbf{P}_{n/m}$ могут присутствовать блоки $\mathbf{P}_{16/32}$ ($q = 2$), $\mathbf{P}_{32/96}$ ($q = 3$) или $\mathbf{P}_{64/192}$ ($q = 3$). Обозначим через z число ненулевых битов в разности, проходящей через управляющий подблок данных L, а через w — число элементарных событий, относящихся к порождению пары активных битов на некотором элементарном блоке $\mathbf{P}_{2/1}$, на управляющий вход которого поступает ненулевая разность (рис. 6.2, *в*). Такие блоки $\mathbf{P}_{2/1}$ будем называть *активными*.

Рис. 6.1. Вероятности прохождения разностей через элементарный переключатель $\mathbf{P}_{2/1}$

Если на вход блока расширения поступает разность Δ_1^L с одним активным битом, то на управляющем входе блока управляемых перестановок появляется разность Δ_q^V с q активными битами. Если при этом на вход блока управляемых перестановок поступает нулевая разность Δ_0^X, то в зависимости от конкретных значений битов, поступающих на входы элементарных переключателей, соответствующих активным битам разности Δ_q^V, на выходе блока управляемых перестановок с некоторой вероятностью может появиться нулевая разность Δ_0^Y или ненулевая разность Δ_{2w}^Y $(w \le q)$ с четным числом активных битов. Учитывая вероятность превращения нулевой разности при прохождении через активный элементарный переключатель $\mathbf{P}_{2/1}$, легко получить следующую формулу:

$$p(\Delta_0^X \to \Delta_{2w}^Y / \Delta_z^L) \approx 2^{-qz} C_{qz}^w .$$

При $z = 1$ данная формула является точной, а при $z \ge 2$ дает хорошее приближение. При значениях $q = 3, 4, 5$ формула дает достаточно точное значение вероятности. Ею можно пользоваться и при более высоких значениях q, однако разности с большим числом активных битов не вносят существенный вклад в дифференциальные характеристики с наибольшими вероятностями.

Другой важный случай соответствует прохождению разности Δ_1^X с одним активным битом через блок управляемых перестановок при наличии ненулевой разности Δ_z^L, где $z \ne 0$ на управляющем входе. При этом активный бит входной разности может проходить мимо всех активных элементарных переключателей с вероятностью $p' \approx (n - 2qz)/n$ и с вероятностью $p'' \approx 2qz/n$ проходить через один

Рис. 6.2. Общая схема прохождения разности через блок управляемых перестановок (*а*), прохождение нулевых битов разности через активные блоки $\mathbf{P}_{2/1}$ без порождения активных битов (*б*) и с порождением активных битов (*в*)

из последних активных элементарных переключателей $\mathbf{P}_{2/1}$. Тогда для $z = 1$ указанные выражения для вероятностей являются точными, а при $z = 2, 3, 4$ дают достаточно хорошее приближение. Учитывая, что каждый активный элементарный переключатель $\mathbf{P}_{2/1}$ может породить с вероятностью 0,5 два активных бита выходной разности, легко получить следующее приближение для значения вероятности рассматриваемого события:

$$p(\Delta_1^X \to \Delta_{1+2w}^Y / \Delta_z^L) \approx 2^{-qz} C_{qz}^w \frac{(n-2qz)}{n} + 2^{1-qz} C_{qz-1}^w \frac{2qz}{n}.$$

При прохождении через блок управляемых перестановок разности с двумя и более активными битами существует вероятность "аннигиляции" четного числа активных битов. Наибольшая вероятность при этом соответствует обнулению двух активных битов. Рассмотрим, как это происходит для случая разностей Δ_2^X и Δ_z^L. С вероятностью близкой к $2qz/n$ один из активных битов разности Δ_2^X попадает на вход одного из активных элементарных переключателей. С вероятностью $1/(n-1)$ второй активный бит попадает на вход того же самого блока $\mathbf{P}_{2/1}$. С вероятностью 0,5 эти активные биты обнуляются одновременно. Оставшиеся $qz-1$ активных элементарных переключателей $\mathbf{P}_{2/1}$ с вероятностью 2^{-qz+1} не породят новых пар активных битов в выходной разности.

Перемножая вероятности всех этих независимых событий, получаем:

$$p(\Delta_2^X \to \Delta_0^Y / \Delta_z^L) \approx 2^{1-qz} \frac{qz}{n(n-1)}.$$

Данная формула выведена для случая усреднения по всем возможным значениям номеров активных битов i и j входной разности $\Delta_{2|i,j}^X$ (в предположении, что i и j являются случайными равновероятными величинами). Из рассмотренного механизма "аннигиляции" активных битов легко видеть, что одновременное обнуление двух и более пар активных битов имеет существенно меньшую вероятность. Некоторые значения вероятностей $p(\Delta^X \to \Delta^Y / \Delta_z^L)$ приведены в табл. 6.1 и 6.2.

Необходимо отметить, что приведенные выше формулы вычисления вероятностей для событий, относящихся к тем или иным классам разностей, характеризующихся в основном различным весом Хэмминга, дают интегральные оценки по достаточно большому классу разностей. В случае порождения дополнительных активных битов какая-либо привязка активных битов к конкретным разрядам отсутствует. Случай "аннигиляции" уже включает привязку второго бита к конкретному разряду, поскольку рассматривается попадание двух активных битов на вход одного и того же активного элементарного переключателя. Исходя из того, что приведенные выше оценки справедливы как для прямых, так и для обратных слоистых блоков управляемых перестановок, событие "аннигиляции" можно рассмотреть, обращая движение битов с выхода ко входу, что приводит к случаю, аналогичному порождению активных битов в обратном блоке управляемых перестановок. Поэтому полезно уяснить, почему вероятности в случае "аннигиляции" имеют существенно более низкие значения. Это обусловлено тем, что факт порождения пары активных битов не включает конкретные входные биты, попадающие на вход одного из активных элементарных переключателей, т. е. данный случай включает намного более многочисленный класс событий. Событие "аннигиляции" имеет такие значения вероятностей, которые соответствуют случаю порождения активных битов в заданных выходных разрядах.

С этим обстоятельством мы встретимся далее при выполнении дифференциального криптоанализа конкретных шифров, использующих управляемые перестановки как базовый криптографический примитив.

Исследование дифференциальных свойств криптосистем, основанных на управляемых операциях, включает также предварительное рассмотрение вероятностей прохождения различных разностей через операцию **G**. Значения вероятностей могут быть легко получены из формул, описывающих булевы функции, соответствующие данной операции. Ввиду того, что в различных шифрах используются разные операции подобного типа, их свойства будут рассмотрены непосредственно при выполнении дифференциального криптоанализа конкретных криптосистем.

Таблица 6.1. *Значения вероятностей* $p(\Delta^X \to \Delta^Y / \Delta_1^L)$ *для блока* $\boldsymbol{P}_{64/192}$

$Z = 1$	Δ_0^Y	Δ_2^Y	Δ_4^Y	Δ_6^Y
Δ_0^X	2^{-3}	$1{,}5\cdot 2^{-2}$	$1{,}5\cdot 2^{-2}$	2^{-3}
Δ_2^X	$1{,}52\cdot 2^{-13}$	$1{,}11\cdot 2^{-3}$	$1{,}42\cdot 2^{-2}$	$1{,}27\cdot 2^{-2}$

$Z = 1$	Δ_1^Y	Δ_3^Y	Δ_5^Y	Δ_7^Y
Δ_1^X	$1{,}1\cdot 2^{-3}$	$1{,}55\cdot 2^{-2}$	$1{,}45\cdot 2^{-2}$	$0{,}91\cdot 2^{-3}$

Таблица 6.2. *Значения вероятностей* $p(\Delta^X \to \Delta^Y / \Delta_2^L)$ *для блока* $\boldsymbol{P}_{64/192}$

$Z = 2$	Δ_0^Y	Δ_2^Y	Δ_4^Y	Δ_6^Y
Δ_0^X	2^{-6}	$1{,}5\cdot 2^{-4}$	$1{,}88\cdot 2^{-3}$	$1{,}25\cdot 2^{-2}$
Δ_2^X	$1{,}52\cdot 2^{-15}$	$1{,}1\cdot 2^{-6}$	$1{,}51\cdot 2^{-4}$	$1{,}72\cdot 2^{-3}$

$z = 2$	Δ_1^Y	Δ_3^Y	Δ_5^Y	Δ_7^Y
Δ_1^X	$1{,}19\cdot 2^{-6}$	$1{,}69\cdot 2^{-4}$	2^{-2}	$1{,}25\cdot 2^{-2}$

6.3. Анализ криптосистемы SPECTR-H64

Использование многослойных блоков управляемых перестановок в блочном шифре SPECTR-H64 приводит к тому, что наиболее высокие вероятности имеют дифференциальные характеристики с малым числом активных битов.

В рассматриваемой схеме разность с одним активным битом в правой ветви (при наличии нулевой разности в левой ветви) проходит один раунд преобразования с вероятностью, равной единице. В этой же схеме при наличии одного активного бита в левой ветви и нулевой разности в правой ветви имеется сравнительно высокая вероятность получения нулевой разности в правой ветви. То есть в данном случае общая выходная разность будет совпадать с входной разностью. Рассмотрим подробнее этот случай, который, с учетом предыдущего, позволяет формировать 2-, 4-, $2k$-раундовые характеристики.

Хотя в рассматриваемом случае на выходе одного раунда разность в правой ветви равна нулю, внутри раунда активный бит, проходящий через левую ветвь криптосхемы, оказывает существенное влияние на формирование новых активных битов в правой ветви. При этом в блоке управляемых перестановок активные биты порождаются вероятностным способом, а при выполнении операции **G** — как вероятностно, так и по детерминированному закону. При наличии одного активного бита на входе операции **G** на ее выходе формируется от двух до шести активных битов (два активных бита порождаются детерминировано). Порождение четного числа активных битов (0, 2, 4, 6) реализуется также в каждом из блоков $\mathbf{P}'$ или $\mathbf{P}''$. После выполнения операции XOR (перед выполнением управляемой перестановки $\mathbf{P}_{32/80}^{-1}$) общее число активных битов может не только увеличиться (до 18), но и уменьшиться (до 0).

Поскольку нас интересует ситуация, когда на выходе блока $\mathbf{P}_{32/80}^{-1}$ реализуется нулевая разность (отсутствие "размножения" активных битов), наибольший интерес представляет случай, когда число таких активных битов после выполнения операции XOR равно нулю либо двум. Именно для данных значений вероятности получения нулевой вероятности на выходе блока $\mathbf{P}_{32/80}^{-1}$ имеют существенные значения.

Эти рассуждения позволяют сформировать 2-раундовую дифференциальную характеристику, представленную на рис. 6.3 с использованием следующих обозначений:

- Δ^L и Δ^R — разности в левом (L) и правом (R) подблоке;
- $\Delta^{\mathbf{F}}$ и $\Delta^{(\mathbf{F})}$ — входная и выходная разности, соответствующие операции **F**;
- $\Delta_{2|i,i+1}^{(\mathbf{G})}$ — разность, в нижнем регистре которой указано общее число активных битов, а после вертикальной черты — номера разрядов, соответствующих активным битам.

При таком обозначении записи $\Delta_{2|i,j}$ и Δ_2 имеют следующие значения: первая обозначает конкретную разность с двумя активными битами, а вторая — лишь одну из разностей с двумя активными битами.

С учетом конструктивных особенностей блока $\mathbf{P}_{32/80}$, используемого в одном раунде шифра SPECTR-H64, активные биты l_i подблока данных L управляют раз-

ным числом блоков $\mathbf{P}_{2/1}$ в зависимости от значения номера разряда i и от значения циклического сдвига подблока данных L перед выполнением процедуры расширения. Число блоков $\mathbf{P}_{2/1}$, управляемых одним битом l_i, обозначим через q.

Рассмотрим основные варианты прохождения одного активного бита по i-му разряду левой ветви без порождения активных битов на выходе блока $\mathbf{P}''$.

Рис. 6.3. Схема формирования двухраундовой характеристики в шифре SPECTR-H64 для события A1

Событие A1

1. На выходе операции $\mathbf{G}$ с вероятностью $p_2^{(i)}$ формируется разность $\Delta_{2|i,i+1}^{(\mathbf{G})}$.
2. На выходе блока $\mathbf{P}'$ с вероятностью $p_3^{(i,i+1)}$ формируется разность $\Delta_{2|i,i+1}^{(\mathbf{P}')}$.
3. На выходе блока $\mathbf{P}''$ с вероятностью $p_1^{(i)} = 2^{-q}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}^{-1}$ формируется нулевая разность, которая проходит через этот блок с вероятностью $p_4^{(i)} = 2^{-q}$.

Событие A2

1. На выходе операции **G** с вероятностью $p_2^{(i)}$ формируется разность $\Delta_{2|i,i+1}^{(\mathbf{G})}$.
2. На выходе блока **P′** с вероятностью $p_3^{(i)} = 2^{-q}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
3. На выходе блока **P″** с вероятностью $p_1^{(i,i+1)}$ формируется разность $\Delta_{2|i,i+1}^{(\mathbf{P}'')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}^{-1}$ формируется нулевая разность, которая проходит через этот блок с вероятностью $p_4^{(i)} = 2^{-q}$.

Событие A3

1. На выходе операции **G** с вероятностью $p_2^{(i)}$ формируется разность $\Delta_{2|i,i+1}^{(\mathbf{G})}$.
2. На выходе блока **P′** с вероятностью $p_3^{(i)}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
3. На выходе блока **P″** с вероятностью $p_1^{(i)}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}^{-1}$ формируется разность $\Delta_{2|i,i+1}$, которая обнуляется в этом блоке с вероятностью $p_4^{(i,i+1)}$.

Поясним, например, вычисление вероятности $p_2^{(i)}$. Для этого запишем общую формулу вычисления бита в i-м разряде на выходе функции **G**, задающей преобразование $Y = \mathbf{G}(X, A, B)$, где A и B — подключи, используемые при выполнении данного преобразования:

$$y_i = x_i \oplus x_{i-1} \oplus x_{i-2}a_i \oplus x_{i-2}x_{i-5}b_i \oplus x_{i-3}x_{i-5} \oplus x_{i-4}b_i,$$

где a_i и b_i — биты подключей. Из последнего соотношения легко получить следующие формулы, характеризующие изменение выходных битов y_i, y_{i+1}, y_{i+2}, y_{i+3}, y_{i+4}, y_{i+5}, обусловленных изменением единственного входного бита x_i ($\Delta x_i = 1$), а именно:

$$\Delta y_i = \Delta x_i; \qquad \Delta y_{i+1} = \Delta x_i;$$
$$\Delta y_{i+2} = \Delta x_i(a_{i+2}); \qquad \Delta y_{i+3} = \Delta x_i(x_{i-2});$$
$$\Delta y_{i+4} = \Delta x_i(b_{i+4}); \qquad \Delta y_{i+5} = \Delta x_i(x_{i+2} \oplus x_{i+3} \oplus b_{i+5}).$$

С учетом данных формул легко вычислить следующие значения вероятностей:

$$p(\Delta y_i = 1/\Delta x_i = 1) = p(\Delta y_{i+1} = 1/\Delta x_i = 1) = 1,$$

а для $k = 2, 3, 4, 5$

$$p(\Delta y_{i+k} = 0/\Delta x_i = 1) = 1/2.$$

Но тогда, в предположении, что подключи A и B являются случайными равновероятными величинами, значение $p_2^{(i)}$ определяется формулой:

$$p_2^{(i)} = p(\Delta y_i = 1 / \Delta x_i = 1)\, p(\Delta y_{i+1} = 1 / \Delta x_i = 1)\, p(\Delta y_{i+2} = 0 / \Delta x_i = 1) \ldots p(\Delta y_{i+5} = 0 / \Delta x_i = 1).$$

Замечание

В формуле необходимо учитывать ограничение на максимально возможное значение индексов i и $i+k$ у сомножителей при Δy_{i+k}, которое не должно превосходить значение 32.

Вероятности $p_2^{(i)}$ для различных значений индекса i приведены ниже.

Для $i = 1, 2$ — $p_2^{(i)} = 0$, поскольку в этих случаях в выходной разности формируется не менее трех активных битов.

Для $3 \le i \le 27$ — $p_2^{(i)} = 2^{-4}$, причем данное значение получается перемножением всех шести сомножителей, присутствующих в приведенной выше формуле.

Для значений $i \ge 28$ с учетом замечания имеем:

- $i = 28$ — $p_2^{(i)} = 2^{-3}$;
- $i = 29$ — $p_2^{(i)} = 2^{-2}$;
- $i = 30$ — $p_2^{(i)} = 2^{-1}$;
- $i = 31$ — $p_2^{(i)} = 1$;
- $i = 32$ — $p_2^{(i)} = 0$.

Для расчета вероятности события A1 можно воспользоваться таблицей распределения влияния битов подблока L на элементарные переключатели блока $\mathbf{P}'$. Из описания алгоритма SPECTR-H64 легко составить такую таблицу (табл. 6.3), в которой ячейки обозначают блоки $\mathbf{P}_{2/1}$, а указанные в них номера (i) — номера соответствующих битов в подблоке L.

Таблица 6.3. *Распределение влияния битов управляющего подблока L в блоке* ***P′***

28	29	30	31	32	1	2	3	4	5	6	7	8	9	10	11
5	6	7	8	9	10	11	4	29	30	31	32	1	2	3	28
9	10	11	4	5	6	7	8	1	2	3	28	29	30	31	32
25	26	27	20	21	22	23	24	17	18	19	12	13	14	15	16
21	22	23	24	25	26	27	20	13	14	15	16	17	18	19	12

Используя данную таблицу, можно определить все значения i, для которых на выходе блока $\mathbf{P}'$ может появиться разность $\Delta_{2|i,i+1}^{(\mathbf{P}'')}$, а также определить значение вероятности этого события, а именно:

$$p(\Delta_0^{\mathbf{P}'} \to \Delta_{2|i,i+1}^{(\mathbf{P}')} / \Delta_{1|i}^{L}) = p_3^{(i,i+1)}.$$

Например, при $i = 28$ активными являются элементарные блоки $\mathbf{P}_{2/1}^{(1)}$, $\mathbf{P}_{2/1}^{(32)}$ и $\mathbf{P}_{2/1}^{(44)}$, на выходе каждого из которых с вероятностью 0,5 может появиться пара активных битов (верхний индекс в обозначении $\mathbf{P}_{2/1}^{(j)}$ соответствует номеру элементарного блока). При этом только для блока $\mathbf{P}_{2/1}^{(1)}$ выходная пара активных битов может попасть одновременно в разряды i и $i + 1$.

Нас интересует случай появления на выходе $\mathbf{P}_{2/1}^{(1)}$ активных битов при одновременном отсутствии активных битов на выходах блоков $\mathbf{P}_{2/1}^{(32)}$ и $\mathbf{P}_{2/1}^{(44)}$. Вероятность данного события равна $p_3^{(i,i+1)} = 2^{-3}2^{-4}2^{-4} = 2^{-11}$, поскольку левый и правый выходные биты блока $\mathbf{P}_{2/1}^{(1)}$ попадут, соответственно, в 28-й и 29-й разряды на выходе блока $\mathbf{P}'$ с вероятностью 2^{-4}.

Всего имеются четыре разряда левого подблока (номера 4, 8, 18 и 28), удовлетворяющие описанной выше ситуации. Для $i = 4, 8, 28$ имеем вероятности $p_3^{(4,5)} = p_3^{(8,9)} = p_3^{(28,29)} = 2^{-11}$, а для $i = 18$ — $p_3^{(18,19)} = 2^{-4}$. Заметим, что именно разряд $i = 18$ вносит основной вклад в формирование значения вероятности события A1.

Вероятность попадания разности Δ_1^R в заданный i-й разряд на выходе первого раунда равна $p^{(i)} = 2^{-5}$. Вероятность того, что на выходе функции $\mathbf{G}$ не будут порождаться активные биты в разрядах $i + 2$, $i + 3$, $i + 4$, $i + 5$ (для $3 \le i \le 27$ и случая усреднения по случайным раундовым ключам) равна $p_2^{(i)} = 2^{-4}$. Соответственно, для $i = 28$ $p_2^{(28,29)} = 2^{-3}$. Таким образом, для события A1 имеем следующую суммарную вероятность:

$$P' = \sum_{i=1}^{i=32} p^{(i)} p_1^{(i)} p_2^{(i)} p_3^{(i,i+1)} p_4^{(i)} = \sum_{i=4,8,18,28} p^{(i)} p_1^{(i)} p_2^{(i)} p_3^{(i,i+1)} p_4^{(i)} \approx 2^{-17}.$$

Преобладающий вклад в значение P' вносит 18-й бит в левом подблоке, для которого имеем более высокие значения вероятностей $p_1^{(18)}$, $p_4^{(18)}$ и $p_3^{(18,19)}$ ($p_1^{(18)} = p_4^{(18)} = 2^{-2}$ и $p_1^{(18,19)} = 2^{-4}$).

Для события A1 для всех значений i имеем $p_1^{(i)} = 2^{-3}$, а для событий A1 и A2 — $p^{(i)} = 2^{-5}$ и $p_4^{(i)} = 2^{-3}$.

С учетом полной симметрии прямой ($\mathbf{P}'$) и обратной ($\mathbf{P}^{-1}$) управляемых перестановок, выполняющих преобразование правого подблока данных в одном раунде шифра SPECTR-H64, вероятности событий A1 и A3 одинаковы. То есть:

$$P''' = P' \approx 2^{-17}.$$

Предполагая, что ключевой элемент на входе блока **P″** является равновероятной случайной величиной, легко вычислить вероятность события A2 по следующей формуле, которая аналогична предыдущей:

$$P'' = \sum_{i=1}^{i=32} p^{(i)} p_1^{(i,i+1)} p_2^{(i)} p_3^{(i)} p_4^{(i)} = \sum_{i=4,8,12,16,21} p^{(i)} p_1^{(i,i+1)} p_2^{(i)} p_3^{(i)} p_4^{(i)} \approx 2^{-15}.$$

Для события A2 для всех значений i имеем $p_3^{(i)} = 2^{-3}$. Однако при вычислении значений $p_1^{(i,i+1)}$ вместо табл. 6.3 необходимо использовать табл. 6.4, поскольку левый подблок, перед тем как он поступает на вход блока расширения **E**, подвергается преобразованию операцией циклического сдвига не на 11 разрядов, а на 17. Преобладающий вклад в значение вероятности P'' вносит 21-й разряд, для которого $p_1^{(21,22)} = 2^{-2}$ и $p_3^{(21)} = p_4^{(21)} = 2^{-2}$.

Таблица 6.4. *Распределение влияния битов управляющего подблока L в блоке **P″***

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
11	12	13	14	15	16	17	10	3	4	5	6	7	8	9	2
15	16	17	10	11	12	13	14	7	8	9	2	3	4	5	6
31	32	1	26	27	28	29	30	23	24	25	18	19	20	21	22
27	28	29	30	31	32	1	26	19	20	21	22	23	24	25	18

Таким образом, вероятность прохождения разности $(0, \Delta_1^R)$ через два раунда равна:

$$P(2) \approx P' + P'' + P''' \approx 1{,}5 \cdot 2^{-15}.$$

В этом значении $P(2)$ не учитывается вклад событий, соответствующих появлению в выходной разности функции **G** четырех или шести активных битов, а также событий с двумя активными битами, приводящих к одновременному формированию трех разностей $\Delta_{2|i,i+1}^{(\mathbf{G})}$, $\Delta_{2|i,z}^{(\mathbf{P}')}$ и $\Delta_{2|i+1,z}^{(\mathbf{P}'')}$, удовлетворяющих условию:

$$\Delta_{2|i,i+1}^{(\mathbf{G})} \oplus \Delta_{2|i,z}^{(\mathbf{P}')} \oplus \Delta_{2|i+1,z}^{(\mathbf{P}'')} = 0 .$$

Данными событиями можно пренебречь, поскольку их суммарная вероятность существенно меньше значения $1{,}5 \cdot 2^{-15}$.

С учетом закономерностей, выявленных при вычислении вероятностей рассматриваемых событий (A1 и A2), можно сделать вывод, что стойкость шифра к дифференциальному криптоанализу существенно зависит от распределения управляющих битов в блоке расширения **E**. При этом значение вероятности двухраундовой дифференциальной характеристики можно существенно уменьшить, если оптимизировать блок расширения **E**. Например, если в табл. 6.3 в нижней строке

поменять местами номера 21 и 20, а в табл. 6.4 — номера 18 и 19, то такая перестановка позволяет уменьшить значения вероятностей, а именно $P' \approx 2^{-23}$, $P'' \approx 2^{-21}$ и $P(2) \approx 1{,}5 \cdot 2^{-21}$. То есть предложенное изменение приводит к уменьшению вероятности двухраундовой характеристики примерно в 2^6 раз.

После такого изменения наиболее эффективной является трехраундовая дифференциальная характеристика, рассматриваемая далее.

Действительно, для шифра SPECTR-H64 имеется еще одна дифференциальная характеристика, соответствующая разностям с малым числом активных битов. Эта характеристика соответствует прохождению разности $(0, \Delta_1^R)$ через три раунда. Ее формирование описывается следующим механизмом. После первого раунда активный бит попадает в 32-й разряд с вероятностью 2^{-5}. Во втором раунде активный бит, проходя через операцию **G**, с вероятностью, равной 1, порождает активный бит в разности, проходящей по правой ветви, которая с вероятностью 2^{-5} оказывается в 32-м разряде на выходе правой ветви второго раунда. Вероятность того, что при выполнении трех операций управляемой перестановки дополнительно не появятся активные биты в правой ветви равна $(2^{-3})^3 = 2^{-9}$. Через третий раунд распространяются разности $\Delta_{1|32}^L$ и $\Delta_{1|32}^R$. С вероятностью 2^{-5} активный бит правой разности после выполнения операции **P′** оказывается в 32-м разряде и после выполнения операции XOR с выходной разностью $\Delta_{1|32}^{\mathbf{G}}$ обнуляется. В третьем раунде также с вероятностью 2^{-9} три операции управляемой перестановки не породят новых активных битов. Учитывая данную схему распространения разности (Δ_0^L, Δ_1^R) через три раунда, для нее получаем следующее значение вероятности: $P(3) \approx 2^{-28}$ (знак "приближенно равно" учитывает, что мы пренебрегаем вкладом событий, в которых формируются промежуточные разности с несколькими активными битами в правом подблоке). Соответственно, для шести раундов вероятность прохождения разности (Δ_0^L, Δ_1^R) с использованием данной трехраундовой характеристики равна $P''(6) \approx (2^{-28})^2 = 2^{-56}$, а с использованием двухраундовой характеристики — $P'(6) \approx (1{,}5 \cdot 2^{-15})^3 \approx 1{,}7 \cdot 2^{-44} >> P''(6)$. Таким образом, при формировании 6-раундовой характеристики с одним активным битом для анализа алгоритма шифрования SPECTR-H64 двухраундовая характеристика более эффективная.

Используя двухраундовую характеристику для построения дифференциальных характеристик с четным числом раундов, можно получить следующие значения вероятности:

- $P(8) \approx (1{,}5 \cdot 2^{-15})^4 \approx 5 \cdot 2^{-60} > 2^{-64}$;
- $P(10) \approx (1{,}5 \cdot 2^{-15})^5 \approx 2^{-72} << 2^{-64}$;
- $P(12) \approx (1{,}5 \cdot 2^{-15})^6 \approx 2^{-87} << 2^{-64}$.

Последние соотношения свидетельствуют, что при использовании десяти или двенадцати раундов шифрования алгоритмом SPECTR-H64 применение соответствующих характеристик, основанных на двухраундовой характеристике с одним активным битом, неприменимо для вскрытия шифра, поскольку данные характеристики не позволяют отличить шифр SPECTR-H64 от случайного шифра. То есть можно говорить о том, что алгоритм SPECTR-H64 является стойким к рассмотренному варианту дифференциального анализа.

Вариант с восемью раундами шифрования в принципе может быть раскрыт после зашифрования около 2^{60} пар специально подобранных входных 64-битовых блоков данных, хотя реализация на практике такой атаки, скорее всего, неосуществима.

Как отмечалось ранее, соответствующая вероятность для двухраундовой характеристики шифра SPECTR-H64 может быть существенно уменьшена путем оптимизации блока расширения **E**. Однако для трехраундовой характеристики аналогичная вероятность слабо зависит от вида распределения управляющих битов, задаваемого расширением **E**. Поэтому после модернизации шифра SPECTR-H64 его стойкость к дифференциальному криптоанализу будет определяться трехраундовой характеристикой. Для уменьшения вероятности в трехраундовой характеристике дополнительно требуется изменить булевы функции, задающие конкретный тип операции **G**.

6.4. Дифференциальный криптоанализ шифра SPECTR-128

Структура итеративной криптосистемы SPECTR-128 аналогична структуре алгоритма SPECTR-H64, что определяет сходство дифференциальных свойств обоих шифров. Для алгоритма SPECTR-128 дифференциальные характеристики с малым числом активных битов также имеют самые высокие вероятности. Наибольшую вероятность имеет характеристика, соответствующая прохождению разности (Δ_0^L, Δ_1^R) через два раунда. Механизм ее распространения аналогичен случаю SPECTR-H64. Однако ввиду особенностей операции **G** и большего размера преобразуемых подблоков вычисления являются более громоздкими. При выполнении дифференциального анализа необходимо учитывать, что при изменении только одного входного бита операции **G** на ее выходе обязательно изменится только один бит, соответствующий тому же разряду, что и измененный входной бит. Кроме того, с вероятностью 0,5 могут измениться еще шесть выходных битов. Основные различные варианты формирования двухраундовой характеристики связаны с рассмотрением появления на выходе операции **G** разностей $\Delta_{2|i,i+1}^{(\mathbf{G})}$, $\Delta_{2|i,i+3}^{(\mathbf{G})}$, $\Delta_{2|i,i+6}^{(\mathbf{G})}$, $\Delta_{2|i,i+7}^{(\mathbf{G})}$, $\Delta_{2|i,i+8}^{(\mathbf{G})}$ и $\Delta_{2|i,i+9}^{(\mathbf{G})}$. Каждая из данных разностей вносит свой вклад в значение вероятности двухраундовой характери-

стики. Этот вклад вычисляется аналогично вычислениям в алгоритме SPECTR-H64 для заданного i и при рассмотрении выходной разности $\Delta^{(\mathbf{G})}_{2|i,i+1}$ с двумя активными битами. Возможность появления различных разностей на выходе операции **G** может быть учтена в каждом из событий A1, A2 и A3, что сделано далее. Используя зависимость значения выходных битов, выраженную в виде булевой функции, легко записать формулы, отражающие изменение выходных битов в разряде $i + k$, где $k = 0, 1, 2, \ldots, 9$ (табл. 6.5).

Таблица 6.5. *Вероятности порождения на выходе операции **G** активного бита в (i + k)-м разряде при изменении i-го входного бита*

Формула	**Вероятность**
$\Delta y_i = \Delta l_i$	$p(\Delta y_i=1) = 1$
$\Delta y_{i+1} = \Delta l_i(a^{(4)}_{i-1} \oplus a^{(3)}_{i-1} l_{i-8} \oplus a^{(4)}_i l_{i-5} l_{i-8})$	$p(\Delta y_{i+1}=1) = ½$
$\Delta y_{i+2} = 0$	$p(\Delta y_{i+2}=1) = 0$
$\Delta y_{i+3} = \Delta l_i\, l_{i-6}$	$p(\Delta y_{i+3}=1) = ½$
$\Delta y_{i+4} = 0$	$p(\Delta y_{i+4}=1) = 0$
$\Delta y_{i+5} = 0$	$p(\Delta y_{i+5}=1) = 0$
$\Delta y_{i+6} = \Delta l_i(l_{i-2} \oplus l_{i-3} l_{i+5} a^{(4)}_{i+5})$	$p(\Delta y_{i+6}=1) = ½$
$\Delta y_{i+7} = \Delta l_i a^{(3)}_{i+5})$	$p(\Delta y_{i+7}=1) = ½$
$\Delta y_{i+8} = \Delta l_i l_{i+2}$	$p(\Delta y_{i+8}=1) = ½$
$\Delta y_{i+9} = \Delta l_i(l_{i+6} \oplus l_{i+8} a^{(3)}_{i+7} \oplus l_{i+3} l_{i+8} a^{(4)}_{i+8})$	$p(\Delta y_{i+9}=1) = ½$

Рассмотрим события A1, A2 и A3 (рис. 6.4) для SPECTR-128.

Событие A1

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$, где $k \in \{1, 3, 6, 7, 8, 9\}$, формируется разность $\Delta^{(\mathbf{G})}_{2|i,i+k}$.
2. На выходе блока **P′** с вероятностью $p_3^{(i,i+k)}$ формируется разность $\Delta^{(\mathbf{P}')}_{2|i,i+k}$.
3. На выходе блока **P″** с вероятностью $p_1^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}'')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}^{-1}_{64/192}$ формируется нулевая разность, которая проходит через этот блок с вероятностью $p_4^{(i)} = 2^{-3}$.

Рис. 6.4. Схема формирования двухраундовой характеристики шифра SPECTR-128 для события A1

Событие A2

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{G})}$.
2. На выходе блока $\mathbf{P}'$ с вероятностью $p_3^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
3. На выходе блока $\mathbf{P}''$ с вероятностью $p_1^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{P}'')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}_{64/192}^{-1}$ формируется нулевая разность, которая проходит через этот блок с вероятностью $p_4^{(i)} = 2^{-3}$.

Событие A3

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{G})}$.
2. На выходе блока $\mathbf{P}'$ с вероятностью $p_3^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.

3. На выходе блока $\mathbf{P}''$ с вероятностью $p_1^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
4. После выполнения двух операций XOR на входе блока $\mathbf{P}^{-1}$ формируется разность $\Delta_{2|i,i+k}$, которая обнуляется в этом блоке с вероятностью $p_4^{(i,i+k)}$.

Как и в случае с алгоритмом SPECTR-H64, ввиду симметрии событий A1 и A3 их вероятности одинаковы. Поэтому достаточно определить вероятности событий A1 и A2.

Используя описание алгоритма SPECTR-128 для вычисления вероятности событий A1 и A2, составим таблицы распределения битов левого подблока в управлении элементарными переключателями блоков $\mathbf{P}'$ и $\mathbf{P}''$, которые представлены в виде табл. 6.6 и 6.7.

Таблица 6.6. *Распределение битов подблока L в блоке **P′** шифра SPECTR-128*

Левая часть P′ (соответствует входным битам с номерами от 1 до 32)															
54	55	56	57	58	59	60	61	62	63	64	1	2	3	4	5
7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	6
15	16	17	18	19	20	21	6	7	8	9	10	11	12	13	14
47	48	49	50	51	52	53	38	39	40	41	42	43	44	45	46
39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	38
22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37
Правая часть P′ (соответствует входным битам с номерами от 33 до 64)															
6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
55	56	57	58	59	60	61	62	63	64	1	2	3	4	5	54
63	64	1	2	3	4	5	54	55	56	57	58	59	60	61	62
31	32	33	34	35	36	37	22	23	24	25	26	27	28	29	30
23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	22
38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53

Вероятность события A1 вычисляется по формуле:

$$P' = \sum_{i=1}^{64} \sum_{k=1}^{9} p^{(i)} p_1^{(i)} p_2^{(i,i+k)} p_3^{(i,i+k)} p_4^{(i)} \approx 1{,}5 \cdot 2^{-21},$$

где $p^{(i)} = 2^{-6}$ — вероятность перехода активного бита разности в i-й разряд после первого раунда. Пользуясь структурой блока $\mathbf{P}_{64/192}^{-1}$ и табл. 6.6 и 6.7, легко определить значения вероятностей, входящих в данную сумму. При этом в значение вероятности основной вклад вносит вариант, соответствующий активному биту

в 43-м разряде левого подблока, для которого вероятность $p_3^{(43,44)} = 2^{-3}$, что составляет примерно около 70% значения P'. Примерно 15% приходится на номера $i = 33$ и 34, для которых имеем $p_3^{(33,39)} = p_3^{(33,40)} = p_3^{(34,36)} = p_3^{(34,40)} = 2^{-7}$. Оставшаяся доля приходится на значения $i = 3$, 7, 8, 11, 12, 15, 16, 20 и 54.

Вероятность события A2 вычисляется по формуле:

$$P'' = \sum_{i=1}^{64} \sum_{k=1}^{9} p^{(i)} p_1^{(i,i+k)} p_2^{(i,i+k)} p_3^{(i)} p_4^{(i)} \approx 1{,}3 \cdot 2^{-21}.$$

Таблица 6.7. *Распределение битов подблока L в блоке* ***P″*** *шифра SPECTR-128*

Левая часть P″															
12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	28
37	38	39	40	41	42	43	28	29	30	31	32	33	34	35	36
5	6	7	8	9	10	11	60	61	62	63	64	1	2	3	4
61	62	63	64	1	2	3	4	5	6	7	8	9	10	11	60
44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
Правая часть P″															
28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43
13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	12
21	22	23	24	25	26	27	12	13	14	15	16	17	18	19	20
53	54	55	56	57	58	59	44	45	46	47	48	49	50	51	52
45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	44
60	61	62	63	64	1	2	3	4	5	6	7	8	9	10	11

Основной вклад в вероятность P' вносят случаи $i = 54$ и 57 ($p_1^{(54,55)} = p_1^{(57,60)} = 2^{-5}$), а также $i = 9$, 10, 11, 44 ($p_1^{(9,15)} = p_1^{(9,16)} = p_1^{(10,13)} = p_1^{(10,16)} = p_1^{(11,14)} = p_1^{(44,45)} = p_1^{(44,47)} = 2^{-7}$). Вероятность прохождения разности (Δ_0^L, Δ_1^R) через два раунда равна

$$P(2) \approx P' + P'' + P''' \approx 1{,}1 \cdot 2^{-19}.$$

Необходимо отметить, что как и при анализе дифференциальных свойств шифра SPECTR-H64, отдельные особенности таблицы распределения влияния битов левого подблока оказывают преобладающее влияние на операции управляемой

перестановки. Внося незначительные изменения в блок расширения **E**, легко можно уменьшить вероятность $P(2)$.

Для шифра SPECTR-128 имеется также и трехраундовая характеристика, соответствующая прохождению разности (Δ_0^L, Δ_1^R) через три раунда. В отличие от алгоритма SPECTR-H64 прохождение разности (Δ_0^L, Δ_1^R) не связано с принадлежностью активного бита в левом подблоке к старшему разряду. Каждый разряд вносит свой вклад, поскольку, благодаря использованию операции **G**, отличной от операции **G** в шифре SPECTR-H64, присутствие активного бита в любом разряде подблока L изменяет предопределенно только один бит на выходе этой операции. Схема формирования трехраундовой характеристики показана на рис. 6.5.

При расчете вероятности трехраундовой характеристики $P(3)$ необходимо различать случаи распространения активного бита левого подблока по различным разрядам. Действительно, вероятность получения разности $\Delta_{1|i}^{(\mathbf{G})}$ для различных значений i на выходе операции **G** равна:

- для $i = 1, 2, \ldots, 55$ — $p = 2^{-6}$;
- для $i = 56$ — $p = 2^{-5}$;
- для $i = 57$ — $p = 2^{-4}$;
- для $i = 58$ — $p = 2^{-3}$;
- для $i = 59, 60, 61$ — $p = 2^{-2}$;
- для $i = 62, 63$ — $p = 2^{-1}$;
- для $i = 64$ — $p = 1$.

Усредняя по всем значениям i, легко получить среднее значение вероятности появления на выходе операции **G** разности $\Delta_1^{(\mathbf{G})}$ с одним активным битом, а именно:

$$p(\Delta_1^{(\mathbf{G})}) = 0{,}97 \cdot 2^{-4} \approx 2^{-4}.$$

Во втором и третьем раундах по левой ветви распространяется один активный бит. В каждом из этих раундов вероятность непорождения пар активных битов на выходах трех блоков управляемых перестановок составляет 2^{-9}. При этом во втором раунде активный бит в правом подблоке порождается за счет операции **G**. В третьем раунде активный бит правого подблока обнуляется активным битом разности $\Delta_{1|h}^{(\mathbf{G})}$ с вероятностью 2^{-6} (это вероятность перемещения активного бита правого подблока в заданный h-й разряд операцией $\mathbf{P}_{64/192}$). С учетом сделанных замечаний легко получить, что $P(3) \approx 2^{-32}$.

Вклад трехраундовой характеристики в вероятность прохождения разности (Δ_0^L, Δ_1^R) через шесть раундов составляет $P''(6) < (2^{-32})^2 = 2^{-64}$. Вклад двухраун-

довой характеристики составляет $P'(6) \approx (1{,}1 \cdot 2^{-19})^3 \approx 1{,}3 \cdot 2^{-57} >> P''(6)$. Таким образом, при определении стойкости шифра SPECTR-128 более эффективной является двухраундовая характеристика.

Используя значение вероятности $P(2)$ можно получить следующие вероятности:

- $P(8) \approx (1{,}1 \cdot 2^{-19})^4 \approx 1{,}5 \cdot 2^{-76}$;
- $P(10) \approx (1{,}1 \cdot 2^{-19})^5 \approx 1{,}7 \cdot 2^{-95}$;
- $P(12) \approx (1{,}1 \cdot 2^{-19})^6 \approx 2^{-113}$.

Рис. 6.5. Схема формирования трехраундовой характеристики шифра SPECTR-128

Если сравнить вероятности двухраундовых характеристик шифров SPECTR-H64 и SPECTR-128, то несложно заметить, что значение $P(2)$ для шифра SPECTR-H64 больше, чем значение $P(2)$ для шифра SPECTR-128, однако первый неразличим с помощью дифференциального криптоанализа от случайного шифра, а второй различим ввиду большего размера входного блока.

Действительно, для случайного 128-битового шифра вероятность появления произвольной выходной разности составляет 2^{-128}, в то время как для шифра SPECTR-128 вероятность $P(12)$ для выходной разности (Δ_0^L, Δ_1^R) равна $\approx 2^{-113}$.

Поэтому с использованием рассмотренной выше разности (Δ_0^L, Δ_1^R) можно отличить шифр SPECTR-128 от случайного шифра. При этом придется осуществить столько операций зашифрования 128-битовых блоков данных, сколько практически неосуществимо для современного уровня вычислительной техники. Например, для 10-раундового варианта шифра SPECTR-128 необходимо выполнить более 2^{94} операций зашифрования 128-битовых блоков данных.

Данный шифр может быть усилен добавлением двух дополнительных раундов шифрования или путем оптимизации блока расширения **E** и операции **G**. Второй вариант позволяет легко уменьшить значение $P(2)$ до величины 2^{-25}.

Дифференциальный криптоанализ шифров SPECTR-H64 и SPECTR-128 является одним из этапов проектирования шифров на основе управляемых операций. На данном этапе в зависимости от полученных значений дифференциальных характеристик и выявленных особенностей распространения разностей необходимо внести изменения с целью улучшения рассматриваемых характеристик, после чего дифференциальный криптоанализ шифра нужно выполнить еще раз.

6.4.1. Стойкость криптосистемы SPECTR-128 с модифицированным блоком расширения

Как отмечалось в предыдущем разделе, вероятность двухраундовой характеристики криптосистемы SPECTR-128 существенно зависит от распределения битов управляющего подблока данных по элементарным переключателям блока управляемых перестановок. При этом данное распределение для блока $\mathbf{P}'$ дополнительно определяется циклическим сдвигом в сторону младших разрядов на 21 бит, для блока $\mathbf{P}''$ — на 43 бита, а также блоком расширения **E**.

Оставляя неизменными значения циклических сдвигов, вероятность двухраундовой характеристики можно уменьшить путем модифицирования таблицы, описывающей блок расширения **E**. При осуществлении такого модифицирования желательно руководствоваться критерием, согласно которому при любом значении управляющего подблока на каждый входной бит блока управляемых перестановок оказывается влияние произвольного управляющего бита не более одного раза.

В табл. 6.8 приводится таблица модифицированного блока расширения **E**, в которой внесены такие изменения. Они сводят к нулю вероятности событий A1, A2 и A3.

Таблица 6.8. *Распределение битов на выходе блока расширения (указаны номера битов, двоичного вектора, поступающего на вход блока* ***E****)*

Левая часть блока E (соответствует входным битам с номерами от 1 до 32)															
33	34	35	36	37	38	39	40	41	42	43	44	62	63	34	60
50	41	52	53	42	61	56	57	61	38	48	55	45	46	47	49
58	59	45	58	62	49	64	63	33	51	52	53	54	55	39	54
26	27	28	29	1	19	10	17	18	31	20	21	22	23	24	25
18	19	20	21	14	23	24	25	26	27	28	29	30	31	32	17
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Правая часть блока E (соответствует входным битам с номерами от 33 до 64)															
35	36	37	43	44	54	55	56	57	58	59	60	61	62	63	64
64	49	50	51	38	39	40	41	42	48	53	45	46	47	52	33
57	46	44	60	43	47	48	50	34	35	36	37	59	56	40	51
32	11	8	9	22	15	16	30	2	3	4	5	6	7	12	13
2	3	4	5	6	7	8	9	10	1	12	13	22	15	16	11
17	18	19	20	21	28	23	24	25	26	27	22	29	30	31	32

Последнее обстоятельство приводит к тому, что другие события, которые вносили до модифицирования существенно меньший вклад в вероятность двухраундовой характеристики и которыми ранее можно было пренебречь, теперь становятся определяющими. Рассмотрим эти события и их вклад.

Событие B. Данное событие включает в себя следующие события:

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{G})}$.
2. На выходе блока $\mathbf{P}'$ с вероятностью $p_3^{(i,t)}$ формируется разность $\Delta_{2|i,t}^{(\mathbf{P}')}$, а на выходе блока $\mathbf{P}''$ с вероятностью $p_1^{(i+k,t)}$ — разность $\Delta_{2|i+k,t}^{(\mathbf{P}'')}$ или на выходе блока $\mathbf{P}'$ с вероятностью $p_3^{(i+k,t)}$ формируется разность $\Delta_{2|i+k,t}^{(\mathbf{P}')}$, а на выходе блока $\mathbf{P}''$ с вероятностью $p_1^{(i,t)}$ — разность $\Delta_{2|i,t}^{(\mathbf{P}'')}$.
3. Нулевая разность, появившаяся в результате событий B1 и B2, а также после выполнения двух операций XOR, проходит через блок управляемых перестановок $\mathbf{P}^{-1}$ с вероятностью 2^{-3}.

Вероятность события B2 равна $p_{1,3}^{(i,i+k)} = p_3^{(i,t)} p_1^{(i+k,t)} + p_3^{(i+k,t)} p_1^{(i,t)}$. В целом, вклад события B в вероятность двухраундовой характеристики составляет $P' \approx 1{,}5 \cdot 2^{-30}$.

Событие C. Данное событие включает в себя следующие события:

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{G})}$.
2. На выходе блока **P′** с вероятностью $p_3^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
3. На выходе блока **P″** с вероятностью $p_1^{(i+k,t)}$ формируется разность $\Delta_{2|i+k,t}^{(\mathbf{P}'')}$ или с вероятностью $p_1^{(i,t)}$ — разность $\Delta_{2|i,t}^{(\mathbf{P}'')}$.
4. После выполнения двух операций XOR разности $\Delta'_{2|i,t} = \Delta_0^{(\mathbf{P}'')} \oplus \Delta_{2|i,i+k}^{(\mathbf{G})} \oplus \Delta_{2|i+k,t}^{(\mathbf{P}'')}$ и $\Delta'_{2|i+k,t} = \Delta_0^{(\mathbf{P}'')} \oplus \Delta_{2|i,i+k}^{(\mathbf{G})} \oplus \Delta_{2|i,t}^{(\mathbf{P}'')}$ "преобразуются" блоком $\mathbf{P}^{-1}$ в нулевую разность с вероятностями $p_4^{(i,t)}$ и $p_4^{(i+k,t)}$, соответственно.

Событие **C** включает появление двух активных битов в разрядах с номерами i и t (или $i + k$ и t) на входе блока $\mathbf{P}^{-1}$ и "аннигиляцию" этой пары активных битов на одном из трех активных элементарных блоков $\mathbf{P}_{2/1}$ внутри блока $\mathbf{P}^{-1}$.

Поскольку событие **C** "симметрично" событию **B**, следовательно, $P' = P(\mathbf{C}) = P(\mathbf{B}) = P''$. То есть вклад событий **B** и **C** в вероятность двухраундовой характеристики $P(2)$ одинаковый.

Событие D. Данное событие включает в себя следующие события:

1. На выходе операции **G** с вероятностью $p_2^{(i,i+k)}$ формируется разность $\Delta_{2|i,i+k}^{(\mathbf{G})}$.
2. На выходе блока **P″** с вероятностью $p_1^{(i)} = 2^{-3}$ формируется разность $\Delta_0^{(\mathbf{P}')}$.
3. На выходе блока **P′** с вероятностью $p_3^{(i+k,t)}$ формируется разность $\Delta_{2|i+k,t}^{(\mathbf{P}')}$ или с вероятностью $p_3^{(i,t)}$ — разность $\Delta_{2|i,t}^{(\mathbf{P}')}$.
4. После выполнения двух операций XOR разности $\Delta'_{2|i,t} = \Delta_0^{(\mathbf{P}'')} \oplus \Delta_{2|i,i+k}^{(\mathbf{G})} \oplus \Delta_{2|i+k,t}^{(\mathbf{P}')}$ и $\Delta'_{2|i+k,t} = \Delta_0^{(\mathbf{P}'')} \oplus \Delta_{2|i,i+k}^{(\mathbf{G})} \oplus \Delta_{2|i,t}^{(\mathbf{P}')}$ "преобразуются" блоком $\mathbf{P}^{-1}$ в нулевую разность с вероятностями $p_4^{(i,t)}$ и $p_4^{(i+k,t)}$, соответственно.

Событие **D** аналогично событию **C**. Особенностью случая **D** является тот факт, что рассматриваются не только два блока управляемых перестановок с симметричной структурой, но и симметрично расположенные активные элементарные переключатели. Вклад события **D** в вероятность $P(2)$, рассчитанный в предполо-

жении, что раундовые подключи являются равномерно распределенными случайными переменными, составляет $P''' \approx 1{,}1 \cdot 2^{-28}$.

Учитывая вклад событий **B, C** и **D** в вероятность $P(2)$, значение вероятности $P(2)$ для двухраундовой характеристики после модифицирования блока расширения **E** имеет вид:

$$P(2) = P' + P'' + P''' \approx 1{,}85 \cdot 2^{-28} \approx 2^{-27}.$$

Таким образом, усовершенствование блока расширения **E** позволило уменьшить вероятность двухраундовой характеристики примерно в 2^8 раза. Соответственно, стойкость к дифференциальному криптоанализу модифицированного шифра SPECTR-128 теперь определяется трехраундовой характеристикой, для которой такое модифицирование блока расширения **E** не является критичным.

То есть, используя значение вероятности трехраундовой характеристики $P(3) \approx 2^{-32}$ для основной модели шифра SPECTR-128, вероятность $P(12)$ прохождения разности $(\Delta^L{}_0, \Delta^R{}_1)$ через все двенадцать раундов равна:

$$P(12) \approx (2^{-32})^4 = 2^{-128}.$$

Полученное значение не превышает значение вероятности появления заданной разности для случайного шифра, поэтому применение рассмотренных дифференциальных характеристик не позволяет использовать метод дифференциального криптоанализа для вскрытия модифицированного варианта шифра SPECTR-128.

Рассмотренный пример, включающий исходную и модифицированную версию алгоритма SPECTR-128, демонстрирует определенный этап проектирования шифра на основе управляемых перестановок, в которой критическое значение имеет структура блока расширения **E**. Так, структура этого блока расширения **E** первоначально выбирается с учетом некоторого общего критерия, после чего делается расчет вероятностей для наиболее значимых дифференциальных характеристик. Затем, на основе выявленных особенностей в распределении битов управляющего подблока данных осуществляется модификация в блоке расширения **E**, приводящая к уменьшению вероятностей указанных характеристик, после чего дифференциальный криптоанализ выполняется еще раз.

6.5. Основные дифференциальные характеристики шифров DDP-S64 и DDP-S128

По сравнению с криптосистемами SPECTR-H64 и SPECTR-128, шифры DDP-S64 и DDP-S128 имеют другую структуру раундового преобразования. Однако ввиду использования блока управляемых перестановок $P_{32/32}^{(L,e)}$ в качестве базового примитива для всех четырех шифров при рассмотрении различных вариантов дифференциальных характеристик можно установить, что наибольшую вероятность

имеют характеристики с малым числом активных битов. Механизм формирования подобных характеристик для первой пары шифров отличается от такового для второй пары, что связано с тем, что в DDP-S64 и DDP-S128 не используются операции **G**, на выходе которых могут формироваться разности с двумя активными битами. Данные шифры также отличаются между собой по механизму прохождения разностей через два раунда, хотя в обоих шифрах наиболее эффективные характеристики связаны с перестановкой одного из активных битов в нужный разряд после прохождения разности через блок управляемых перестановок.

6.5.1. Анализ шифра DDP-S64

Для данного шифра имеет место однораундовая характеристика с разностью (Δ_0^A, Δ_1^B), где Δ_0^A и Δ_1^B — левый и правый подблоки входной разности. Эта разность проходит через один раунд следующим образом (рис. 6.6). Правый подблок, прошедший через одну из операций управляемой перестановки, накладывается с помощью операции суммирования по модулю 2^{64} на левый подблок, внося в него, по крайней мере, один активный бит.

Рис. 6.6. Схема формирования однораундовой характеристики в шифре DDP-S64

Нас будет интересовать случай формирования в левом подблоке одного активного бита. Перед завершением раунда левый подблок Δ_1^A, прошедший через операцию $P_{32/32}^{(A,0)}$, участвует в преобразовании правого подблока с использованием операции вычитания по модулю 2^{32}. Благодаря выполнению операций управляемой

перестановки над обоими подблоками, с вероятностью 2^{-5} активные биты в обоих подблоках оказываются в одноименных разрядах. В результате чего после выполнения операции вычитания активный бит в правом подблоке обнуляется с усредненной вероятностью, близкой к 0,5 (усреднение по всем разрядам). Раунд завершается перестановкой подблоков Δ_1^A и Δ_1^B, формирующей выходную разность, совпадающую с входной разностью.

В данном механизме четыре раза один из активных битов проходит через арифметические операции. При этом активный бит проходит каждую из этих операций без формирования бита переноса с вероятностью $\approx 2^{-1}$. При выполнении четырех операций управляемой перестановки на управляющий вход соответствующего блока управляемых перестановок подается один активный бит. Вероятность того, что при выполнении одной операции управляемой перестановки не генерируются активные биты, приближенно равна 2^{-3} (принимается во внимание тот факт, что через блок управляемых перестановок проходит ненулевая разность). Учитывая также вероятности всех этих событий, связанных с прохождением разности через один раунд, находим вероятность однораундовой характеристики:

$$P(1) \approx (2^{-1})^4 \cdot (2^{-3})^4 \cdot 2^{-5} = 2^{-21}.$$

Более эффективной является двухраундовая характеристика (Δ_1^A, Δ_0^B), которую и следует принимать во внимание при выборе числа раундов шифрования для достижения псевдослучайного преобразования. Схема прохождения этой разности через два раунда показана на рис. 6.7.

Алгоритм прохождения данной разности через два раунда имеет следующий вид.

1. Активный бит в левом подблоке порождает активный бит в правом в конце первого раунда при выполнении операции вычитания по модулю 2^{32}.
2. После этого на вход второго раунда поступает разность (Δ_1^A, Δ_1^B).
3. В начале второго раунда активный бит в левом подблоке обнуляется после выполнения операции сложения по модулю 2^{32}.
4. По левой ветви второго раунда распространяется нулевая разность.

В первом раунде активный бит проходит через три арифметические операции и влияет на выполнение двух операций управляемой перестановки, что определяет вероятность формирования на выходе первого раунда разности (Δ_1^A, Δ_1^B), равной $P' \approx (2^{-1})^3 \cdot (2^{-3})^2 = 2^{-9}$. Во втором раунде три арифметические операции выполняются с участием активных битов. Кроме того, две операции управляемой перестановки выполняются при наличии одного активного бита на управляющем входе. Это определяет вероятность P'' формирования на выходе второго раунда разности (Δ_1^A, Δ_0^B):

$$P'' \approx (2^{-1})^3 \cdot (2^{-3})^2 \cdot 2^{-5} = 2^{-14}.$$

Таким образом, вероятность двухраундовой характеристики равна

$$P(2) \approx P' \cdot P'' \approx 2^{-9} \cdot 2^{-14} = 2^{-23}.$$

Вкладом однораундовой характеристики можно пренебречь, поскольку он равен $(P(1))^2 \approx 2^{-42} << 2^{-23}$. Число раундов R, для которого шифр DDP-S64 неотличим от случайного преобразования, при выполнении дифференциального анализа можно определить из следующего соотношения: $2^{-64} \geq P^{r/2}(2) \approx 2^{-23r/2}$, откуда легко получить $r \geq 6$. Для получения некоторого дополнительного запаса по стойкости можно добавить еще два раунда и рекомендовать значения $r \geq 8$.

Рис. 6.7. Схема формирования двухраундовой характеристики в шифре DDP-S64

6.5.2. Анализ шифра DDP-S128

Для данного шифра наиболее эффективной является двухраундовая характеристика с разностью $(\Delta_1^A, \Delta_0^B, \Delta_0^C, \Delta_1^D)$. Эта разность проходит через два раунда в соответствии с рис. 6.8. С вероятностью, равной единице, разности Δ_1^A и Δ_1^D проходят первый раунд и после перестановки подблоков переходят в разности Δ_1^B и Δ_1^C, соответственно. Таким образом, входная разность первого раунда $(\Delta_1^A, \Delta_0^B, \Delta_0^C, \Delta_1^D)$ с вероятностью, равной единице, преобразуется во входную разность второго раунда $(\Delta_0^A, \Delta_1^B, \Delta_1^C, \Delta_0^D)$. Во втором раунде два внутренних подблока с одним активным битом суммируются по модулю 2^{-32} два раза, причем на разных шагах преобразования подблоков *B* и *C*.

При выполнении каждой операции сложения выходная разность с вероятностью 2^{-6} не будет содержать активных битов, поскольку с вероятностью 2^{-5} активные биты в разных подблоках оказываются в одноименных разрядах, и с вероятностью 2^{-1} не порождается бит переноса. В этом случае операции XOR, выполняемые над подблоками *A* и *D*, не добавляют активных битов в разности Δ^A и Δ^D.

Кроме того, каждый из активных битов участвует в выполнении трех различных операций управляемой перестановки. То есть необходимо рассматривать шесть независимых событий, каждое из которых состоит в том, что в соответствующих блоках управляемых перестановок не порождается ни одной пары активных битов с вероятностью, приблизительно равной 2^{-3}. Необходимо также учесть, что две операции сложения внутренних подблоков с раундовыми подключами с вероятностью, близкой к 2^{-1}, могут породить дополнительный активный бит (за счет бита переноса).

Принимая во внимание все эти элементарные события, которые связаны с механизмом формирования двухраундовой характеристики, мы получаем следующее значение для ее вероятности:

$$P(2) \approx (2^{-6})^2 \cdot (2^{-3})^6 \cdot (2^{-1})^2 = 2^{-32}.$$

Определим число раундов, для которого шифр DDP-S128 неотличим при дифференциальном анализе от случайного преобразования. Для этого воспользуемся соотношением $2^{-128} \geq P^{r/2}(2) \approx 2^{-16r}$, из которого следует, что $r \geq 8$.

Суммируя результаты дифференциального криптоанализа ряда шифров, основанных на использовании управляемых перестановок, можно отметить следующее общее свойство. Активный бит, поступающий на вход операции управляемой перестановки, не вносит вклад в лавинный эффект. Свойство лавинного эффекта, вносимого операцией управляемой перестановки, связано с наличием активных битов в подблоках, используемых для формирования управляющих

векторов. В то же время, если рассматривать не биты разностей, а биты самих данных, то легко видеть, что каждый бит на входе блока управляемых перестановок влияет на все выходные биты этой операции.

Рис. 6.8. Механизм формирования двухраундовой характеристики в шифре DDP-S128

6.6. Оценка стойкости шифров COBRA-F64a и COBRA-F64b

В алгоритмах COBRA-F64a и COBRA-F64b, также как и во всех ранее рассмотренных шифрах, активные биты вносят наибольший вклад в лавинный эффект тогда, когда они попадают на управляющий вход блока управляемых перестановок. Это приводит к тому, что наибольшую вероятность имеют дифференциальные характеристики с малым числом активных битов. При этом надо найти такие характеристики, в схемах формирования которых промежуточные разности также имеют минимальное число активных битов. Действительно, наличие одного активного бита на управляющем входе блока управляемых перестановок дает множитель 2^{-3} в выражении для вероятности, а наличие дополнительного активного бита на входе блока управляемых перестановок дает множитель 2^{-5} в случае, когда ожидается событие, связанное с его переходом в заданный разряд (что требуется для того, чтобы обнулить пару активных битов при выполнении операций сложения — в противном случае число активных битов лавинообразно нарастает). Для алгоритмов COBRA-F64a и COBRA-F64b удается построить характеристики, в которых все промежуточные разности содержат не более одного активного бита в правом и левом подблоке. Видимо, такие характеристики обладают максимальной вероятностью.

Схемы формирования дифференциальных характеристик в шифрах COBRA-F64a и COBRA-F64b приведены на рис. 6.9. В обоих шифрах наиболее эффективные характеристики связаны с разностью (Δ_0^L, Δ_1^R), проходящей через два или три раунда. В одном раунде шифра COBRA-F64a активный бит разности переносится один раз из правой ветви в левую при выполнении операции XOR. Поэтому, для того чтобы вернуться к исходной разности, в криптосхеме COBRA-F64a нужно выполнить три раунда, причем второй раунд связан с событием попадания активных битов в левом и правом подблоках в одноименный разряд.

Отличием шифра COBRA-F64b от шифра COBRA-F64a является то, что в одном раунде активный бит разности Δ_1^R переносится из правой ветви в левую два раза, что определяет возможность сохранения разности (Δ_0^L, Δ_1^R) на выходе первого раунда. Однако после первого раунда осуществляется перестановка, преобразующая разность (Δ_0^L, Δ_1^R) в разность (Δ_1^L, Δ_0^R), поэтому в схему формирования дифференциальной характеристики включен второй раунд, в котором с вероятностью 2^{-3} разность Δ_1^L проходит через левую ветвь без порождения дополнительных активных битов в правом и левом подблоках. Перестановка подблоков после второго раунда приводит к формированию исходной разности (Δ_0^L, Δ_1^R).

Рассматривая для входной разности (Δ_0^L, Δ_1^R) вероятности событий, связанных с формированием трехраундовой характеристики шифра COBRA-F64a, легко

определить, что вероятность формирования разности (Δ_1^L, Δ_1^R) на входе второго раунда равна:

$$P' = P\{(\Delta_0^L, \Delta_1^R) \to (\Delta_1^L, \Delta_1^R)\} = 2^{-4}.$$

Данная вероятность определяется событием прохождения активного бита через операцию вычитания в правой ветви, порождением активного бита в левом подблоке после выполнения операции XOR и выполнением одной операции управляемой перестановки при наличии активного бита в управляющем подблоке.

Рис. 6.9. Схема формирования трехраундовой характеристики в шифре COBRA-F64a (*а*) и двухраундовой характеристики в COBRA-F64b (*б*)

Разность (Δ_1^L, Δ_1^R) после прохождения второго раунда (и операции перестановки) переходит в разность (Δ_1^L, Δ_0^R) с вероятностью

$$P'' = P\{(\Delta_1^L, \Delta_1^R) \to (\Delta_1^L, \Delta_0^R)\} = 2^{-1} \cdot 2^{-1} \cdot 2^{-3} \cdot 2^{-5} = 2^{-10}.$$

После прохождения второго раунда, включая и операции перестановки, разность (Δ_1^L, Δ_0^R) трансформируется в разность (Δ_0^L, Δ_1^R) с вероятностью

$$P''' = P\{(\Delta_1^L, \Delta_0^R) \to (\Delta_0^L, \Delta_1^R)\} = 2^{-1} \cdot 2^{-3} \cdot 2^{-3} = 2^{-7}.$$

Теперь для вероятности трехраундовой характеристики получаем $P(3) = P'P''P''' = 2^{-21}$. Используя это значение можно установить, что при числе раундов $r \geq 10$ шифр COBRA-F64a неотличим от случайного преобразования при дифференциальном криптоанализе с использованием указанной характеристики. Таким образом, 16-раундовый алгоритм COBRA-F64a обеспечивает достаточный запас по стойкости к дифференциальному анализу. Отметим, что трехраундовая характеристика для этой криптосистемы может использовать в качестве входных разностей также (Δ_1^L, Δ_1^R) и (Δ_1^L, Δ_0^R). Данные варианты трехраундовой характеристики имеют такое же значение вероятности, что и рассмотренная выше характеристика. Однако использование трех вариантов не является полностью эквивалентным. Например, при числе раундов, которое не является кратным трем, прохождение разности через последний или последний и предпоследний раунды соответствуют ее прохождению через неполную схему формирования дифференциальной характеристики. Из последней, в зависимости от варианта входной разности, исключается один или два последних раунда, которые характеризуются различными значениями вероятностей. Если исключается один раунд, то с ним может быть связана вероятность 2^{-7}, 2^{-10} или 2^{-4}. Если же исключаются два раунда, то возможно исключение пар, соответствующих вероятностям $p_1 = 2^{-7} \cdot 2^{-10}$, $p_2 = 2^{-10} \cdot 2^{-4}$ или $p_3 = 2^{-4} \cdot 2^{-7}$.

Эти примеры показывают, что для разных вариантов отношение значений получаемых вероятностей для всего шифра может составить 2^6. Данное обстоятельство полезно учитывать при выборе минимального числа раундов с целью повышения скорости шифрования.

Следуя схеме формирования двухраундовой характеристики COBRA-F64b, в которой используется входная разность (Δ_0^L, Δ_1^R), легко увидеть, что вероятность формирования разности (Δ_1^L, Δ_0^R) на входе второго раунда равна:

$$P' = P\{(\Delta_0^L, \Delta_1^R) \to (\Delta_1^L, \Delta_0^R)\} = 2^{-7}.$$

Данная вероятность определяется прохождением активного бита через операцию вычитания в левой ветви и порождением в ней одного активного бита (сомножи-

тель $\approx 2^{-1}$), совпадением номеров разрядов, соответствующих активным битам правого и левого подблока (сомножитель 2^{-5}) и прохождением активных битов через операцию суммирования в левой ветви, приводящей к обнулению активного бита в левом подблоке (сомножитель 2^{-1}).

При прохождении через второй раунд (включая операцию перестановки подблоков) разность (Δ_1^L, Δ_0^R) с вероятностью $P'' = P\{(\Delta_1^L, \Delta_0^R) \to (\Delta_0^L, \Delta_1^R)\} = 2^{-5}$ трансформируется в разность (Δ_0^L, Δ_1^R). При этом вероятность двухраундовой характеристики равна:

$$P(2) = P'P'' = 2^{-12}.$$

Другой вариант двухраундовой характеристики связан с прохождением разности (Δ_1^L, Δ_0^R) через два раунда. На самом деле оба варианта внутри отдельных раундов включают одни и те же механизмы. Некоторое различие в выборе входной разности имеет значение для случая нечетного числа раундов шифрования. Отношение значений вероятностей, получаемых для нечетного числа раундов шифрования в зависимости от выбора одного из этих двух вариантов разности, составляет 2^{12}. Аналогичное замечание можно сделать также и для шифров SPECTR-H64, SPECTR-128 и др.

Для некоторых шифров вероятность наблюдения некоторой ожидаемой разности на выходе алгоритма шифрования может быть повышена путем выбора определенной входной разности, отличной от разностей, соответствующих наиболее эффективным характеристикам. Повышение вероятности достигается принудительным заданием необходимой разности с активными битами в заданных разрядах при выполнении самого первого раунда. Например, это может быть реализовано в случае шифра COBRA-F64b, в котором входная разность $(\Delta_{1|32}^L, \Delta_{1|32}^R)$ на выходе первого раунда трансформируется в разность (Δ_1^L, Δ_0^R) с вероятностью, равной единице, а дальнейшее ее прохождение рассматривается в соответствии с двухраундовой характеристикой.

При этом можно легко получить следующие формулы для вероятности появления разности (Δ_1^L, Δ_0^R) на выходе алгоритма:

- $P(r) = P\{(\Delta_{1|32}^L, \Delta_{1|32}^R) \to (\Delta_1^L, \Delta_0^R)\} = 2^{-6r+5}$ для четного r;
- $P(r) = P\{(\Delta_{1|32}^L, \Delta_{1|32}^R) \to (\Delta_1^L, \Delta_0^R)\} = 2^{-6(r-1)}$ для нечетного r.

Из условия $P(r) \le 2^{-64}$ минимальное рекомендуемое число раундов шифрования при использовании алгоритма COBRA-F64b: $r_{\min} = 12$. В случае $r \ge r_{\min}$ преобразование, задаваемое данным алгоритмом, является неотличимым от случайного преобразования.

Приведем итоговую таблицу по дифференциальным свойствам рассмотренных алгоритмов (табл. 6.9).

Таблица 6.9. *Сравнительные данные для дифференциальных характеристик блочных шифров, использующих управляемые операции*

Шифр	***r***	**Характеристика**		***P*(*r*)** **
		Разность	**Вероятность**	
SPECTR-H64	12	(Δ_0^L, Δ_1^R)	$P(2) \approx 1.5 \cdot 2^{-15}$	2^{-87}
SPECTR-H64	12	(Δ_0^L, Δ_1^R)	$P(3) \approx 2^{-28}$	2^{-112}
SPECTR-128	12	(Δ_0^L, Δ_1^R)	$P(2) \approx 1.1 \cdot 2^{-19}$	2^{-113}
SPECTR-128*	12	(Δ_0^L, Δ_1^R)	$P(3) \approx 2^{-32}$	2^{-128}
SPECTR-128*	12	(Δ_0^L, Δ_1^R)	$P(2) \approx 2^{-27}$	2^{-162}
COBRA-F64a	16	(Δ_1^L, Δ_1^R)	$P(3) = 2^{-21}$	$< 2^{-105}$
COBRA-F64b	20	(Δ_0^L, Δ_1^R)	$P(2) = 2^{-12}$	2^{-120}
DDP-S64	10	(Δ_1^A, Δ_0^B)	$P(2) \approx 2^{-23}$	2^{-115}
DDP-S128	12	$(\Delta_1^A, \Delta_0^B, \Delta_0^C, \Delta_1^D)$	$P(2) \approx 2^{-32}$	2^{-192}

*модифицированный вариант;

**вклад характеристики в вероятность прохождения разности через *r* раундов.

Выполненный дифференциальный криптоанализ является частью комплексных исследований, осуществляемых при проектировании шифра с целью оптимизации отдельных примитивов, а также обоснования его криптографической стойкости.

В частности, было показано, что таблица распределения управляющих битов **E** является критическим местом. Ее составление осуществляется в несколько этапов:

1. Первоначально формулируются общие критерии построения таблицы.
2. Выполняется расчет дифференциальных характеристик, в результате чего выявляются номера управляющих битов, вклад которых в вероятность характеристик существенно превышает средние показатели.
3. Осуществляется модифицирование таблицы путем изменения позиции выявленных битов.
4. Повторно выполняется дифференциальный криптоанализ.
5. Выполняются эксперименты для определения вероятностей дифференциальных характеристик, соответствующих одному или нескольким раундам шифрования.

6. Теоретические результаты сопоставляются с экспериментальными данными.
7. Если теория и эксперимент согласуются между собой, тогда делается вывод о том, что в теоретической модели учтены основные механизмы формирования разностей, а оценки, полученные в результате проведенного криптоанализа, заслуживают доверия.

Таким образом, для более полного завершения дифференциального анализа рассмотренных шифров необходимо выполнение последних трех пунктов. При этом может оказаться, что экспериментальные значения вероятностей окажутся существенно больше по сравнению с теоретическими значениями. Это будет означать то, что в моделях формирования характеристик не учитываются некоторые механизмы, вносящие существенный вклад.

Несомненно, что инициативный читатель сможет самостоятельно завершить цикл дифференциального анализа и при этом найдет необходимые уточнения моделей, использованных выше для оценки стойкости шифров SPECTR-H64, SPECTR-128, COBRA-F64a, COBRA-F64, DDP-S64, DDP-S128.

6.7. Атака на основе аппаратных ошибок

При сопоставлении различных алгоритмов шифрования с целью выбора криптосистемы для конкретного приложения представляет интерес рассмотрение таких видов нападений, которые дают криптоаналитику значительно больше возможностей, чем знание алгоритма шифрования и большого числа соответствующих друг другу пар открытых и зашифрованных текстов, включая специально подобранные тексты.

Одним из видов нападений, предоставляющих криптоаналитику дополнительные возможности, является атака на основе генерации ошибок шифратора (или микропроцессора в случае программных шифров), обусловливаемых влиянием внешнего воздействия. Данный вид атаки обладает высокой эффективностью для большого числа известных и практически используемых криптосистем.

Вид нападений, связанных с ожиданием или с целенаправленной генерацией аппаратных ошибок устройства шифрования, является весьма специфичным, однако в связи с массовым использованием интеллектуальных электронных карточек его рассмотрение приобрело важное практическое значение. Очевидно, что случай ожидания ошибок отличается от случая генерации ошибок только тем, что самопроизвольные ошибки случаются крайне редко, а появление ошибок в момент криптографического преобразования заданных входных блоков является еще более редким событием. Остановимся на случае, когда криптоаналитик имеет возможность целенаправленно производить случайные ошибки шифрующего устройства при преобразовании некоторых выбранных им блоков данных.

Таким образом, будем рассматривать модель, в которой криптоаналитику предоставляется дополнительная возможность оказывать на шифратор с установ-

ленным в нем ключом некоторое внешнее воздействие с целью вызвать сбои в работе аппаратуры и полученные выходные данные сопоставлять с выходными данными, полученными без сбоев. При этом криптоаналитик может подавать на вход специально подобранные тексты. Предполагается, что доступ к области памяти, содержащей ключевые параметры и алгоритм шифрования, не может быть осуществлен без стирания находящейся в ней информации. Это предположение основано на том, что современная технологическая база позволяет изготовление микроэлектронных устройств с защищенной памятью. Таким образом, криптоаналитик может обладать устройством шифрования с введенным в него секретным ключом, но не может осуществить его непосредственное считывание, а также внести целенаправленные изменения в алгоритм шифрования.

Интенсивность внешнего воздействия на шифратор (например, нагревание, высокочастотное или ионизирующее излучение) подбирается криптоаналитиком такого уровня, чтобы в процессе шифрования одного блока данных вызвать одну или несколько одиночных ошибок в регистрах микропроцессора. Невозможно заранее задать место генерации ошибки, но в достаточно длинной серии экспериментов в некоторых из опытов будут иметь место ошибки заданного типа, например инверсия одного из двоичных разрядов регистра после выполнения i-го раунда шифрования и перед $(i + 1)$-м. Таким образом, ожидаемая ошибка характеризуется как пространственной, так и физической локализацией. Для многих известных шифров имеется принципиальная возможность распознавания опытов, в которых произошло ожидаемое событие, путем подбора входных блоков со специальной структурой и анализа различий в структуре выходного блока, вызванных ошибками. Вычислительная сложность распознавания может изменяться в широких пределах и зависит от конкретного механизма шифрования. В некоторых шифрах ряд ошибок распознаются тривиально.

Множество сбоев аппаратуры в процессе шифрования можно разделить на два основных типа:

- ❒ ошибки в области данных;
- ❒ ошибки в области команд.

В принципе, ошибки второго типа могут привести непосредственно к формированию на выходе шифратора участков ключа шифрования, однако вероятность событий, при которых исполняемый код после случайного модифицирования будет соответствовать некоторой полезной для криптоаналитика программе, является очень низкой. Мы будем рассматривать случай генерации ошибок в области данных. (Результаты опытов, в которых имела место генерация ошибок второго типа, криптоаналитик может просто отбрасывать.) Например, могут использоваться случайные ошибки в 8-битовых участках регистров, содержащих значения преобразуемых 32-битовых подблоков. Причем ошибки ожидаются в определенный промежуток времени выполнения процедуры шифрования (обычно в пределах выполнения одной или нескольких команд микропроцессора).

Будем считать, что экспериментально подобрана такая интенсивность внешнего воздействия, при которой в среднем генерируется одна ошибка за полное время шифрования одного блока. Сделаем приближенную оценку среднего количества опытов, которые необходимо осуществить для формирования одной ожидаемой ошибки, т. е. ошибки, которая возникает в заданном регистре, содержащем некоторый подблок данных, на заданном шаге преобразования. Будем полагать, что вероятность генерации ошибки пропорциональна времени облучения соответствующих регистров, находящихся в благоприятном для возникновения ошибки состоянии, и количеству двоичных разрядов, в пределах которых мы ожидаем ошибку. При этом правдоподобном допущении легко оценить вероятность генерации ожидаемой ошибки p в одном опыте, т. е. при шифровании одного блока

$$p_{512} \approx p_d/RZ,$$

где p_d — вероятность генерации ошибки в области данных за время выполнения процедуры шифрования одного блока данных, R — число раундов преобразования, Z — число элементарных шагов преобразования в одном раунде. Поскольку ожидается ошибка в течение времени выполнения какого-то одного конкретного шага шифрования, то вероятность появления нужной ошибки обратно пропорциональна общему числу шагов преобразования. (Это приводит к тому, что в случае 512-байтовых шифров значение p существенно ниже по сравнению с 64- и 128-битовыми шифрами.) Криптоаналитик может использовать импульсное излучение, синхронизированное с процессом шифрования. В этом случае может быть использована более высокая интенсивность излучения, а длительность импульсного воздействия может быть выбрана равной продолжительности времени нахождения преобразуемых подблоков данных в благоприятном для криптоаналитика состоянии.

При использовании импульсных воздействий и синхронизации с процессом шифрования указанные вероятности можно повысить на один два порядка. Ожидаемая трудоемкость формирования достаточного количества нужных ошибок не представляется высокой ни в одном из этих случаев. Нет принципиального различия между случаями импульсного и стационарного воздействия на шифратор. Главным является то, что возможность целенаправленной генерации случайных аппаратных ошибок в регистрах данных является реальной. В дальнейшем мы будем считать, что используется стационарное воздействие на устройство шифрования.

В случае программной реализации шифра как совокупности повторений типового цикла необходимо использовать счетчик числа раундов шифрования. Это требует учесть возможность генерации ошибки в регистре, содержащем текущее число выполненных циклов. Для предотвращения завершения шифрования после выполнения малого числа раундов шифрования (например, одного раунда) условие выхода из цикла должно быть составлено соответствующим образом. Наиболее надежным способом предотвращения преждевременного завершения процедуры шифрования является повторение процедуры раундового шифрования

R раз, хотя это увеличивает размер программы шифрования. Рассмотрим стойкость некоторых скоростных шифров к этому виду атаки.

6.7.1. Криптоанализ шифра RC5

Шифр RC5 обеспечивает высокую скорость шифрования данных и при использовании 12 и более раундов преобразования является стойким к известным методам криптоанализа при условии, что обеспечивается целостность алгоритма преобразования. Однако в случае, когда у криптоаналитика имеется возможность формирования случайных аппаратных ошибок, криптосистема RC5 оказывается уязвимой. Рассмотрим стойкость шифра RC5 к этому виду атаки.

В шифре RC5 предусматривается возможность выбора длины ($2n$) входного блока и различного числа раундов шифрования (R). В данной криптосистеме используются предвычисления, которые обеспечивают формирование по секретному ключу расширенного ключа шифрования в виде последовательности n-битовых подключей $S_0, S_1, S_2, ..., S_{2r+1}$.

Шифр RC5 описывается следующим псевдокодом:

$A := A + S_0 \pmod{2^n}$,

$B := B + S_1 \pmod{2^n}$,

for $i = 1$ to R do

$$A := (A \oplus B)^{<B<} + S_{2i} \pmod{2^n}, \tag{6.1}$$

$$B := [(B \oplus A)^{<A<}] + S_{2i+1} \pmod{2^n}, \tag{6.2}$$

где A и B — левый и правый n-битовые подблоки данных, а запись "$W^{<b<}$" обозначает циклический сдвиг влево слова W на b бит. Будем считать, что экспериментально подобрана такая интенсивность внешнего воздействия, при которой в среднем генерируется одна ошибка за полное время шифрования одного блока. За сравнительно малое число опытов можно сформировать ошибку в регистре, содержащем подблок A после выполнения преобразования по формуле (6.1) при $i = R$. Тот факт, что имела место ошибка, на данном шаге легко распознается по структуре блока шифртекста, полученного из данного входного блока без введения ошибок ($C = A \parallel B$) и при введении ошибок ($\widetilde{C} = \widetilde{A} \parallel \widetilde{B}$).

Из (6.2) легко получить соотношение

$$A \oplus \widetilde{A} = \{(\widetilde{B} - S_{2R+1})^{>\widetilde{A}>} \oplus (B - S_{2R+1})^{>A>}\} \bmod 2^n . \tag{6.3}$$

В (6.3) неизвестным является только S_{2R+1}. С высокой вероятностью можно получить $A \bmod 2^5 = \widetilde{A} \bmod 2^5$. В этом случае выражение (6.3) преобразуется к виду

$$(A \oplus \widetilde{A})^{<A<} = \{(\widetilde{B} - S_{2R+1}) \oplus (B - S_{2R+1})\} \bmod 2^n . \tag{6.4}$$

используя последнее соотношение, легко вычислить часть подключа S_{2R+1}. Формируя различные ошибки можно определить полное значение подключа S_{2R+1}. После этого, формируя ошибки в подблоке B после выполнения преобразований, заданных формулой (6.2) и соответствующих раунду $i = R - 1$, аналогичным способом можно вычислить подключ S_{2R}. Формирование данных ошибок легко распознается при известном подключе S_{2R+1}, который дает возможность восстановить значение подблока B после $(R - 1)$-го раунда. Действуя последовательно, этим способом легко вычислить подключи $S_{2R-1}, S_{2R-2}, ..., S_0$. Для значения $R = 10 \div 30$ трудоемкость вычисления всех подключей не превосходит 10^8 операций. Представляет интерес тот факт, что атака на основе генерации аппаратных ошибок весьма эффективна для раскрытия гибких криптосистем, в которых алгоритм шифрования формируется в зависимости от секретного ключа, т. е. является неизвестным криптоаналитику.

6.7.2. Криптоанализ гибкого шифра

В работе [98] описан гибкий R-раундовый 64-битовый алгоритм, в котором в зависимости от ключа шифрования формируются примерно 10^{16R} неэквивалентных модификаций криптоалгоритма. Каждая из потенциально реализуемых модификаций криптоалгоритма использует механизм выборки подключей в зависимости от преобразуемых данных (data-dependent subkey selection). Отражая этот факт, будем называть данный алгоритм DDSS-1. Этот шифр включает предвычисления, с помощью которых исходный секретный ключ преобразуется в расширенный ключ шифрования, представляемый в виде совокупности 32-битовых подключей Q_i, где $i = 0, 1, \ldots, 255$. На рис. 6.10 показана структура одного раунда зашифрования в криптосистеме DDSS-1.

На рисунке используются следующие обозначения:

- "$>c_i>$" — операция циклического сдвига вправо на c_i бит;
- "$*_j$" — одна из трех возможных операций:
  - XOR;
  - сложение по модулю 2^{32};
  - вычитание по модулю 2^{32}.

На этапе предвычислений в зависимости от секретного ключа в разных раундах шифрования в идентичных позициях устанавливаются независимые значения c_i и "$*_j$", что определяет большое число различных модификаций криптоалгоритма. Сокращение SS обозначает процедуру выборки подключа. Выходом функции SS является значение текущего выбираемого 32-битового подключа.

Промежуточные значения преобразуемых подблоков данных обозначим через X^i, Y^j. Будем полагать, что выбрано такое внешнее воздействие на шифратор, при котором за время шифрования одного блока данных в среднем формируется одна

Рис. 6.10. Один раунд в шифре DDSS-1

одиночная ошибка. Для получения ошибки в подблоках X^i, Y^j на заданных шагах преобразования потребуется осуществить шифрование определенного количества входных блоков, часть из которых можно выбрать одинаковыми. Вначале зашифруем открытый текст P_1 и получим шифртекст C_{10}. Затем, при наличии внешнего воздействия, обусловливающего формирование одиночных ошибок, многократно осуществим шифрование блока P_1. В этих экспериментах будем фиксировать шифртексты, в которых при неизменных значениях подблоков

(x_3^7, x_2^7, x_1^7) значение в подблоке x_4^7 получено с ошибкой. Обозначим через C_{11}, C_{12}, ..., C_{1n} шифртексты, соответствующие первому открытому тексту P_1 и наличию одиночной ошибки только в подблоке x_4^7. Теперь мы можем вычислить разности подключей

$$\delta_{x_4^7 \tilde{x}_4^7} = Q_{x_4^7} \overline{*}_8 Q_{\tilde{x}_4^7}, \tag{6.5}$$

где x_4^7 — подблок, не содержащий ошибок, и $\tilde{x}_4^7$ — подблоки, содержащие ошибки (подблоки $\tilde{x}_4^7$ являются частью шифртекстов C_{11}, C_{12}, ..., C_{1n}, подблок x_4^7 — частью C_{10}), а $\overline{*}_8$ — есть групповая операция, обратная операции $*_8$).

В соответствии с нашей схемой атаки все подблоки x_4^7 и $\tilde{x}_4^7$ соответствуют одному и тому же открытому тексту P_1.

Обозначим разности как δ_{ij}, где $i = x_4^7$ и $j = \tilde{x}_4^7$. Если i является фиксированным, то общее число таких разностей равно $n = 255$. Нам нет необходимости ожидать появления всех возможных значений j для данного i. Чтобы определить все возможные разности подключей можно взять другие открытые тексты P_2, P_3, ..., P_n и наблюдать соответствующие им шифртексты C_{20}, C_{30}, ..., C_{n0}. Поскольку алгоритм шифрования задает псевдослучайные значения x_4^7 в этих шифртекстах, то в среднем мы должны взять $256(1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{256}) \approx 1618$ открытых текстов, чтобы получить все возможные значения (их число равно 256) подблока x_4^7. Теперь обозначим открытые тексты, соответствующие различным значениям x_4^7, как P_1, P_2, ..., P_{256}. Зашифруем каждый из текстов P_1, P_2, ..., P_{256} в условиях формирования аппаратных ошибок n и получим наборы шифртекстов C_{l1}, C_{l2}, ..., C_{ln}, где $l = 2, 3, ..., 256$, которые содержат ошибки только в подблоке x_4^7.

Вычисляемые разности $\delta_{i,j} = Q_{x_4^7} \overline{*}_8 Q_{\tilde{x}_4^7}$, соответствующие открытым текстам P_2, P_3, ..., P_{256}, будут содержать произвольные пары значений индексов. Это позволяет воспользоваться групповым свойством операции "$*_8$" и вычислить 256 различных разностей, которые соответствуют одному и тому же значению индекса i. Для этого можно воспользоваться формулой

$$\delta_{ij} = \delta_{ik} *_8 \delta_{kj}. \tag{6.6}$$

Легко видеть, что соотношение (6.6) при наличии полного набора значений подблока x_4^7 позволяет найти такое i_0, которое дает полный набор разностей $\delta_{i_0 j}$, $j = 0, 1, 2, ..., 255$. Это обеспечивает возможность вычислить все подключи путем

подбора значения подключа Q_{i_0}. Имеется только 2^{32} различных вариантов, и их можно перебрать достаточно быстро. Однако прямая проверка вариантов по известным значениям открытого текста и шифртекста затруднена тем, что неизвестны операции "$*_1$" ..."$*_8$" и "$> c_1 >$" ..."$> c_{12} >$". Предварительно необходимо их определить. Это можно сделать по частям, используя тот факт, что при угадывании правильной комбинации операций вычисляемые разности подключей δ_{ij} будут зависеть только от индексов (i, j) и не будут зависеть от входного текста. Имеется 93 различных комбинаций операций "$*_8$" и "$> c_{12} >$", поэтому легко подобрать правильные значения обеих операций, которые удовлетворят указанному выше условию.

После определения операций "$*_8$" и "$> c_{12} >$" мы можем перейти к определению расширенного ключа. Выберем некоторое значение Q_{i_0}. Это сразу задает все подключи ($Q_0, Q_1, ..., Q_{255}$), поскольку разности $\delta_{i_0 j}$, $j = 0, 1, 2, ..., 255$, мы уже определили. А вычислив расширенный ключ и зная операции "$*_8$" и "$> c_{12} >$", мы можем вычислить Y^4 (а также подблок y_4^4) для любого данного открытого текста P и определить подключ $Q_{y_4^4}$. Затем зашифровываем текст P в условиях формирования аппаратных ошибок и наблюдаем шифртекст ($\tilde{X}^7 \parallel \tilde{Y}^6$). При правильно угаданном подключе Q_{i_0} мы с достаточно большой вероятностью обнаружим одиночную ошибку в подблоке y_4^4. Если текущее значение Q_{i_0} ложно, появление одиночной ошибки в подблоке Y^4 имеет малую вероятность, т. к. последний будет вычисляться с использованием ложного значения подключа $Q_{x_4^7}$. Этот факт можно использовать как критерий правильного выбора значения Q_{i_0}.

Зная истинный расширенный ключ, легко определить остальные неизвестные операции. Для определения операции "$*_7$" сформируем ошибку в подблоке y_4^4 при условии отсутствия ошибок в (y_3^4, y_2^4, y_1^4). Такое событие мы можем отобрать, поскольку подблок Y^4 может быть вычислен при известном Y^6. Если мы угадаем правильно "$*_7$", то значение X^6, вычисленное по известным X^7 и $Q_{y_4^4}$, является одинаковым и в случае наличия, и в случае отсутствия ошибок для одного и того же текста P. Это является критерием распознания истинной операции "$*_7$", независимо от значений операций на предыдущих шагах шифрования. Аналогичным способом мы можем определить последовательно операции "$*_6$", "$*_5$" и т. д., пока не будут определены все операции во всех раундах (для многораундовой криптосхемы DDSS-1 приведенная выше атака относится к последнему раунду). Заметим, что нам не надо каждый раз осуществлять шифрование при одновременной генерации ошибок, поскольку требуемые случаи мы найдем в уже имеющемся наборе экспериментальных данных. Это как раз те случаи, которые были ненужными для предыдущих шагов атаки. Теперь они могут быть исполь-

зованы, так как уже имеется возможность вычислять предыдущие значения подблоков X^i, Y^j. Легко заметить, что вычислительно наиболее сложным этапом атаки является подбор правильного значения подключа Q_{i_0}, что требует рассмотрения в среднем 2^{31} вариантов. Этот этап требует выполнения в среднем около $\approx 2^{32} \cdot l_0$ операций, где l_0 — число попыток, которые необходимы для получения ошибки в нужном месте. Существенное повышение стойкости (например, до значения 10^{30}) к данному виду атаки за счет увеличения значения l_0 означает специальную реализацию алгоритма DDSS-1, поэтому сам по себе этот шифр не может считаться стойким к атаке на основе генерации аппаратных ошибок.

6.7.3. Стойкость алгоритма ГОСТ 28147-89

Российский стандарт шифрования ГОСТ 28147-89 является примером широко используемой блочной криптосистемы, стойкость которой определяется как секретностью ключа, так и секретностью таблиц подстановок. Эта криптосистема аналогична по структуре шифру DES и использует 256-битовый секретный ключ и секретные блоки подстановок размера 4×4. Полное преобразование включает 32 раунда шифрования. Атака алгоритма ГОСТ 28147-89 с использованием генерации аппаратных ошибок предполагает вычисление 32-битовых подключей в каждом раунде и секретных таблиц подстановки в последнем раунде. Общее число вариантов таблиц подстановок составляет $(16!)^8$, но, несмотря на столь большое число, они легко вычисляются. Слабость данного алгоритма к атаке на основе аппаратных ошибок состоит в использовании операции сложения по модулю 2^{32} и в малом размере блоков подстановки. Это делает весьма эффективным атаку, основанную на использовании лавинного эффекта, обусловленного битом переноса. Для осуществления такой атаки достаточно двух зашифрованных сообщений (одного с ошибками, а другого без ошибок), выработанных из одного и того же исходного сообщения размером около 10^5 байт. При этом исходное сообщение может быть неизвестным для атакующего. Трудоемкость такой атаки не превышает 10^{10} операций.

6.7.4. Стойкость шифров на основе псевдослучайной выборки подключей

Возможность формирования случайных ошибок позволяет атакующему использовать следующий прием. Вместо того чтобы рассматривать запутанные процедуры шифрования с большим числом неизвестных подключей, изучается различие результатов преобразования некоторого неизвестного промежуточного подблока, полученных до и после внесения ошибки. Причем ошибка вносится в другой подблок данных, который влияет на ход преобразования первого. Сравнение результатов преобразования до и после внесения ошибки позволяет составить необходимую статистику для идентификации всех неизвестных параметров преобразования на выделенном шаге шифрования. Поскольку на отдельных ша-

гах преобразования количество различных значений неизвестных параметров сравнительно невелико, то их можно вычислить простым перебором, причем это можно сделать как для определения численных значений подключей, так и для определения неизвестных операций и процедур.

В этой схеме криптоанализ начинается с последнего шага шифрования. Ошибки, внесенные до выполнения предпоследнего и предыдущих шагов шифрования, сильно рассеиваются к моменту завершения шифрования, поэтому они не могут быть непосредственно использованы для определения соответствующих этим шагам ключевых параметров. Учитывая это обстоятельство, можно предложить простой способ усиления алгоритма шифрования против рассматриваемого вида нападений, который заключается в применении на последнем шаге дополнительных механизмов преобразования, например, операций подстановки над подблоками большого размера (8, 16 или 32 бит), выполняемых по секретным таблицам (т. е. по таблицам, формируемым на этапе предвычислений в зависимости от секретного ключа).

Рассмотренные выше примеры криптоанализа шифров RC5, DDSS-1 и ГОСТ 28147 показывают, что использование секретных таблиц замены, ключевых операций и операций, зависящих от преобразуемых данных, не может само по себе обеспечить высокую стойкость к атаке на основе генерации случайных аппаратных ошибок. Для достижения высокой стойкости к данному виду атаки необходимо использовать алгоритмы со специальной структурой преобразований. Учитывая особенности осуществления такой атаки, можно предположить, что 512-байтовые алгоритмы, описанные в *гл. 2* и использующие более эффективный механизм выборки подключей в зависимости от преобразуемых данных, обладают высокой стойкостью к атакам на основе случайных аппаратных ошибок. Это связано со следующими особенностями механизма преобразования:

- ❒ индексы выбираемых подключей не задаются непосредственно преобразуемыми подблоками данных;
- ❒ подключи не используются непосредственно при преобразовании подблоков данных, а служат только для модифицирования аккумулирующих переменных;
- ❒ недетерминированные 512-байтовые алгоритмы не позволяют осуществить раздельное вычисление отдельных неизвестных операций, используемых в одном раунде шифрования.

Таким образом, высокая стойкость к атакам на основе аппаратных ошибок достигается применением специальной структуры раундовой функции преобразования, но не благодаря использованию дополнительных раундов шифрования. Одним из способов обеспечения стойкости к атакам на основе генерации аппаратных ошибок служит использование вероятностного шифрования, однако это приводит к увеличению размера зашифрованных данных. Предпочтительнее применять алгоритмы шифрования, которые сами по себе являются стойкими к атакам на основе формирования ошибок.

Заключение

Данная книга не претендует на исчерпывающее изложение содержания современной криптографии, являющейся чрезвычайно многообразным и сложным разделом современной прикладной математики. Авторы преследовали цель в доступной форме изложить введение в общую проблематику и методологию криптографии, многие аспекты которой представлены в свете развиваемой авторами концепции управляемых криптографических преобразований. В книге дается теоретическое обоснование данной концепции и на ее базе излагаются новые методы синтеза скоростных блочных шифров. Полученные результаты, по мнению авторов, составляют содержание одного из перспективных направлений открытых исследований в области криптографии, ведущихся в России.

Использование криптографических преобразований, зависящих от преобразуемых данных, имеет большое теоретическое и практическое значение. Теоретическое значение состоит в обосновании нового класса криптографических примитивов и возможности расширения и совершенствования общих принципов построения итеративных схем блочных алгоритмов. Теоретически важным является также то обстоятельство, что математические свойства сложных на первый взгляд новых криптографических примитивов, связанных с целой системой битовых преобразований, достаточно просто и эффективно определяются аналитически.

Практическое значение концепции управляемых преобразований состоит в возможности создания на ее основе скоростных шифров, максимально адаптированных к аппаратной реализации в виде заказных и программируемых логических интегральных схем (крипточипов). В ряде случаев предлагаемые блочные шифры допускают и скоростную программную реализацию.

Высокая эффективность аппаратной реализации блочных шифров на базе управляемых преобразований и разнообразие задач, при решении которых используются операции битовых перестановок, делают актуальным вопрос о внедрении новой команды управляемой перестановки в систему стандартных инструкций универсального процессора. В криптографическом плане появление такой

команды приведет к скачкообразному повышению производительности современных программных шифров и хэш-функций, удовлетворяющих высоким требованиям криптостойкости.

Изучение концепции управляемых преобразований имеет и важное методическое значение, поскольку способы синтеза и анализа новых криптографических примитивов иллюстрируют методику применения специального математического аппарата исследования криптографических свойств преобразований, связанных с принципами рассеивания и перемешивания битовой информации в блочных шифрах.

Синтезируемые блочные шифры на базе управляемых преобразований могут использоваться в качестве учебных примеров для вычисления дифференциальных и линейных характеристик не только отдельных элементов шифра, но и всей криптосхемы в целом.

Таким образом, данная книга представляется интересной широкому кругу читателей — математикам, ведущим теоретические исследования в области одноключевой криптографии, специалистам-практикам, а также преподавателям высших учебных заведений, студентам и аспирантам.

Авторы осознают, что впервые отражают одно из развивающихся направлений криптографии и поэтому изложение материала книги может оказаться далеко не оптимальным и требующим улучшения. Несомненно, в ближайшей перспективе будут получены и новые результаты по развитию предлагаемого в книге подхода к построению скоростных блочных шифров нового поколения, что приведет к необходимости расширения последующих изданий книги, в которых авторы надеются внести все обнаруженные неточности и опечатки.

Глоссарий

Алгоритм защитного контрольного суммирования — алгоритм вычисления некоторого двоичного вектора сравнительно малого размера, зависящего от каждого бита сообщения произвольной длины. Важнейшим типом алгоритмов защитного контрольного суммирования являются хэш-функции.

Аппаратный шифр — шифр, ориентированный на реализацию в виде электронного устройства.

Аутентификация — процедура установления подлинности пользователя (абонента сети, отправителя сообщения), программы, устройства или данных (информации, получаемого сообщения, ключа). Частным вариантом аутентификации является установление принадлежности сообщения конкретному автору.

Блочный шифр — шифр, входными текстами для которого являются блоки фиксированного размера.

Вероятностное шифрование — процесс шифрования с использованием случайных параметров.

Гибкий шифр — шифр, описываемый как набор криптоалгоритмов, выбираемых в зависимости от секретного ключа.

Гибридная криптосистема — криптосистема, в которой распределение ключей осуществляется с помощью двухключевых криптоалгоритмов, а процесс шифрования информации — с помощью одноключевых. Гибридные криптосистемы сочетают в себе удобство распределения секретных ключей и высокую скорость шифрования.

Двухключевая криптография — направление исследований в области криптографии и разработки криптосистем, основанных на использовании двух ключей — открытого и закрытого. Открытый (публичный, общедоступный) ключ предполагается известным всем пользователям криптосистемы и потенциальному противнику. Закрытый ключ предполагается известным только одному

пользователю. Существенным является то, что для реализации различных протоколов в двухключевых криптосистемах не требуется, чтобы секретный ключ был известен более чем одному пользователю. Это позволяет реализовать криптографические протоколы, предусматривающие взаимодействие сторон в условиях недоверия друг к другу. Такая особенность двухключевых криптосистем лежит в основе систем электронной цифровой подписи.

Зашифрование — процесс преобразования информации с целью защиты от несанкционированного доступа. Зашифрование преобразует исходный открытый текст в шифртекст (или криптограмму). Шифртекст называют иногда шифрованным текстом. Зашифрование есть процесс защитного преобразования данных, обеспечивающий защиту информации путем сокрытия ее смысла. Стойкость зашифрования (или уровень защищенности) определяется тем, что при зашифровании используется один или несколько секретных параметров, которые предполагаются известными только законным (легальным) пользователям.

Имитозащита — способ защиты от навязывания ложных сообщений путем передачи вместе с сообщением небольшой дополнительной информации, называемой имитовставкой.

Имитовставка — криптографическая контрольная сумма, зависящая от каждого бита сообщения и вычисляемая с использованием криптографических алгоритмов (например, алгоритмов шифрования) и ключа.

Ключ — секретный параметр, управляющий ходом зашифрования. Синонимами являются секретный ключ, ключ шифрования. Ключ определяет выбор конкретного варианта преобразования для зашифрования и расшифрования из множества преобразований, составляющих шифр.

Криптоанализ — процесс (алгоритм) получения исходного текста по шифрованному без знания ключа или процесс вычисления ключа по исходному и шифрованному тексту. Криптоанализ выполняется противником с целью получения возможности осуществления несанкционированного доступа или разработчиком с целью получения оценки стойкости шифра.

Криптографическое преобразование — процедура специального преобразования информации для решения одной из следующих криптографических задач: шифрования данных, формирования цифровой электронной подписи, вычисления специальных криптографических контрольных сумм и имитовставки.

Криптографический примитив — в широком смысле это операция или процедура, используемая в качестве элемента шифра, в узком смысле это операции и процедуры, которые определяют требуемые свойства криптосистемы (стойкость, возможность зашифрования и расшифрования с использованием различных ключей и т. п.).

Криптографический протокол — протокол, предусматривающий взаимодействие двух и более сторон с использованием криптографических алгоритмов.

Криптосистема с закрытым ключом — традиционная криптосистема, использующая секретный ключ, известный более чем одному пользователю.

Криптостойкость (стойкость шифра) — способность криптосистемы выдерживать различные нападения. Количественно выражается числом операций некоторого типа, которые необходимо выполнить для решения задачи криптоанализа. При этом указывается тип атаки на криптосистему, т. е. исходные данные для криптоанализа. Если исходные данные не указываются, то подразумевается стойкость к лучшему известному алгоритму криптоанализа (для атаки на основе специально подобранных текстов).

Лавинный эффект — свойство распространения влияния одного входного бита на многие выходные биты (свойство размножения ошибок).

Лобовое нападение (силовая атака) — криптоанализ путем исчерпывающего перебора всех возможных ключей, т. е. методом подбора ключа. Для криптосистем с конечным ключом этот метод является универсальным, т. е. он применим к любому шифру такого типа. Однако вероятность данного нападения мала. Для задания низкой вероятности успеха такой атаки для симметричных криптосистем требуется использовать ключи размером не менее 80 бит. В настоящее время считается, что гарантированную стойкость к лобовому нападению обеспечивают равновероятно генерируемые случайные ключи размером 128 бит.

Поточный шифр — шифр, преобразующий последовательно отдельные биты или знаки исходного текста и ориентированный на реализацию в виде программы.

Противник — субъект (специальная программа, оснащенная группа специалистов или физическое лицо), пытающийся преобразовать шифртекст в открытый текст без знания ключа или вычислить ключ по известным исходным и шифрованным текстам. Синонимами являются термины атакующий, нападающий, криптоаналитик, криптоаналитик противника.

Программный шифр — шифр, ориентированный на реализацию в виде программы.

Протокол рукопожатия — протокол, позволяющий двум удаленным сторонам, владеющим некоторым общим секретом, осуществить взаимную проверку подлинности без раскрытия секрета.

Расшифрование — процесс восстановления открытого текста по шифрованному тексту и известному ключу. (Отметим, что вместо термина "расшифрование" иногда используется термин "дешифрование". В российской специальной литературе под термином "дешифрование" обычно понимают процесс восстановления открытого текста без знания ключа или вычисление ключа по открытым текстам и шифртекстам.)

Раскрытие шифра (криптосистемы, криптоалгоритма) — нахождение способа решения задачи криптоанализа за разумное время при использовании современных вычислительных средств. В качестве синонима используется термин "взлом шифра".

Симметричная криптосистема — криптосистема с закрытым ключом.

Система тайного электронного голосования — криптографический протокол, обеспечивающий тайну голосования для каждого голосующего при возможности проверки со стороны последнего, как учтен его голос ("за" или "против"). При этом никто, включая избирательный комитет, не сможет установить, как проголосовал избиратель.

Слепая подпись — протокол формирования цифровой электронной подписи, позволяющий сформировать правильную ЭЦП, соответствующую некоторому цифровому сообщению, которое подписывающая сторона получает в зашифрованном (т. е. недоступном для нее) виде.

Стеганография — способ скрытной передачи информации или способ формирования скрытного канала. Часто используется для передачи шифртекстов. Широкое применение компьютерных технологий дало новый сильный импульс в развитии стеганографических методов. В настоящее время наблюдается тенденция использования криптографических идей в современной стеганографии, например, секретных ключей. Актуальным направлением стеганографии является создание цифровых водяных знаков.

Управление ключами — совокупность мероприятий и процедур, обеспечивающих защищенную генерацию, распределение, хранение и уничтожение ключей.

Управляемые операции — операции, описываемые как множество некоторых более простых операций, выбираемых в зависимости от значения некоторого управляющего кода.

Хэширование — процесс вычисления значения хэш-функции.

Хэш-функция — криптографическая функция (процедура, алгоритм), аргументом которой являются произвольные сообщения (тексты, документы), представленные в виде последовательности битов, и значения которой лежат в области от 0 до $2^m - 1$, где m — размер хэш-кода (выходного значения хэш-функции). Хэш-функции представляют собой специальный класс криптографических контрольных сумм, вычисляемых без использования секретных параметров и обеспечивающих сложную зависимость выходного значения от каждого бита входного сообщения.

Цифровая подпись — электронная цифровая подпись.

Шифр с открытым ключом (двухключевой шифр) — двухключевая криптосистема. Синонимами являются термины "шифр (криптосистема) с публичным ключом", "асимметричная криптосистема".

Шифр (криптосистема) — совокупность алгоритмов, используемых при зашифровании и расшифровании.

Шифрование — процесс преобразования информации с использованием некоторой дополнительной информации, управляющей этим процессом и называемой ключом. Реализуется в виде процедуры расшифрования или зашифрования. Часто под шифрованием понимается зашифрование.

Шифратор — электронное устройство или программа, реализующая алгоритмы шифрования.

Электронная цифровая подпись — некоторая дополнительная информация, соответствующая данному электронному документу (сообщению), которая могла быть сформирована только владельцем некоторого секрета — закрытого ключа и которая позволяет с использованием специального алгоритма установить факт соответствия подписи закрытому ключу подписывающего. Под электронной цифровой подписью (ЭЦП) понимается также криптографическая система (совокупность алгоритмов и правил), позволяющая подписывать цифровые сообщения и проверять правильность формируемых цифровых подписей.

Список литературы

1. Алексеев Л. Е., Белкин Т. Г., Молдовян А. А., Молдовян Н. А. Способ итеративного шифрования блоков данных. Патент РФ № 2140714. МПК 6 H 04 L 9/20. — Бюл. № 30 от 27.10.99.
2. Алексеев Л. Е., Молдовян А. А., Молдовян Н. А. Алгоритмы защиты информации с СЗИ НСД "СПЕКТР-Z" // Вопросы защиты информации. — 2000. — № 3. — С. 63—68.
3. Белкин Т. Г., Гуц Н. Д., Молдовян А. А., Молдовян Н. А. Способ скоростного шифрования на базе управляемых операций. Управляющие системы и машины. — 1999. — № 6. — С. 78—87.
4. Биленко Ю. А., Жуков А. Е. Криптографические свойства преобразований, легко реализуемых на стандартных PC-процессорах // Безопасность информационных технологий. — 2000. — № 2. — С. 40—49.
5. Бодров А. В., Молдовян Н. А., Молдовяну П. А. Оптимизация механизма управления перестановки в скоростных шифрах // Вопросы защиты информации. — 2002. — № 1. — С. 44—50.
6. Боровков А. А. Курс теории вероятностей. — М.: Наука, 1972. — 287 с.
7. Варфоломеев А. А., Жуков А. Е., Мельников А. Б., Устюжанин Д. Д. Блочные криптосистемы. Основные свойства и методы анализа стойкости // Учебное пособие. — М.: МИФИ, 1998. — 200 с.
8. Винокуров А. Ю., Применко Э. А. Анализ тенденции подходов к синтезу симметричных блочных шифров // Безопасность информационных технологий. — 2001. — № 2. — С. 5—14.
9. Герасименко В. А. Защита информации в автоматизированных системах обработки данных. В 2-х кн. — М.: Энергоатомиздат, 1994. — 576 с.
10. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. — М. Госстандарт СССР, 1989.

11. Гуц Н. Д., Еремеев М. А., Молдовян А. А. Алгоритм формирования расширенного ключа на основе блоков управляемых перестановок // Вопросы защиты информации. — 2001. — № 3. — С. 41—46.

12. Гуц Н. Д., Изотов Б. В., Молдовян А. А., Молдовян Н. А. Итеративный способ блочного шифрования. Патент РФ № 2172075. МПК 7 H 04 L 9/00. — Бюл. № 22 от 10.08.2001.

13. Гуц Н. Д., Изотов Б. В., Молдовян А. А., Молдовян Н. А. Проектирование двухместных управляемых операций для скоростных гибких криптосистем // Безопасность информационных технологий. — 2001. — № 2. — С. 14—23.

14. Гуц Н. Д., Изотов Б. В., Молдовян Н. А. Скоростной алгоритм шифрования SPECTR-H64 // Безопасность информационных технологий. — 2000. — № 4. — С. 37—50.

15. Гуц Н. Д., Изотов Б. В., Молдовян Н. А. Управляемые перестановки с симметричной структурой в блочных шифрах // Вопросы защиты информации. — 2000. — № 4. — С. 57—65.

16. Гуц Н. Д., Молдовян А. А., Молдовян Н. А. Гибкие аппаратно-ориентированные шифры на базе управляемых сумматоров // Вопросы защиты информации. — 2000. — № 1. — С. 8—15.

17. Гуц Н. Д., Молдовян А. А., Молдовян Н. А. Обоснование полноцикловых перестановок битов переноса в управляемых сумматорах гибких шифров // Вопросы защиты информации. — 2000. — № 2. — С. 23—28.

18. Гуц Н. Д., Молдовян А. А., Молдовян Н. А. Построение управляемых блоков перестановок с заданными свойствами // Вопросы защиты информации. — 1999. — № 4. — С. 39—49.

19. Иванов М. А. Криптографические методы защиты информации в компьютерных системах и сетях. — М.: КУДИЦ-ОБРАЗ, 2001. — 368 с.

20. Изотов Б. В., Молдовян А. А., Молдовян Н. А. Скоростные методы защиты информации в АСУ на базе управляемых операций // Автоматика и телемеханика. — 2001. — № 6. — С. 168—184.

21. Карякин Ю. Д. Технология "AXIS-2000" защиты материальных объектов от подделки // Управление защитой информации. — Минск. — 1997. — Т. 1. — № 2. — С. 90—98.

22. Кострикин А. И. Введение в алгебру. Основы алгебры: Учебник для вузов. — М.: Физматлит, 1994. — 320 с.

23. Крамер Г. Математические методы статистики. — М.: Мир, 1975. — 648 с.

24. Молдовян А. А., Молдовян Н. А. Вероятностные механизмы в недетерминированных блочных шифрах // Безопасность информационных технологий. — 1997. — № 3. — С. 58—61.

25. Молдовян А. А., Молдовян Н. А. Метод скоростного преобразования для защиты информации в АСУ // Автоматика и телемеханика. — 2000. — № 4. — С. 151—165.

26. Молдовян А. А., Молдовян Н. А., Молдовяну П. А. Новый метод криптографических преобразований для современных систем защиты ПЭВМ // Управляющие системы и машины. — Киев. — 1992. — № 9/10. — С. 44—50.

27. Молдовян А. А., Молдовян Н. А. Псевдовероятностные скоростные блочные шифры для программной реализации. Кибернетика и системный анализ. — Киев.— 1997. — № 4. — С. 133—141.

28. Молдовян А. А., Молдовян Н. А. Скоростные шифры на базе нового криптографического примитива // Безопасность информационных технологий. — 1999. — № 1. — С. 82—88.

29. Молдовян А. А., Молдовян Н. А., Советов Б. Я. Криптография. — СПб.: Лань, 2000. — 224 с., ил.

30. Молдовян А. А., Молдовян Н. А. Способ блочного шифрования данных. Патент РФ № 2106752. МПК 6 H 04 L 9/00. — Бюл. № 7 от 10.03.98.

31. Молдовян А. А. Некоторые вопросы защиты программной среды ПЭВМ // Безопасность информационных технологий. — 1995. — № 2. — С. 22—28.

32. Молдовян Н. А. Скоростные блочные шифры. — СПб.: Издательство СПбГУ, 1998. — 230 с.

33. Смид М. Э., Бранстед Д. К. Стандарт шифрования данных: Прошлое и Будущее. The Data Encryption Standard: Past and Future // ТИИЭР. — 1988. — Т. 76. — № 5. — С. 43—53.

34. Сэвидж Д. Э. Сложность вычислений. — М.: Факториал, 1998. — 368 с., ил.

35. Шеннон К. Э. Символический анализ релейных и переключательных схем // В кн.: Шеннон К. Э. Работы по теории информации и кибернетике. — М.: Иностранная литература, 1963. — С. 9—45.

36. Шеннон К. Э. Синтез двухполюсных переключательных схем // В кн.: Шеннон К. Э. Работы по теории информации и кибернетике. — М.: Иностранная литература, 1963. — С. 59—105.

37. Шеннон К. Э. Теория связи в секретных системах // В кн.: Шеннон К. Э. Работы по теории информации и кибернетике. — М.: Иностранная литература, 1963. — С. 333—402.

38. Benes V. E. Algebraic and Topological Properties of Connecting Networks // Bell Systems Technical Journal. — 1962. — V. 41. — P. 1249—1274.

39. Benes V. E. Mathematical Theory of Connecting Networks and Telephone Traffic. — New York.: Academic Press, 1965. — 319 p.

40. Benes V. E. On Rearrangeable Three-stage Connecting Networks // Bell Systems Technical Journal. — 1962. — V. 41. — P. 1481—1492.

41. Bennett C. H., Brassard G. Quantum Cryptography: Public Key Distribution and Coin Tossing // Proceedings of the IEEE International Conference on Computers, Systems, and Signal Processing // Banjalore, India. — 1984. — P. 175—179.

42. Biham E., Biryukov A., Shamir A. Cryptanalysis Skipjack Reduced to 31 Round Using Impossible Differentials // Advances in Cryptology — EUROCRYPT'99 // Lecture Notes in Computer Science. Springer-Verlag. — 1999. — V. 1592. — P. 12—23.

43. Biham E. On Matsui's Linear Cryptanalysis // Advances in Cryptology — EUROCRYPT'94 // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 950. — P. 341—355.

44. Biham E. and Shamir A., Differential Cryptanalysis of the Full 16-round DES // Advances in Cryptology — CRYPTO'92 // Lecture Notes in Computer Science. Springer-Verlag. — 1993. — V. 740. — P. 487—496.

45. Biham E., Shamir A. Differential cryptanalysis of DES-like cryptosystems (extended abstract) // Advances in Cryptology — CRYPTO'90 // Lecture Notes in Computer Science. Springer-Verlag. — 1991. — V. 537. — P. 2—21.

46. Biham E., Shamir A. Differential Fault Analysis of Secret Key Cryptosystems // Advances in Cryptology — CRYPTO'97 // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1294. — P. 513—525.

47. Biryukov A., Wagner D. Advanced Slide Attacks // Advances in Cryptology — EUROCRYPT'2000 // Lecture Notes in Computer Science. Springer-Verlag. — 2000. — V. 1807. — P. 589—606.

48. Biryukov A., Wagner D. Slide Attacks // Fast Software Encryption, 6th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1999. — V. 1636. — P. 245—259.

49. Burwick C., Coppersmith D., D'Avingnon E., Gennaro R., Halevi Sh., Jutla Ch., Matyas Jr.S.M., O'Connor L., Peyravian M., Safford D., Zunic N. MARS — a Candidate Cipher for AES // Proc. of 1st Advanced Encryption Standard Candidate Conference, Venture, California, Aug. 20-22, 1998 (http://www.nist.gov/aes).

50. Camion P., Carlet C., Charpin P., Sendrier. On Correlation-immune Functions // Advances in Cryptology — CRYPTO'91 // Lecture Notes in Computer Science. Springer-Verlag. — 1992. — V. 576. — P. 87—100.

51. Carlet C. Two New Classes of Bent Functions // Advances in Cryptology — EUROCRYPT'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 765. — P. 77—101.

52. Chabaud F., Vaudenay S. Links Between Differential and Linear Cryptanalysis // Advances in Cryptology — EUROCRYPT'94 // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 950. — P. 356—365.

53. Charnes C., O'Connor L., Pieprzyk J., Safavi-Naini R., Zeng Y. Comments on Soviet Encryption Algorithm // Advances in Cryptology — EUROCRYPT'94 //

Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 950. — P. 433—438.

54. Chaum D. Blind Signatures for Untraceable Payments // Advances in Cryptology — Crypto'82 // Plenum Press. — 1983. — P. 199—203.
55. Chaum D. Blind Signature Systems // U.S. Patent # 4,759,063. — 19 Jul 1988.
56. Chaum D. Security Without Identification: Transaction Systems to Make Big Brother Obsolete // Communications of the ACM. — 1985. — V. 28. — N. 10. — P. 1030—1044.
57. Chor B., Goldreich O., Hastard J., Friedman J., Rudich S., Smolensky R. The Bit Extraction Problem or t-resilient Functions // IEEE Symposium on Foundations of Computer Science. — 1985. — V. 26. — P. 396—407.
58. Clos C. A Study of Nonblocking Switching Networks // Bell System Technical Journal. — 1953. — V. 32. — P. 406—424.
59. Daemen J. Cipher and Hash Function Design, // Ph.D. Theses, Katholieke Universiteit Leuven, Mar 1995 (http://csrc.nist.gov/encryption/aes).
60. Daemen. J., Govaerts R. Vandevalle J. Correlation Matrices // Fast Software Encryption, Second International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 275—285.
61. Daemen. J., Rijmen V. AES proposal: Rijndael. In First AES conference, August 1998 (http://csrc.nist.gov/encryption/aes/aes).
62. Diffie W., Hellman M.E. New Directions in Cryptography // IEEE Transactions on Information Theory. — 1976. — V. IT-22. N.6. — P. 644—654.
63. Dillon J. Elementary Hadamard Difference Sets // Proceedings of the 6th Southeastern Conference Combinatorics Graph Theory, and Computing. — 1975. — P. 237—249.
64. Dobbertin H. Construction of Bent Functions and Balanced Boolean Functions with High Nonlinearity // Fast Software Encryption, Second International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 61—74.
65. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms // IEEE Transactions on Information Theory. —1985. — V. IT-31. N. 4. — P. 469—472.
66. Feistel H. Cryptography and Computer privacy // Scientific American. — 1973. — V. 228. — N. 5. — P. 15—23.
67. Forre R. The Strict Avalanche Criterion: Spectral Properties of Boolean Functions and an Extended Definition // Advances in Cryptology — CRYPTO'88 // Lecture Notes in Computer Science. Springer-Verlag. — 1989. — V. 403. — P. 450—468.
68. Goots N. D., Moldovyan A. A., Moldovyan N. A. Fast Encryption Algorithm SPECTR-H64 // Mathematical Methods, Models and Architectures for Computer

Networks Security // Lecture Notes in Computer Science. Springer-Verlag. — 2001. — V. 2052. — P. 275—286.

69. Goots N. D., Moldovyan A. A., Moldovyan N. A. Two fast flexible software-oriented ciphers // В кн. "Комплексная защита информации". Вып. 3. Институт технической кибернетики НАН Беларуси. — Минск, 2000. — С. 95—101.

70. Harpers C., Kramer G. G., Massey J. L. A generalization of linear cryptanalysis and the applicability of Matsui's piling-up lemma // Advances in Cryptology — EUROCRYPT'95 // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 921. — P. 24—38.

71. Hong S., Lee S., Lim J., Sung J., Cheon D., Cho I. Provable Security against Differential and Linear Cryptanalysis for the SPN Structure // Fast Software Encryption, 7th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 2000. — V. 1978. — P. 273—283.

72. Izotov B. V., Moldovyan A. A., Moldovyan N. A. Controlled Operations as a Cryptographic Primitive // Methods, Models and Architectures for Network Security // Lecture Notes in Computer Science. Springer-Verlag. — 2001. — V. 2052. — P. 230—241.

73. Jakobsen T., Knudsen L. R. The Interpolation Attack on Block Ciphers // Fast Software Encryption, 4th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1267. — P. 28—40.

74. Kaliski B. S.(Jr), Yin Y. L. On Differential and Linear Cryptanalysis of the RC5 Encryption Algorithm // Advances in Cryptology — CRYPTO'95 // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 963. — P. 171—184.

75. Kam J. B., Davida G. I. Structured Design of Substitution-Permutation Encryption Networks // IEEE Transactions on Computers. — 1979. — V. 28. — N. 10. — P. 747—753.

76. Kelsey J., Schneier B., Wagner D., Related-Key Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES // Advances in Cryptology — CRYPTO'96 // Lecture Notes in Computer Science. Springer-Verlag. — 1996. — V. 1109. — P. 237—251.

77. Knudsen L. R., Berson T. A. Truncated Differentials of SAFER // Fast Software Encryption, Third International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1996. — V. 1039. — P. 15—26.

78. Knudsen L. R. Truncated and higher order differentials // Fast Software Encryption, Second International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 196—211.

79. Kocher P., Jaffe J., Jun B. Introduction to Differential Power Analysis and Related Attacks (http://www.cryptography.com/dpa/technical/index.html).

80. Kruskal C. P., Snir M., A Unified Theory of Interconnection Network Structure // Theoretical Computer Science. — 1986. — V. 48. — P. 75—94.

81. Kurosawa K., Satoh T. Design of SAC/PC(l) of Order k Boolean Functions and Three Other Cryptographic Criteria // Advances in Cryptology — EUROCRYPT'97 // Lecture Notes in Computer Science. Springer-Verlag. — 1998. — V. 1233. — P. 434—449.

82. Kwan M. The Design of the ICE Encryption Algorithm // Fast Software Encryption, 4th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1267. — P. 69—82.

83. Lai X., Massey J. L. Hash Functions Based on Block Ciphers // Advances in Cryptology — EUROCRYPT'92 // Lecture Notes in Computer Science. Springer-Verlag. — 1993. — V. 658. — P. 55—70.

84. Lai X., Massey J., Murphy S. Markov Ciphers and Differential Cryptanalysis // Advances in Cryptology — EUROCRYPT'91 // Lecture Notes in Computer Science. Springer-Verlag. — 1992. — V. 547. — P. 17—38.

85. Lloyd S. Counting Functions Satisfying a Higher Order Strict Avalanche criterion // Advances in Cryptology — EUROCRYPT'89 // Lecture Notes in Computer Science. Springer-Verlag. — 1990. — V. 434. — P. 64—74.

86. Matsui M. Linear Cryptanalysis Method for DES Cipher // Advances in Cryptology — EUROCRYPT'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 765. — P. 386—397.

87. Matsui M. New structure of Block Ciphers with Provable Security Against Differential and Linear Cryptanalysis // Advances in Cryptology — EUROCRYPT'96 // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1267. — P. 205—218.

88. McFarland R. A Family of Difference Sets in Non-Cyclic Group // Journal of Combinatorial Theory. — 1973. — V. A-15. — P. 1—10.

89. Menezes A. J., Van Oorscchot P., Vanstone Scott. A. Handbook of Applied Cryptography. — CRC Press. — 1996. — 780 p.

90. Moldovyan A. A., Moldovyan N. A., Andronati N. R., Rogozhin Yu. V. Software-Oriented Approach to Computer Processing Protection // Proceedings of the European Simulation Meeting: "Simulation Tools and Applications" // Gyor, Hungary, 28-30 Aug 1995. — P. 143—149.

91. Moldovyan A. A., Moldovyan N. A. Fast Software Encryption System Based on Local Pseudorandomness // Computer Science Journal of Moldova. — 1995. — V. 3. — N. 3(9). — P. 252—262.

92. Moldovyan A. A., Moldovyan N. A. Fast Software Encryption Systems for Secure and Private Communication // Proceedings of the 12th International Conference on Computer Communication // Seoul, Korea. — 21-24 August 1995. — V. 1. — P. 415—420.

93. Moldovyan A. A., Moldovyan N. A. Flexible Block Ciphers with Provably Inequivalent Cryptalgorithm Modifications // Cryptologia. — 1998. — V. XXII. — N. 2. — P. 134—140.

94. Moldovyan A. A., Moldovyan N. A., Moldovyan P. A. Effective software-oriented cryptosystem in complex PC security software // Computer Science Journal of Moldova. — 1994. — V. 2. — N. 3. — P. 269—282.

95. Moldovyan A. A., Moldovyan N. A. Provably Indeterminate 128-bit Cipher // Computer Science Journal of Moldova. — 1997. — V. 5. — N. 2. — P. 185—197.

96. Moldovyan A. A., Moldovyan N. A. Software Encryption Algorithms for Transparent protection Technology // Cryptologia. — 1998. — V. XXII. — N. 1. — P. 56—68.

97. Moldovyan A. A., Moldovyan N. A., Sovetov B. Ya. Software-Oriented Ciphers for Computer Communication Protection // Proceedings of International Conference Applications of Computer Systems ACS'97. November 13-14, 1997. — Szczecin, Poland. — P. 443—450.

98. Moldovyan A. A., Moldovyan N. A., Zaikin O. A. Undetermined Software-Oriented Ciphers Based on Data-Dependent Subkey Selection // Proceedings of International Conference of Computer's Methods in Control Systems CMCS'97. December 11-12, 1997. — Szczecin, Poland. — P. 157—164.

99. Naor M., Reingold O. Constructing Pseudo-Random Permutations with a Prescribed Structure // (http://www.Iacr.org/ePrint/2000 Pentium/C/N101/Permutation Networks).

100. Nyberg K. Differentially uniform mappings for cryptography // Advances in Cryptology — EUROCRYPT'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 765. — P. 55—64.

101. Nyberg K. Linear Approximation of Block Ciphers // Advances in Cryptology — EUROCRYPT'94 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 950. — P. 439—444.

102. Nyberg K., Knudsen L. Provable Security Against Differential Cryptanalysis // Advances in Cryptology — CRYPTO'92 // Lecture Notes in Computer Science. Springer-Verlag. — 1993. — V. 740. — P. 566—574.

103. Nyberg K. On the Construction of Highly Nonlinear Permutations // Advances in Cryptology — EUROCRYPT'92 // Lecture Notes in Computer Science. Springer-Verlag. — 1993. — V. 658. — P. 92—98.

104. Nyberg K. New Bent Mappings Suitable for Fast Implementation // Fast Software Encryption, Second International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 179—184.

105. Nyberg K. Perfect Nonlinear S-Boxes // Advances in Cryptology — EUROCRYPT'91 // Lecture Notes in Computer Science. Springer-Verlag. — 1991. — V. 547. — P. 378—385.

106. Nyberg K. S-Boxes and Round Functions with Controllable Linearity and Differential Uniformity // Fast Software Encryption, Second International Workshop //

Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 111—130.

107. O'Connor L. On the Distribution of Characteristics in Composite Permutations // Advances in Cryptology — CRYPTO'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 773. — P. 403—412.

108. Parker S. Notes on Shuffle/Exchange-Type Switching Networks // IEEE Transactions on Computers. — 1980. — V. C-29. — N. 5. — P. 213—222.

109. Portz M. A Generalized Description of DES-based and Benes-based Permutation generators // Advances in Cryptology — AUSCRYPT'92 // Lecture Notes in Computer Science. Springer-Verlag. — 1992. — V. 718. — P. 397—409.

110. Portz M. On the use of interconnection networks in cryptography // Advances in Cryptology — EUROCRYPT'91 // Lecture Notes in Computer Science. Springer-Verlag. — 1991. — V. 547. — P. 302—315.

111. Preneel B., Bosselaers A., Rijmen V., Van Rompay B., Granboulan L., Stern J., Murphy S., Dichtl M., Serf P., Biham E., Dunkelman O., Furman V., Koeune F., Piret G., Quisquater J-J., Knudsen L., Raddum H // Comments by the NESSIE Project on the AES Finalists // 24 may 2000 (http://www.nist.gav/aes).

112. Preneel B., Van Leekwijck W., Van Linden L., Govaerts R., Vandewalle J. Propagation Characteristics of Boolean Functions // Advances in Cryptology — EUROCRYPT'90 // Lecture Notes in Computer Science. Springer-Verlag. — 1991. — V. 473. — P. 161—173.

113. Rijmen V., Preneel B. A Family of Trapdoor Ciphers // Fast Software Encryption, 4th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1267. — P. 139—150.

114. Rivest R. L., Robshaw M. J. B., Sidney R. and Yin Y. L. The RC6™ Block Cipher // Proceedings of the 1st Advanced Encryption Standard // Candidate Conference, Venture, California, Aug. 20-22, 1998 (http://www.nist.gov/aes).

115. Rivest R. L. The RC5 Encryption Algorithm // Fast Software Encryption, Second International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 1008. — P. 86—96.

116. Rivest R., Shamir A, Adleman L. A method for Obtaining Digital Signatures and Public-Key Cryptosystems // Communication of the ACM. — 1978. — V. 21. — N. 2. — P. 120—126.

117. Rothaus O. On Bent Functions // Journal of Combinatorial Theory. — 1976. — V. A-20. — P. 300—305.

118. Sakurai K., Faruya S. Improving Linear Cryptanalysis of LOKI91 by Probabilistic Counting Method // Fast Software Encryption, 4th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1997. — V. 1267. — P. 114—133.

119. Sarkar P., Maitra S. New Directions in Design of Resilient Boolean Functions // Indian Statistical Institute. Technical Report No. ASD/2000/04. 8 Feb. 2000 (http://www.elibrary.ru).

120. Schneier B., Applied Cryptography: Protocols, Algorithms, and Source Code (Second Edition). — New York.: John Wiley & Sons. — 1996. — 758 p.

121. Schneier B. Description of a New Variable-Length Key, 64-Bit Block Cipher (Blowfish) // Fast Software Encryption, First International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 809. — P. 191—204.

122. Seberry J., Zhang X-M., Zheng Y. Nonlinearly Balanced Boolean Functions and Their Propagation Characteristics // Advances in Cryptology — CRYPTO'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 773. — P. 49—60.

123. Seberry J., Zhang X-M., Zheng Y. On Constructions and Nonlinearity of Correlation Immune Functions // Advances in Cryptology — EUROCRYPT'93 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 765. — P. 181—199.

124. Seberry J., Zhang X-M., Zheng Y. Pitfalls in Designing Substitution Boxes // Advances in Cryptology — CRYPTO'94 // Lecture Notes in Computer Science. Springer-Verlag. — 1994. — V. 839. — P. 383—396.

125. Seberry J., Zhang X-M., Zheng Y. Relations Among Nonlinearity Criteria // Advances in Cryptology — EUROCRYPT'94 // Lecture Notes in Computer Science. Springer-Verlag. — 1995. — V. 950. — P. 376—388.

126. Shannon C. E. Communication Theory of Secrecy Systems // Bell Systems Technical Journal. — 1949. — V. 28. — P.656—715.

127. Siegenthaler T. Correlation-Immunity of Nonlinear Combining Functions for Cryptographic Applications // IEEE Transactions on Information Theory. — 1984 — V. IT-30. — 776—780.

128. Staffelbach O., Meier W. Nonlinearity Criteria for Cryptographic Functions // Advances in Cryptology — EUROCRYPT'89 // Lecture Notes in Computer Science. Springer-Verlag. — 1990. — V. 434. — P. 549—562.

129. Stinson D.R. Cryptography. Theory and Practice. — New York. CRC Press LLC. — 1995. — 434 p.

130. Stone H. S. Parallel processing with the perfect shuffle // IEEE Transactions on Computers. — 1971. — V. C-20. — P. 153—161.

131. Tarannikov Y. On Resilient Boolean Functions with Maximal Possible Nonlinearity. Department Moscow State University (http://www.elibrary.ru).

132. Wagner D. The Boomerang Attack // Fast Software Encryption, 6th International Workshop // Lecture Notes in Computer Science. Springer-Verlag. — 1999. — V. 1636. — P. 156—170.

133. Waksman A. A Permutation Network // Journal of the ACM. — 1968. — V. 15. — N. 1. — P. 159—163.

134. Webster A. F., Tavares S. E. On the Design of S-boxes // Advances in Cryptology — CRYPTO'85 // Lecture Notes in Computer Science. Springer-Verlag. — 1986. — V. 219. — P. 523—534.

135. Xiao G., Massey J. A Spectral Characterization of Correlation-immune Functions // IEEE Transactions on Information Theory. — 1988. — V. 34. — N. 3. — P. 569—571.

136. Zhang X-M., Zheng Y., Imai H. Relating Differential Distribution Tables to Other Properties of Substitution Boxes // Designs, Codes and Cryptography. — Boston, 2000. — V. 19. — N. 1. — P. 45—63.

137. Zeng Y., Mihaljevic M., Imai H. A Cellular Automaton Based Fast One-way Hash Function Suitable for Hardware Implementation // International Workshop on Practice and Theory in Public Key Cryptography (PKC'98). — Feb. 1998. Japan. Yokohama. Pre-Proceedings. — P. 187—200.

Предметный указатель

B

Bent function 252

C

Correlation Immunity (CI) 250

P

Propagation Criterion (PC) 247

R

Resilient function 250

S

Strict Avalanche Criterion (SAC) 247

А

Автокорреляция булевой функции 246
Алгебраическая нормальная форма (АНФ) 240
Алгебраическая степень булевой функции 240
Алгоритм:
◊ инициализации 24
◊ предвычислений 24
◊ преобразования данных, блочный 333
Аутентификация:
◊ информации 33
◊ пользователей 30

Б

Бент-функция 252
◊ векторная 252
Биекция 238
Бит переноса 276
Блок:
◊ h-го порядка 178
◊ максимального порядка 179
◊ равновероятного смещения 177
◊ управляемых перестановок 167
▫ обратный 169
▫ полноцикловых 201
◊ управляемых подстановок 163, 164

Блочный шифр 231, 321
◊ RC5 234
◊ RC6 234
◊ SPECTR-H64 328
◊ ГОСТ 28147-89 314
◊ марковский 298
Большая интегральная схема 320
Булева функция 158, 239
◊ аффинная 241
◊ векторная 158, 239
◊ линейная 241
◊ образующая 158, 235, 239, 262, 267
◊ сбалансированная 158, 239
◊ совершенно нелинейная 252
◊ эластичная 250

В

Вентиль 170
Вес Хэмминга 158
Вложение 238

Г

Гипотеза о стохастической эквивалентности 296
Глубина булевой функции 320
Глубина схемы 171

Д

Декомпозиция блока 176
Дискретное преобразование Фурье 241

З

Заключительное преобразование 349
Защита информации 18
Злоумышленник 17

И

Имитовставка 30
Имитозащита 30
Инволюция 204, 241
◊ перестановочная 204
Инъекция 238

К

Канонический вид переходной функции 270
Каскадная структура 182
Керкхофф 25
◊ правило 26
◊ принцип 113
Клеточный автомат 236
Ключ:
◊ особый 315
◊ открытый 34
◊ рабочий 315
◊ расширенный 28, 319, 323
◊ раундовый 322
◊ секретный 23, 28, 34, 315, 316, 322
◊ слабый 315
Коллизия 258, 316
Кольцо 157
Комбинационная сложность булевой функции 320
Комбинированный управляемый сумматор 280
Коммутатор фиксированный 165
Композиция отображений 160
Конкатенация 158
Контроль целостности информации 32
Концепция управляемых операций 332
Корреляционная иммунность 250
Корреляционная матрица 306
Корреляция булевых функций 243
Криптоанализ 18, 27
◊ дифференциальный 291, 294
▫ высокого порядка 291
▫ на основе невозможных разностей 292
▫ усеченный 291
◊ интерполяционный 291
◊ линейный 291, 304
◊ на основе бумеранговой атаки 292
◊ на основе генерации аппаратных ошибок 291, 292, 311, 316
◊ на основе замера мощности излучения 291
◊ на основе относительных ключей 292, 314
◊ на основе скользящей атаки 292
◊ частотный 49
Криптограмма 23
Криптографическая защита ценных бумаг 36
Криптографический примитив 231

Криптология 18
Криптосхема Фейстеля 52, 260, 294, 343
◊ усовершенствованная 325
Критерий:
◊ полноты 344
◊ распространения изменений (PC) 247
◊ строгого лавинного эффекта (SAC) 247

Л

Лавинный вектор 249
Лавинный эффект 249
Линейная аппроксимация 304
Линейная размерность 254
Линейная структура 253
Линейная характеристика 244, 305
Линейное векторное пространство 157
Линейность булевой функции 245
Линейный криптоанализ 206
Логические операции 170

М

Маска объединенная 207
Матрица Адамара 242
Матрица разностей 298
Матрица Уолша-Адамара 242
Матричная структура 173
Метод полного (тотального) перебора 334
Модель комбинаторно-вероятностная 116
Модель управляемой подстановочной операции:
◊ базовая рекурсивная 268
◊ биективная 264
◊ биективная рекурсивная 268
◊ обобщенная рекурсивная 268
◊ общая 259
◊ последовательная 263
◊ последовательная небиективная 281
◊ последовательная параметрически схемно-обратимая 266
◊ последовательная параметрически функционально-обратимая 267
◊ рекурсивная 267
◊ рекурсивная параметрически обратимая 268
◊ рекурсивная параметрически схемно-обратимая 274
◊ рекурсивная параметрически функционально-обратимая 272
Модификация 164
◊ алгоритма шифрования 29
◊ операции суммирования 55
◊ преобразования 231
◊ уникальная 164, 258
◊ управляемой операции 13, 258
◊ эквивалентная 164, 258
Мощность множества 157

Н

Наложение 238
Нарушитель 17
Начальное преобразование 349
Нелинейность булевой функции 244
◊ относительно линейных структур 254
Неправильная пара 297

О

Образующие полиномы 270
Обратный элемент 160
Общая линейность 245
Общая нелинейность 245
Операционный:
◊ S-блок 161
◊ P-блок 165
Операция:
◊ гибкая 15
◊ зависящая от данных 14
◊ зависящая от ключа 14
◊ управляемая 13, 57
◊ управляемая двухместная 55
◊ управляемая подстановочная 55, 57
Определяющее рекуррентное соотношение 270
Отбеливание 339
Отношение "сигнал-шум" 297
Отображение 158
◊ биективное 52, 238
◊ взаимно однозначное 238
◊ инъективное 238
◊ регулярное 240
◊ сюръективное 238

П

Парадокс дней рождений 315
Параметр обратимости 262
Параметрическая обратимость 262

Перестановка 161
◊ один в один 161
◊ управляемая 56
Переходная функция 267
Подстановка 159, 160, 238
◊ с коммутацией разрядов 165
◊ управляемая 57
◊ циклическая 160
Полином Жегалкина 240
Поразрядное:
◊ сложение векторов 157
◊ умножение векторов 157
Почти бент-функция 255
Правильная пара 297
Преобразование:
◊ биективное 168
◊ биективное подстановочное 167
◊ ключей динамическое 323
◊ параметрическое 167
◊ тождественное 166
◊ Уолша-Адамара 241
▫ обратное 242
Протокол рукопожатия 32
Прототипы образующих булевых функций 283
Процедура генерации расширенного ключа 336
◊ стойкая 336
Процедура расширения ключа 323, 324, 328
Процедура расширения управляющего вектора 226

Р

Равенство Парсеваля 244
Расписание ключей 324, 336
Расстояние:
◊ единственности 44
◊ Хэмминга между векторами 159
▫ между булевыми функциями 243
Расширенный (рабочий) ключ 348
Раундовая разность (дифференциал) 295
Раундовая функция 260
Раундовая характеристика 295
Раундовое преобразование 260
Раундовый ключ 260
Рекурсивная модель 184
Робастность 292

С

Секретность практическая 46, 47
Сеть:
◊ Клоса 182
▫ двухуровневая 183
▫ трехуровневая 183
◊ Омега 186
◊ переключаемая 174
◊ перестановочная 174
◊ подстановочно-перестановочная 334
◊ с послойной структурой 174
Симметрическая группа 160
◊ перестановок 163
◊ подстановок 163
Скалярное произведение 159
Сложность:
◊ вычислений 170
◊ комбинационная 171
Смещение 207, 305
Сопряженная функция 240
Средняя линейность 245
Средняя нелинейность 245
Стеганография 94
Стойкость:
◊ безусловная 23
◊ практическая 46
Сумматор:
◊ управляемый 55
◊ по модулю 2^n 275
Схема:
◊ Бенеша 194
◊ Бенеша, модернизированная 195
◊ Ваксмана 196
◊ универсальная 192
Сюръекция 238

Т

Таблица истинности 239
Таблица линейных зависимостей 209
Такт задержки 171
Текст:
◊ закрытый 23
◊ исходный 23
◊ открытый 23

У

Управляемая операция:
◊ двухместная 235, 257, 276

◊ подстановочная 163
  ▫ полная 258
  ▫ эффективно полная 258
Управляемое преобразование 232
Управляемый схемный преобразователь 266
Управляющий:
◊ бит 320
◊ вектор 163, 257
◊ код 163
◊ операнд 276
◊ расширенный вектор 323

Ф

Функция:
◊ выхода 267
◊ четности вектора 159

Х

Характеристика:
◊ линейная 208
◊ наилучшая 208
◊ однораундовая 208
◊ суммарная, линейная 210
Характеристическая последовательность 240

Ш

Шифр:
◊ безусловно стойкий 43
◊ блочный 333
◊ гибкий 29
◊ итеративный, итерационный 335
◊ композиционный 50
◊ недетерминированный 29, 113
◊ перестановки 161
◊ программный 100
◊ совершенно секретный 43
◊ составной 50
◊ условно стойкий 44
Шифртекст 23

Э

Электронная цифровая подпись 34
Элемент криптосистемы:
◊ долговременный 24, 26
◊ легко сменяемый 23, 26
◊ сменяемый 23
Элементарный переключатель 174
Элементарный управляемый сумматор 276
Энергетический спектр 246

Я

Ядро параметрической обратимости 271