

СЕРГЕЙ ПАНАСЕНКО



АЛГОРИТМЫ ШИФРОВАНИЯ

**специальный
справочник**

Классификация
алгоритмов шифрования
и методов их вскрытия

Новейшая история
симметричного
шифрования

Описание алгоритмов



Сергей Панасенко

АЛГОРИТМЫ ШИФРОВАНИЯ

**специальный
справочник**

Санкт-Петербург

«БХВ-Петербург»

2009

УДК 681.3.06
ББК 32.973.26-018.2
П16

Панасенко С. П.

П16 Алгоритмы шифрования. Специальный справочник. —
СПб.: БХВ-Петербург, 2009. — 576 с.: ил.

ISBN 978-5-9775-0319-8

Книга посвящена алгоритмам блочного симметричного шифрования. Дана общая классификация криптографических алгоритмов. Рассмотрено более 50 алгоритмов шифрования: история создания и использования, основные характеристики и структура, достоинства и недостатки. Описаны различные виды криптоаналитических атак на алгоритмы шифрования и на их реализации в виде программных или аппаратных шифраторов. Рассказано о конкурсах по выбору стандартов шифрования США и Евросоюза.

*Для специалистов в области информационных технологий,
преподавателей, студентов и аспирантов*

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	<i>Екатерина Кондукова</i>
Зам. главного редактора	<i>Игорь Шишигин</i>
Зав. редакцией	<i>Григорий Добин</i>
Редактор	<i>Ирина Иноземцева</i>
Компьютерная верстка	<i>Натали Каравашевой</i>
Корректор	<i>Наталья Першакова</i>
Дизайн обложки	<i>Елены Беляевой</i>
Зав. производством	<i>Николай Тверских</i>

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 24.10.08.
Формат 70×100¹/₁₆. Печать офсетная. Усл. печ. л. 46,44.
Тираж 2000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.
Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 978-5-9775-0319-8

© Панасенко С. П., 2008
© Оформление, издательство "БХВ-Петербург", 2008

Оглавление

Введение	1
Глава 1. Классификация алгоритмов шифрования и методов их вскрытия	3
1.1. Криптографические алгоритмы	3
1.2. Категории алгоритмов шифрования.....	7
1.3. Структура алгоритмов симметричного шифрования	8
Алгоритмы на основе сети Фейстеля	9
Алгоритмы на основе подстановочно-перестановочных сетей	11
Алгоритмы со структурой «квадрат».....	11
Алгоритмы с нестандартной структурой	12
1.4. Режимы работы алгоритмов.....	12
Электронная кодовая книга	13
Сцепление блоков шифра	14
Обратная связь по шифртексту	15
Обратная связь по выходу	16
Другие режимы работы.....	18
1.5. Атаки на алгоритмы шифрования	18
Цели атак	18
Классификация атак	19
Количественная оценка криптостойкости алгоритмов шифрования	21
Криптоанализ модифицированных алгоритмов	22
1.6. Криптоаналитические методы, используемые в атаках	23
Метод «грубой силы».....	23
Атаки класса «встреча посередине»	25
Дифференциальный криптоанализ	27
Линейный криптоанализ.....	34
Метод бумеранга	38
Сдвиговая атака	40

Метод интерполяции	42
Невозможные дифференциалы	43
Заключение.....	44
1.7. Атаки на шифраторы, использующие утечку данных	
по побочным каналам	44
Атака по времени выполнения	45
Атаки по потребляемой мощности	46
Другие пассивные атаки	47
1.8. Активные атаки на шифраторы, использующие утечку данных	
по побочным каналам	48
Виды воздействий на шифратор	48
Дифференциальный анализ на основе сбоев	50
Противодействие активным атакам	52
1.9. Криптоаналитические атаки на связанных ключах	53
Расширение ключа.....	53
«Классическая» атака на связанных ключах.....	54
Атакуемые алгоритмы.....	57
Другие возможные проблемы процедуры расширения ключа	59
Глава 2. Новейшая история симметричного шифрования	61
2.1. Конкурс AES.....	61
Алгоритмы — участники конкурса	63
Достоинства и недостатки алгоритмов-финалистов	65
2.2. Конкурс NESSIE.....	69
Глава 3. Описание алгоритмов	73
3.1. Алгоритм ГОСТ 28147-89	73
Описание алгоритма.....	73
Режимы работы алгоритма	75
Криптостойкость алгоритма	79
Анализ таблиц замен	79
Модификации алгоритма и их анализ	80
Анализ полнораундового алгоритма	81
Заключение.....	81
3.2. Алгоритм Aardvark.....	82
3.3. Алгоритм AES (Rijndael).....	84
Структура алгоритма.....	84
Процедура расширения ключа	88
Расшифровывание	89

Отличия AES от исходного алгоритма Rijndael	91
Первичная оценка криптостойкости алгоритма Rijndael.....	92
Криптоанализ алгоритма Rijndael в рамках конкурса AES	93
Криптоанализ алгоритма после конкурса AES.....	95
Заключение.....	97
3.4. Алгоритм Akeelarre	97
Структура алгоритма.....	98
Процедура расширения ключа	101
Криптоанализ алгоритма.....	102
3.5. Алгоритм Anubis	103
Структура алгоритма.....	103
Процедура расширения ключа	107
Достоинства и недостатки алгоритма.....	108
3.6. Алгоритмы Bear, Lion и Lioness	108
Алгоритм Bear.....	109
Алгоритм Lion.....	111
Алгоритм Lioness.....	112
Варианты использования	113
Криптоанализ алгоритмов	114
3.7. Алгоритмы BEAST и BEAST-RK.....	114
Криптостойкость алгоритма BEAST	116
Алгоритм BEAST-RK.....	117
3.8. Алгоритм Blowfish	118
Структура алгоритма.....	119
Процедура расширения ключа	121
Достоинства и недостатки алгоритма.....	122
3.9. Алгоритм Camellia	123
Структура алгоритма.....	123
Таблицы замен	127
Расшифровывание	128
Процедура расширения ключа	129
Криптоанализ алгоритма Camellia	132
3.10. Алгоритм CAST-128	134
3.11. Алгоритм CAST-256	135
Основные характеристики и структура алгоритма	135
Процедура расширения ключа	139
Достоинства и недостатки алгоритма.....	141
3.12. Алгоритм Crypton.....	141
Основные характеристики и структура алгоритма	141

Процедура расширения ключа	145
Достоинства и недостатки алгоритма.....	147
3.13. CS-Cipher.....	148
Структура алгоритма.....	149
Расшифровывание	153
Процедура расширения ключа	154
Достоинства и недостатки алгоритма.....	155
3.14. Алгоритм DEAL	156
Основные характеристики и структура алгоритма	156
Процедура расширения ключа	157
Достоинства и недостатки алгоритма.....	160
3.15. Алгоритм DES и его варианты.....	160
История создания алгоритма	161
Основные характеристики и структура алгоритма	162
Процедура расширения ключа	166
Криптостойкость алгоритма DES	168
Продолжение истории алгоритма	171
Алгоритм Double DES	172
Алгоритм 2-key Triple DES.....	173
Алгоритм 3-key Triple DES.....	174
Режимы работы Double DES и Triple DES.....	175
Режимы работы с маскированием.....	178
Алгоритм Quadruple DES.....	181
Алгоритм Ladder-DES	181
Алгоритм DESX.....	184
Алгоритм DES с независимыми ключами раундов.....	185
Алгоритм GDES	186
Алгоритм RDES	188
Алгоритмы s^2 DES, s^3 DES и s^5 DES	190
Алгоритм Biham-DES.....	191
Алгоритмы x DES ¹ и x DES ²	193
Другие варианты алгоритма DES.....	195
Заключение.....	195
3.16. Алгоритм DFC	195
Основные характеристики и структура алгоритма	195
Процедура расширения ключа	198
Достоинства и недостатки алгоритма.....	199
3.17. Алгоритм E2	201
Структура алгоритма.....	201
Почему алгоритм E2 не вышел в финал конкурса AES.....	206

3.18. Алгоритм FEAL	206
История создания алгоритма	206
Структура алгоритма	207
Процедура расширения ключа	209
Почему FEAL не используется.....	211
3.19. Алгоритм FROG	212
Основные характеристики и структура алгоритма	212
Процедура расширения ключа	214
Форматирование ключа	216
Достоинства и недостатки алгоритма.....	218
3.20. Алгоритм Grand Cru.....	219
Структура алгоритма.....	219
Расшифровывание	225
Процедура расширения ключа	226
Достоинства и недостатки алгоритма.....	229
3.21. Алгоритм Hierocrypt-L1	229
Структура алгоритма.....	229
Процедура расширения ключа	234
Достоинства и недостатки алгоритма.....	239
3.22. Алгоритм Hierocrypt-3	239
Отличия от Hierocrypt-L1.....	239
Процедура расширения ключа	241
Криптоанализ алгоритма.....	246
3.23. Алгоритм HPC	246
Основные характеристики алгоритма	246
Структура раунда.....	247
Процедура расширения ключа	250
Достоинства и недостатки алгоритма.....	253
3.24. Алгоритм ICE.....	254
История создания алгоритма	254
Структура алгоритма.....	254
Варианты алгоритма.....	257
Процедура расширения ключа	257
Криптоанализ алгоритма.....	259
3.25. Алгоритм ICEBERG.....	259
Структура алгоритма.....	259
Процедура расширения ключа	263
Криптоанализ алгоритма.....	265

3.26. Алгоритмы IDEA, PES, IPES.....	265
Основные характеристики и структура.....	266
Процедура расширения ключа.....	268
Криптостойкость алгоритма.....	268
3.27. Алгоритм KASUMI.....	270
Структура алгоритма.....	271
Процедура расширения ключа.....	275
Использование алгоритма.....	277
Криптоанализ алгоритма.....	279
3.28. Алгоритм Khazad.....	282
Структура алгоритма.....	282
Процедура расширения ключа.....	284
Модификация алгоритма.....	285
Криптоанализ алгоритма Khazad.....	287
3.29. Алгоритмы Khufu и Khafre.....	288
История создания алгоритмов.....	288
Структура алгоритма Khufu.....	289
Процедура расширения ключа и таблицы замен.....	290
Алгоритм Khafre.....	292
Сравнение алгоритмов.....	293
Алгоритм Snefru.....	294
Криптоанализ алгоритмов.....	294
3.30. Алгоритм LOKI97.....	295
История создания алгоритма.....	295
Основные характеристики и структура алгоритма.....	296
Процедура расширения ключа.....	299
Почему LOKI97 не вышел в финал конкурса AES.....	300
3.31. Алгоритм Lucifer.....	301
Вариант № 1.....	301
Вариант № 2.....	304
Вариант № 3.....	308
Вариант № 4.....	310
Криптоанализ вариантов алгоритма.....	311
3.32. Алгоритм MacGuffin.....	312
Структура алгоритма.....	313
Процедура расширения ключа.....	315
Криптоанализ алгоритма.....	316
3.33. Алгоритм MAGENTA.....	316
Структура алгоритма.....	317
Достоинства и недостатки алгоритма.....	319

3.34. Алгоритм MARS.....	319
Структура алгоритма.....	320
Процедура расширения ключа.....	326
Достоинства и недостатки алгоритма.....	328
3.35. Алгоритм Mercy.....	328
Структура алгоритма.....	328
Процедура расширения ключа.....	333
Криптостойкость алгоритма.....	334
3.36. Алгоритмы MISTY1 и MISTY2.....	334
Структура алгоритма MISTY1.....	334
Расшифровывание.....	342
Процедура расширения ключа.....	344
Алгоритм MISTY2.....	345
Алгоритм KASUMI.....	346
Криптоанализ алгоритмов MISTY1 и MISTY2.....	347
3.37. Алгоритм Nimbus.....	348
Структура алгоритма.....	348
Процедура расширения ключа.....	349
Достоинства и недостатки алгоритма.....	351
3.38. Алгоритм Noekeon.....	351
Структура алгоритма.....	352
Расшифровывание.....	356
Процедура расширения ключа.....	357
Криптоанализ алгоритма.....	357
3.39. Алгоритм NUSH.....	357
Структура алгоритма.....	357
Расшифровывание.....	362
Процедура расширения ключа.....	363
Криптостойкость алгоритма.....	363
128-битный вариант.....	364
256-битный вариант.....	368
Криптоанализ 128- и 256-битного вариантов алгоритма.....	373
3.40. Алгоритм Q.....	374
Структура алгоритма.....	374
Криптоанализ алгоритма.....	375
3.41. Алгоритм RC2.....	375
Структура алгоритма.....	376
Процедура расширения ключа.....	378
Расшифрование.....	380
Криптостойкость алгоритма.....	381

3.42. Алгоритм RC5.....	381
Структура алгоритма.....	382
Процедура расширения ключа.....	385
Криптоанализ алгоритма.....	386
Варианты RC5.....	387
Продолжение истории алгоритма.....	390
3.43. Алгоритм RC6.....	391
Структура алгоритма.....	391
Процедура расширения ключа.....	393
Достоинства и недостатки алгоритма.....	394
3.44. Алгоритмы SAFER K и SAFER SK.....	394
Структура алгоритма SAFER K-64.....	395
Расшифрование.....	399
Процедура расширения ключа.....	400
Криптоанализ алгоритма SAFER K-64.....	402
Алгоритм SAFER K-128.....	403
Алгоритмы SAFER SK-64, SAFER SK-128 и SAFER SK-40.....	404
Криптоанализ алгоритмов SAFER SK-64 и SAFER SK-128.....	408
3.45. Алгоритм SAFER+.....	408
Структура алгоритма.....	408
Расшифрование.....	412
Процедура расширения ключа.....	413
Криптоанализ алгоритма SAFER+.....	415
Единое расширение ключа SAFER+.....	416
3.46. Алгоритм SAFER++.....	418
Структура алгоритма.....	418
64-битный вариант SAFER++.....	421
Криптоанализ алгоритма SAFER++.....	423
Заключение.....	424
3.47. Алгоритм SC2000.....	424
Структура алгоритма.....	424
Расшифрование.....	428
Процедура расширения ключа.....	429
Криптоанализ алгоритма.....	432
3.48. Алгоритм SERPENT.....	432
Структура алгоритма.....	433
Расшифрование.....	439

Процедура расширения ключа	441
Криптостойкость алгоритма	442
3.49. Алгоритм SHACAL	442
Алгоритм SHACAL-1	442
Алгоритм SHACAL-0	445
Алгоритм SHACAL-2	446
Криптоанализ алгоритма SHACAL-1	449
Криптоанализ алгоритма SHACAL-2	450
3.50. Алгоритмы SHARK и SHARK*	451
3.51. Алгоритм Sha-zam	454
Структура алгоритма	454
Процедура расширения ключа	456
Криптостойкость алгоритма	456
3.52. Алгоритм Skipjack	457
Структура алгоритма	457
Криптостойкость алгоритма	461
3.53. Алгоритм SPEED	462
Структура алгоритма	462
Расшифровывание	465
Процедура расширения ключа	466
Достоинства и недостатки алгоритма	468
3.54. Алгоритм Square	469
Структура алгоритма	469
Процедура расширения ключа	472
Криптостойкость алгоритма	473
3.55. Алгоритмы TEA, XTEA и их варианты	474
Алгоритм TEA	474
Криптоанализ алгоритма TEA	476
Алгоритм XTEA	478
Криптоанализ алгоритма XTEA	480
Алгоритм Block TEA	480
Алгоритм XXTEA	481
3.56. Алгоритмы Twofish и Twofish-FK	483
Структура алгоритма	483
Процедура расширения ключа	486
Алгоритм Twofish-FK	491
Достоинства и недостатки алгоритма	492

Приложение. Таблицы замен.....	493
П1. Алгоритм Blowfish	493
П2. Алгоритм Camellia.....	498
П3. Алгоритм CAST-128.....	500
П4. Алгоритм CAST-256.....	510
П5. Алгоритм Crypton 0.....	515
П6. Алгоритм DES	517
П7. Алгоритм KASUMI	519
П8. Алгоритм MARS.....	522
П9. Алгоритм s^2 DES.....	525
П10. Алгоритм s^3 DES.....	527
П11. Алгоритм s^5 DES.....	529
Литература	531

Введение

Те или иные способы защиты информации использовались людьми на протяжении тысячелетий. Но именно в течение нескольких последних десятилетий криптография — наука о защите информации — переживает невиданный доселе прогресс, обусловленный, как минимум, двумя важными факторами:

- бурное развитие вычислительной техники и ее повсеместное использование привело к тому, что теперь в подавляющем большинстве случаев криптография защищает именно компьютерную информацию;
- тогда как раньше криптография была делом государственных структур, сейчас криптографические методы защиты могут использовать (и используют) обычные люди и организации, хотя бы для защиты своей собственной переписки от посторонних глаз.

То же самое касается и разработки криптографических алгоритмов — известно множество алгоритмов шифрования, и далеко не все из них разработаны «в недрах спецслужб» или научными институтами — встречаются весьма удачные и широко используемые алгоритмы, разработанные частными лицами.

Эта книга посвящена алгоритмам шифрования, а именно, алгоритмам блочного симметричного шифрования, которые, на взгляд автора, являются наиболее широко используемыми в современном компьютерном мире. В книге собраны описания более 50 алгоритмов шифрования (и еще около ста их вариантов), которые автор считает наиболее широко используемыми, наиболее удачными, обладающими наиболее полезными свойствами или просто наиболее интересными по своей структуре или истории.

Как известно, шифрование решает одну из основных проблем защиты информации — а именно, проблему обеспечения конфиденциальности данных. Известно также, что во все времена предпринимались усилия получить именно ту информацию, которая по той или иной причине защищена от посторонних. Поэтому немалая часть книги посвящена криптоаналитическим атакам,

т. е. методам, с помощью которых можно попытаться вскрыть зашифрованную информацию.

Книга состоит из трех глав и приложения.

В *главе 1* приведена общая классификация криптографических алгоритмов, описана структура алгоритмов шифрования и наиболее часто используемые режимы их применения. Там же описаны различные виды криптоаналитических атак на алгоритмы шифрования и на их реализации в виде программных или аппаратных шифраторов.

Глава 2 посвящена краткому обзору новейшей истории алгоритмов шифрования. Она описывает прошедшие в течение последнего десятилетия конкурсы по выбору стандартов шифрования США и Евросоюза. Именно эти конкурсы, по мнению автора, очень сильно повлияли на современное развитие криптографии и криптоанализа.

Глава 3 содержит описание алгоритмов шифрования, т. е., в основном, следующие сведения:

- историю создания и использования алгоритмов;
- основные характеристики и структуру алгоритмов, включая подробное описание используемых в алгоритмах преобразований;
- достоинства и недостатки алгоритмов;
- описание известных методов вскрытия алгоритмов.

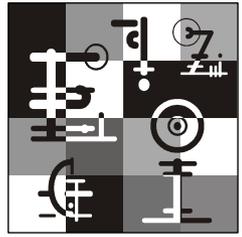
Алгоритмы снабжены схемами, существенно упрощающими понимание структуры конкретного алгоритма. Стоит отметить, что описание большинства алгоритмов достаточно подробно для разработки на их основе программного или аппаратного шифратора.

В *Приложение 1* вынесены громоздкие таблицы, нахождение которых в основном тексте помешало бы его восприятию читателями.

Стоит сказать и о том, что при написании книги не использована какая-либо информация ограниченного пользования — абсолютно все источники информации об алгоритмах (за исключением ряда общедоступных книг и журнальных статей) найдены автором на открытых ресурсах сети Интернет.

Автор выражает благодарность своим коллегам по работе в ООО «АНКАД», многие из которых оказывали автору различное содействие при создании этой книги.

Автор будет признателен читателям за любые замечания по этой книге, в том числе критические. Жду ваших писем по адресу serg@panasenko.ru.



Глава 1

Классификация алгоритмов шифрования и методов их вскрытия

В этой главе дана общая классификация криптографических алгоритмов, приведены структуры алгоритмов шифрования и наиболее часто используемые режимы их применения. Здесь же описаны различные виды криптоаналитических атак на алгоритмы шифрования и на их реализации в виде программных или аппаратных шифраторов.

1.1. Криптографические алгоритмы

Для начала обсудим, какие же типы криптографических алгоритмов используются для защиты информации. Криптоалгоритмы, прежде всего, делятся на три категории (рис. 1.1) [7, 263]:

- *бесключевые* алгоритмы, которые не используют каких-либо ключей в процессе криптографических преобразований;
- *одноключевые* алгоритмы, использующие в своих вычислениях некий секретный ключ;
- *двухключевые* алгоритмы, в которых на различных этапах вычислений применяются два вида ключей: секретные и открытые.

Рассмотрим вкратце основные типы криптоалгоритмов.

- *Хэш-функции*. Выполняют «свертку» данных переменной длины в последовательность фиксированного размера — фактически это контрольное суммирование данных, которое может выполняться как с участием некоего ключа, так и без него. Такие функции имеют весьма широкое применение в области защиты компьютерной информации, например:
 - для подтверждения целостности любых данных в тех случаях, когда использование электронной подписи (см. далее) невозможно (например, из-за большой ресурсоемкости) или является избыточным;

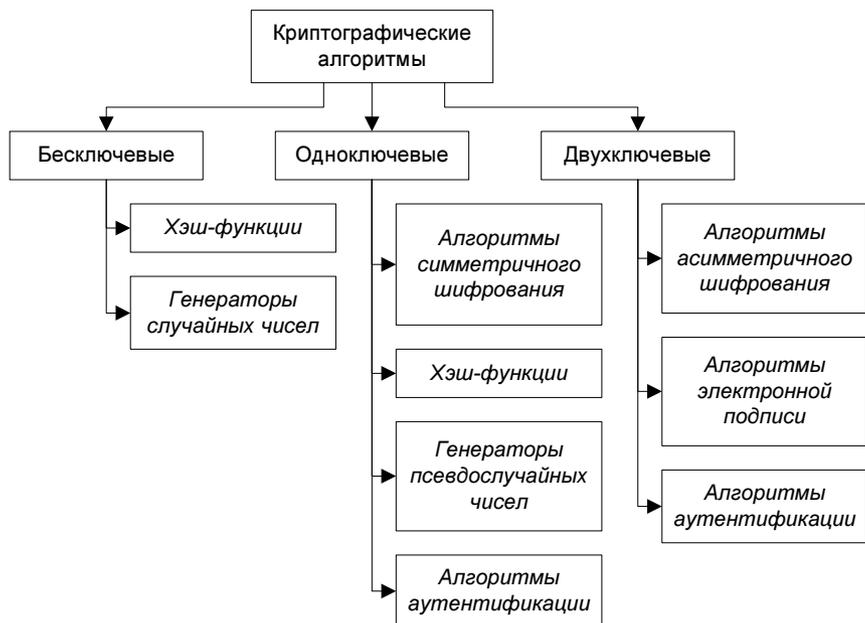


Рис. 1.1. Классификация криптографических алгоритмов

- в самих схемах электронной подписи — подписывается обычно хэш данных, а не все данные целиком;
- в различных схемах аутентификации пользователей (см. далее).

- *Генераторы случайных чисел.* Случайные числа необходимы, в основном, для генерации секретных ключей шифрования, которые, в идеале, должны быть абсолютно случайными. Нужны они и для вычисления электронной цифровой подписи, и для работы многих алгоритмов аутентификации.
- *Алгоритмы симметричного шифрования* — алгоритмы шифрования, в которых для зашифровывания и расшифровывания используется один и тот же ключ, или ключ расшифровывания легко вычисляется из ключа зашифровывания и наоборот.

Симметричное шифрование бывает двух видов: блочное и потоковое.

- *Блочное шифрование* — в этом случае информация разбивается на блоки фиксированной длины (например, 64 или 128 битов), после чего эти блоки поочередно шифруются. Причем в различных алгоритмах шифрования или даже в разных режимах работы одного и того же алгоритма блоки могут шифроваться независимо друг от друга или «со сцеплением» — когда результат зашифровывания текущего блока данных

зависит от значения предыдущего блока или от результата зашифрования предыдущего блока (см. разд. 1.4). Данная книга посвящена описанию криптоалгоритмов именно этой, наиболее обширной, категории алгоритмов шифрования.

- *Потоковое шифрование* — необходимо, прежде всего, в тех случаях, когда информацию невозможно разбить на блоки — скажем, некий поток данных, каждый символ которых должен быть зашифрован и отправлен куда-либо, не дожидаясь остальных данных, достаточных для формирования блока. Поэтому алгоритмы потокового шифрования шифруют данные побитно или посимвольно. Хотя стоит сказать, что некоторые классификации не разделяют блочное и потоковое шифрование, считая, что потоковое шифрование — это шифрование блоков единичной длины [263].
- *Генераторы псевдослучайных чисел.* Не всегда возможно получение абсолютно случайных чисел — для этого необходимо наличие качественных аппаратных генераторов. Однако на основе алгоритмов симметричного шифрования можно построить очень качественный генератор псевдослучайных чисел.
- *Алгоритмы аутентификации.* Позволяют проверить, что пользователь (или удаленный компьютер) действительно является тем, за кого себя выдает. Простейшая схема аутентификации — парольная — не требует наличия каких-либо криптографических ключей, но доказанно является слабой. А с помощью секретного ключа можно построить заметно более сильные схемы аутентификации. Пример аутентификации пользователя сервером (рис. 1.2):
 - Этап 1. Сервер генерирует случайное число
 - Этап 2. и отправляет его пользователю.
 - Этап 3. Пользователь зашифровывает полученное число секретным ключом и отправляет результат серверу.
 - Этап 4. Сервер расшифровывает полученные данные таким же секретным ключом
 - Этап 5. и сравнивает с исходным числом.Равенство чисел означает, что пользователь обладает требуемым секретным ключом, т. е. ему удалось доказать свою легитимность.
- *Алгоритмы асимметричного шифрования.* Применяют два вида ключей: открытый ключ для зашифрования информации и секретный — для расшифрования. Секретный и открытый ключи связаны между собой достаточно

сложным соотношением, главное в котором — легкость вычисления открытого ключа из секретного и невозможность (за ограниченное время при реальных ресурсах) вычисления секретного ключа из открытого при достаточно большой размерности операндов. Любая информация, зашифрованная общедоступным открытым ключом, может быть расшифрована только обладателем секретного ключа, из которого был вычислен данный открытый (рис. 1.3):

Этап 1. Пользователь В зашифровывает сообщение на открытом ключе пользователя А (который когда-либо передал его пользователю В).

Этап 2. Пользователь А расшифровывает сообщение своим секретным ключом.

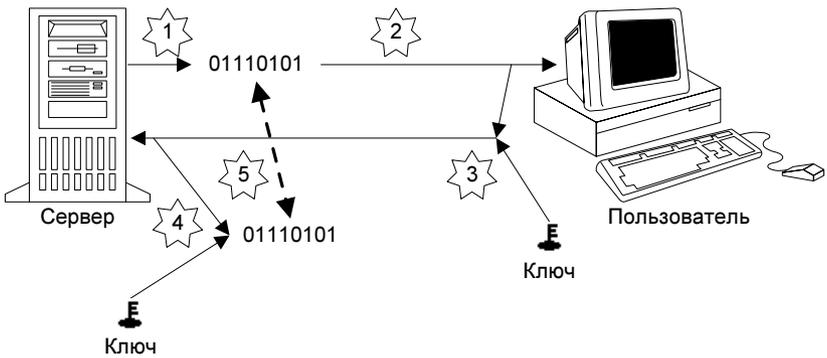


Рис. 1.2. Пример аутентификации пользователя сервером

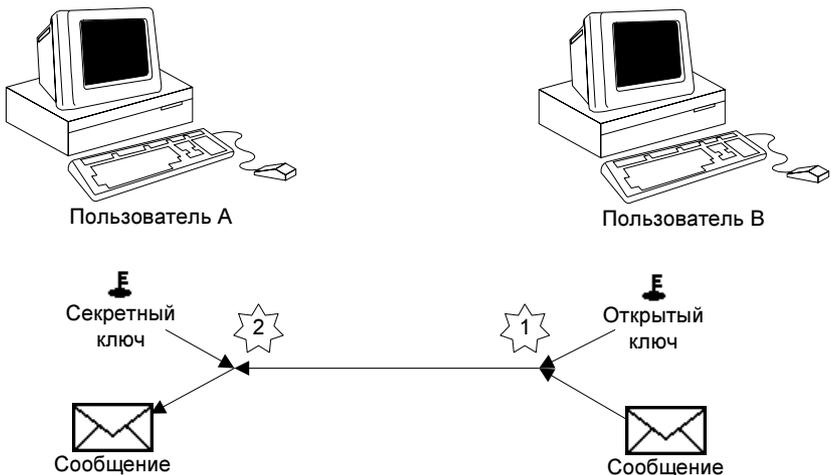


Рис. 1.3. Асимметричное шифрование

- *Алгоритмы электронной подписи.* Используют секретный ключ для вычисления электронной цифровой подписи данных, а вычисляемый из него открытый — для ее проверки.

Как и асимметричное шифрование, это двухключевые алгоритмы с таким же простым вычислением открытого ключа из секретного и практической невозможностью обратного вычисления. Однако назначение ключей является совершенно другим:

- секретный ключ используется для вычисления электронной подписи;
- открытый ключ необходим для ее проверки.

При соблюдении безопасного хранения секретного ключа, никто, кроме его владельца, не в состоянии вычислить верную электронную подпись какого-либо электронного документа.

1.2. Категории алгоритмов шифрования

Шифрование — это основной криптографический метод защиты информации, обеспечивающий ее конфиденциальность [14, 21].

Шифрование информации — это преобразование открытых данных в зашифрованные и наоборот. Первая часть этого процесса называется зашифрованием, вторая — расшифрованием.

Шифрование можно представить в виде следующих формул:

$$C = E_{k_1}(M);$$

$$M' = D_{k_2}(C).$$

где:

- M (message) — открытая информация, в литературе по защите информации часто обозначается словосочетанием *открытый текст*;
- C (cipher text) — полученный в результате зашифрования *шифртекст* (или *криптограмма*);
- E (encryption) — функция зашифрования, выполняющая криптографические преобразования над исходным текстом;
- k_1 (key) — параметр функции E , называемый *ключом зашифрования*;
- M' — информация, полученная в результате расшифрования;
- k_2 — ключ, с помощью которого выполняется расшифрование информации;
- D (decryption) — функция расшифрования, выполняющая обратные зашифрованию криптографические преобразования над шифртекстом.

В стандарте ГОСТ 28147-89 (см. разд. 3.1) понятие «ключ» определено следующим образом: «Конкретное секретное состояние некоторых параметров алгоритма криптографического преобразования, обеспечивающее выбор одного преобразования из совокупности всевозможных для данного алгоритма преобразований» [4]. Иными словами, ключ является уникальным элементом, с помощью которого можно варьировать результат работы алгоритма шифрования: один и тот же открытый текст при использовании различных ключей будет зашифрован по-разному.

Для того чтобы результат операций зашифровывания и последующего расшифровывания совпал с исходным сообщением (т. е. для получения $M' = M$), необходимо одновременное выполнение двух условий:

- функция расшифровывания D должна соответствовать функции зашифровывания E ;
- ключ расшифровывания k_2 должен соответствовать ключу зашифровывания k_1 .

При отсутствии верного ключа k_2 получить исходное сообщение $M' = M$ невозможно, если для зашифровывания использовался криптографически стойкий алгоритм шифрования. Понятие *критическостойкости* описано в разд. 1.5.

Алгоритмы шифрования можно разделить на две категории.

- Алгоритмы симметричного шифрования. Определяются следующим соотношением ключей зашифровывания и расшифровывания:

$$k_1 = k_2 = k,$$

т. е. функциями E и D используется один и тот же ключ шифрования.

- Алгоритмы асимметричного шифрования, в которых ключ зашифровывания k_1 вычисляется из ключа k_2 таким образом, что обратное вычисление невозможно; например, по следующей формуле:

$$k_1 = a^{k_2} \bmod p,$$

где a и p — параметры используемого алгоритма шифрования.

1.3. Структура алгоритмов симметричного шифрования

подавляющее большинство современных алгоритмов шифрования работают весьма схожим образом: над шифруемым текстом выполняется некое преобразование с участием ключа шифрования, которое повторяется определенное число раз (раундов). При этом по виду повторяющегося преобразования алгоритмы шифрования принято делить на несколько категорий. Здесь также

существуют различные классификации, приведу одну из них. Итак, по своей структуре алгоритмы шифрования классифицируются следующим образом [9, 50, 187, 284, 395].

Алгоритмы на основе сети Фейстеля

Сеть Фейстеля (Feistel network) подразумевает разбиение обрабатываемого блока данных на несколько субблоков (чаще всего — на два), один из которых обрабатывается некоей функцией f и накладывается на один или несколько остальных субблоков. На рис. 1.4 приведена наиболее часто встречающаяся структура алгоритмов на основе сети Фейстеля.

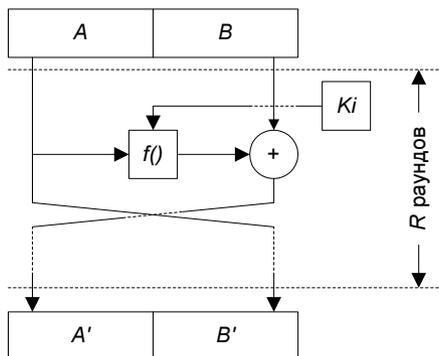


Рис. 1.4. Сеть Фейстеля

Дополнительный аргумент функции f , обозначенный на рис. 1.4 как K_i , называется *ключом раунда*. Ключ раунда является результатом обработки ключа шифрования процедурой расширения ключа, задача которой — получение необходимого количества ключей K_i из исходного ключа шифрования относительно небольшого размера (в настоящее время достаточным для ключа симметричного шифрования считается размер 128 битов). В простейших случаях процедура расширения ключа просто разбивает ключ на несколько фрагментов, которые поочередно используются в раундах шифрования; существенно чаще процедура расширения ключа является достаточно сложной, а ключи K_i зависят от значений большинства битов исходного ключа шифрования.

Наложение обработанного субблока на необработанный чаще всего выполняется с помощью логической операции «исключающее или» (Exclusive OR, XOR), как показано на рис. 1.4. Достаточно часто вместо XOR здесь используется сложение по модулю 2^n , где n — размер субблока в битах. После наложения субблока меняются местами, т. е. в следующем раунде алгоритма обрабатывается уже другой субблок данных.

Такая структура алгоритмов шифрования получила свое название по имени Хорста Фейстеля (Horst Feistel) — одного из разработчиков алгоритма шифрования Lucifer (см. разд. 3.31) и разработанного на его основе алгоритма DES (Data Encryption Standard) — бывшего (но до сих пор широко используемого) стандарта шифрования США (см. разд. 3.15). Оба этих алгоритма имеют структуру, аналогичную показанной на рис. 1.4. Среди других алгоритмов, основанных на сети Фейстеля, можно привести в пример отечественный стандарт шифрования ГОСТ 28147-89 (см. разд. 3.1), а также другие весьма известные алгоритмы: RC5 (см. разд. 3.42), Blowfish (см. разд. 3.8), TEA (см. разд. 3.55), CAST-128 (см. разд. 3.10) и т. д.

На сети Фейстеля основано большинство современных алгоритмов шифрования — благодаря множеству преимуществ подобной структуры, среди которых стоит отметить следующие:

- алгоритмы на основе сети Фейстеля могут быть сконструированы таким образом, что для зашифровывания и расшифровывания может использоваться один и тот же код алгоритма — разница между этими операциями может состоять лишь в порядке применения ключей K_i ; такое свойство алгоритма наиболее полезно при его аппаратной реализации или на платформах с ограниченными ресурсами; в качестве примера такого алгоритма можно привести ГОСТ 28147-89;
- алгоритмы на основе сети Фейстеля являются наиболее изученными — таким алгоритмам посвящено огромное количество криптоаналитических исследований, что является несомненным преимуществом как при разработке алгоритма, так и при его анализе.

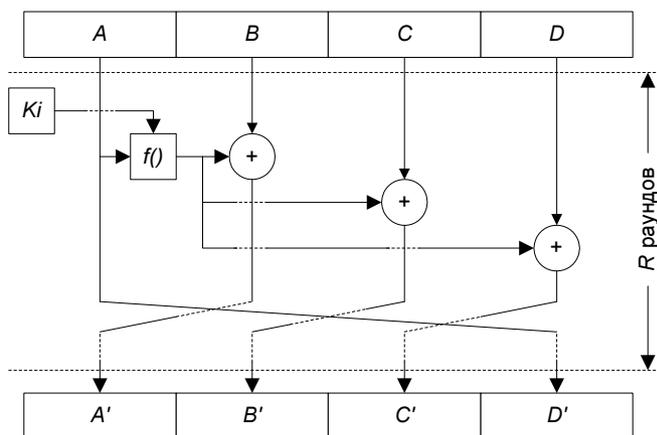


Рис. 1.5. Расширенная сеть Фейстеля

Существует и более сложная структура сети Фейстеля, пример которой приведен на рис. 1.5. Такая структура называется *обобщенной* или *расширенной* сетью Фейстеля и используется существенно реже традиционной сети Фейстеля. Примером такой сети Фейстеля может служить алгоритм RC6 (см. разд. 3.43).

Алгоритмы на основе подстановочно-перестановочных сетей

В отличие от сети Фейстеля, *SP-сети* (Substitution-permutation network, подстановочно-перестановочная сеть) обрабатывают за один раунд целиком шифруемый блок. Обработка данных сводится, в основном, к заменам (когда, например, фрагмент входного значения заменяется другим фрагментом в соответствии с таблицей замен, которая может зависеть от значения ключа K_i) и перестановкам, зависящим от ключа K_i (упрощенная схема показана на рис. 1.6). Впрочем, такие операции характерны и для других видов алгоритмов шифрования, поэтому, на мой взгляд, название «подстановочно-перестановочная сеть» является достаточно условным.

SP-сети являются гораздо менее распространенными, чем сети Фейстеля; в качестве примера SP-сетей можно привести алгоритмы Serpent (см. разд. 3.48) или SAFER+ (см. разд. 3.45).

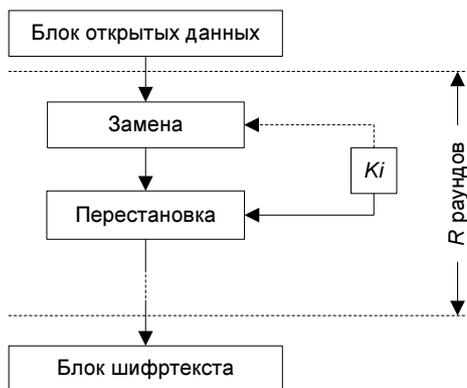


Рис. 1.6. SP-сеть

Алгоритмы со структурой «квадрат»

Для структуры «квадрат» (Square) характерно представление шифруемого блока данных в виде двумерного байтового массива. Криптографические преобразования могут выполняться над отдельными байтами массива, а также над его строками или столбцами.

Эта структура получила свое название от алгоритма Square (*см. разд. 3.54*), который был разработан в 1996 г. Винсентом Риджменом (Vincent Rijmen) и Джоан Деймен (Joan Daemen) — будущими авторами алгоритма Rijndael (*см. разд. 3.3*), ставшего в США новым стандартом шифрования AES после победы на открытом конкурсе (*см. разд. 2.1*). Алгоритм Rijndael также имеет Square-подобную структуру; кроме того, в качестве примера можно привести алгоритмы SHARK (более ранняя разработка Риджмена и Деймена, *см. разд. 3.50*) и Crypton (*см. разд. 3.12*). Недостатком алгоритмов со структурой «квадрат» является их недостаточная изученность, что не помешало алгоритму Rijndael стать новым стандартом шифрования США.

Алгоритмы с нестандартной структурой

Изобретательность безгранична, поэтому как-либо классифицировать все возможные варианты алгоритмов шифрования представляется сложным, т. е. существуют такие алгоритмы, которые невозможно причислить ни к одному из перечисленных типов. В качестве примера алгоритма с нестандартной структурой можно привести уникальный по своей структуре алгоритм FROG (*см. разд. 3.19*), в каждом раунде которого по достаточно сложным правилам выполняется модификация двух байтов шифруемых данных.

Строгие границы между описанными выше структурами не определены, поэтому достаточно часто встречаются алгоритмы, причисляемые различными экспертами к разным типам структур. Например, алгоритм CAST-256 (*см. разд. 3.11*) относится его автором к SP-сети, а многими экспертами называется расширенной сетью Фейстеля. Другой пример — алгоритм HPC (*см. разд. 3.23*), называемый его автором сетью Фейстеля, но относимый экспертами к алгоритмам с нестандартной структурой [50, 187].

1.4. Режимы работы алгоритмов

В 1980 г. в США был принят стандарт [151], определяющий режимы работы алгоритма DES — стандарта шифрования США (*см. разд. 3.15*). Можно сказать, что этот стандарт уточнял подробности реализации DES для различных применений.

В стандарте были определены следующие режимы работы алгоритма DES:

- электронная кодовая книга ECB (Electronic Code Book);
- сцепление блоков шифра CBC (Cipher Block Chaining);
- обратная связь по шифртексту CFB (Cipher Feed Back);
- обратная связь по выходу OFB (Output Feed Back).

Рассмотренные далее режимы не привязаны к конкретному алгоритму — фактически любые алгоритмы блочного симметричного шифрования могут быть использованы (и используются) в данных режимах работы.

Электронная кодовая книга

Наиболее простым из них является режим ECB. Суть его состоит в том (рис. 1.7), что каждый блок шифруемых данных «прогоняется» через алгоритм шифрования отдельно и независимо от других блоков:

$$C_i = E_k(M_i),$$

где:

- E_k — функция зашифровывания на ключе k ;
- M_i и C_i — блоки открытого текста и соответствующие им блоки шифртекста.

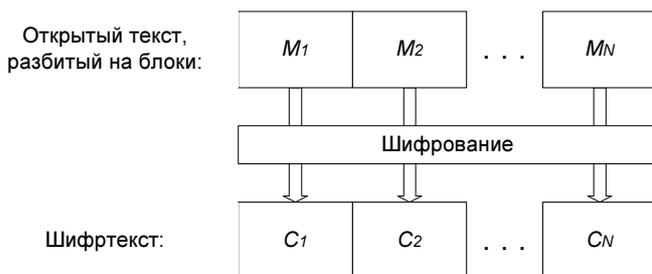


Рис. 1.7. Режим ECB

Аналогично выполняется и расшифровывание — блоки шифртекста обрабатываются поочередно и независимо.

Режим ECB плох тем, что если открытый текст содержит какое-либо количество блоков с одинаковым содержанием (например, большой массив, проинициализированный нулевыми или единичными битами), то и шифртекст будет содержать такое же количество одинаковых блоков. Это непозволительно, поскольку дает криптоаналитику информацию о структуре зашифрованной информации, что может существенно облегчить вскрытие алгоритма шифрования (т. е. получение открытого текста из зашифрованного или вычисление ключа шифрования). Поэтому режим ECB должен использоваться только для шифрования ключей друг на друге в многоключевых схемах [14]. Допускается также шифрование небольших фрагментов данных при условии их неповторяемости [263].

Есть и еще один недостаток — в режиме ECB алгоритм шифрует данные только поблочно. При необходимости, например, зашифровать алгоритмом DES 8 битов придется дополнить эти данные до 64-битного размера блока, зашифровать блок целиком, а при расшифровывании отбросить дополняющие биты. Такое не всегда возможно.

Достоинство режима ECB — простота реализации по сравнению с другими режимами. Однако, по словам известного криптолога Брюса Шнайера (Bruce Schneier), «из-за своей простоты в большинстве существующих коммерческих программ используется режим ECB, хотя этот режим наиболее уязвим к вскрытию» [28].

Сцепление блоков шифра

Существенно более стойким является режим CBC. В этом режиме (рис. 1.8) перед шифрованием каждого i -го блока шифруемых данных выполняется его побитовое сложение по модулю 2 с результатом шифрования предыдущего, $(i - 1)$ -го, блока. То есть для режима CBC шифрование выглядит так:

$$C_i = E_k(M_i \oplus C_{i-1}).$$

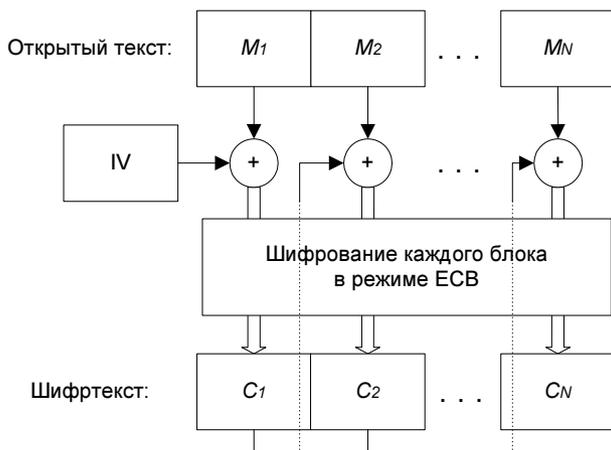


Рис. 1.8. Режим CBC

Видно, что результат шифрования каждого блока зависит не только от содержимого шифруемого блока, но и от всех предыдущих блоков открытого текста. При шифровании первого блока открытого текста вместо результата

зашифровывания предыдущего блока используется *вектор инициализации* (Initialization Vector, IV):

$$C_1 = E_k(M_1 \oplus IV).$$

Варьируя значение вектора инициализации, можно получать различные шифртексты для одинаковых открытых текстов. Естественно, при расшифровании шифртекста, полученного в режиме CBC, вектор инициализации должен быть известен. Это справедливо и для рассмотренных далее режимов CFB и OFB.

Аналогично предыдущему режиму, данные, размер которых меньше 64-битного блока, придется перед обработкой дополнять до 64 битов.

Режим CBC используется непосредственно для зашифровывания данных, в том числе в больших объемах. Кроме того, последний блок шифртекста C_N может использоваться для контроля целостности сообщений, поскольку его значение зависит от содержимого всех блоков открытого текста, вектора инициализации и ключа:

$$C_N = f(M_1, M_2, \dots, M_N, IV, k).$$

Обратная связь по шифртексту

Режим CFB более сложен в реализации, чем два предыдущих. Шифрование данных в этом режиме выполняется следующим образом (рис. 1.9):

1. Вектор инициализации IV записывается в регистр 1.
2. 64-битное содержимое регистра 1 зашифровывается, результат помещается в регистр 2.
3. Из регистра 2 выбираются L левых битов ($1 \leq L \leq 64$), которые накладываются операцией XOR на L -битный блок шифруемого текста. Результат этого шага — L -битный блок шифртекста.
4. Содержимое регистра 1 сдвигается влево на L битов.
5. Регистр 1 дополняется справа L -битным блоком шифртекста. При необходимости продолжить шифрование данных шаги 2–5 повторяются в цикле до обработки всех шифруемых данных.

Фактически в режиме CFB алгоритм шифрования на основе вектора инициализации, ключа шифрования и предыдущих блоков шифртекста генерирует псевдослучайную последовательность, которая накладывается на открытый текст. Поскольку операция XOR обладает следующим свойством:

$$(x \oplus y) \oplus y = x$$

для любых значений x и y , то для расшифровывания следует сгенерировать такую же последовательность. Не имея ключа и значения вектора инициализации, такую же последовательность сгенерировать невозможно.

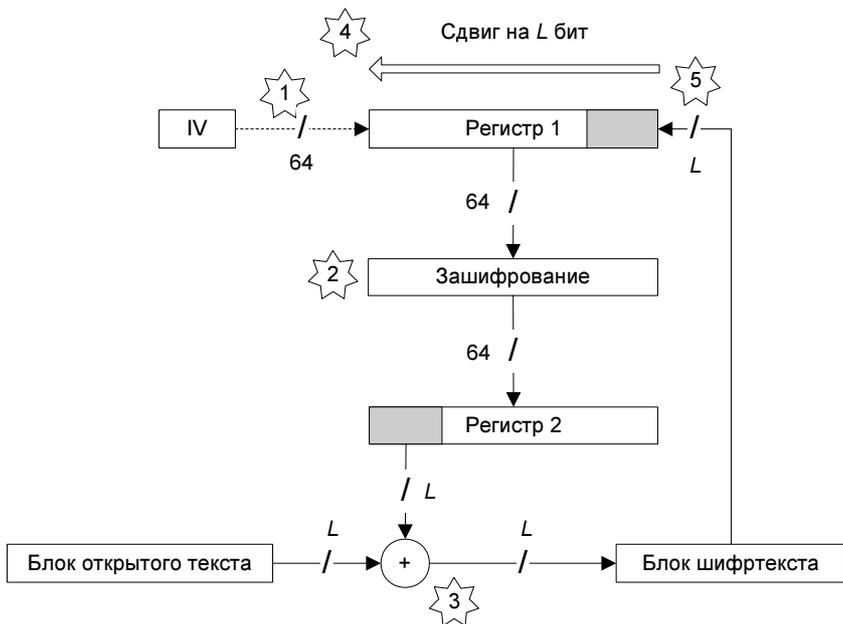


Рис. 1.9. Режим CFB

Обратная связь по выходу

Режим шифрования OFB похож на предыдущий (рис. 1.10):

1. Вектор инициализации IV записывается в регистр 1.
2. Содержимое регистра 1 зашифровывается, результат помещается в регистры 1 и 2.
3. Из регистра 2 выбираются L левых битов, которые накладываются операцией XOR на L -битный блок шифруемого текста, в результате получается L -битный блок шифртекста. При необходимости продолжить шифрование шаги 2 и 3 повторятся в цикле.

Здесь приведена более поздняя версия режима OFB (описанная в стандарте ISO 10116); согласно [151] в регистр 1 «возвращался» не полностью 64-битный результат зашифровывания регистра 1, а младшие L битов (что больше напоминает режим CFB); исходный вариант режима OFB считается менее криптографически стойким, чем модифицированный [28, 263].

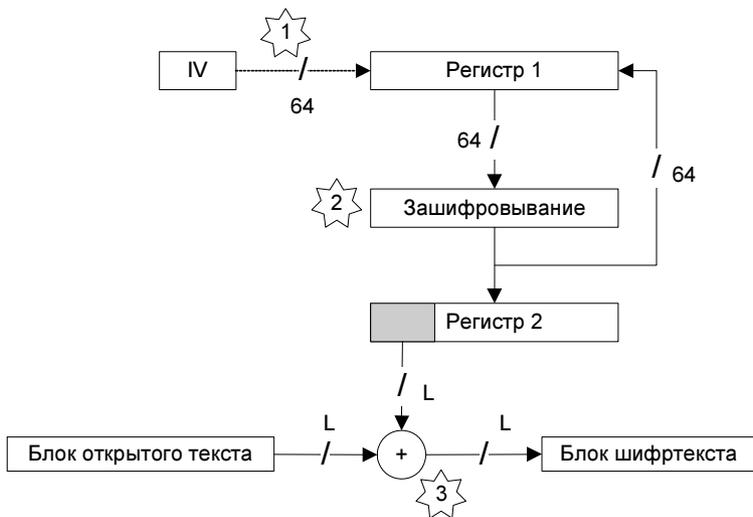


Рис. 1.10. Режим OFB

Существует и более простой режим OFB — *режим счетчика* [263], в котором обратная связь вообще отсутствует, а содержимое регистра 1 перед каждым его зашифровыванием увеличивается на 1.

Аналогично режиму CFB, для расшифровывания необходимо сгенерировать ту же последовательность и наложить ее на шифртекст.

Отличие от предыдущего режима лишь в том, что значение регистра 1 заменяется в цикле его же зашифрованным содержимым. Отсюда проистекает важное свойство режима OFB — накладываемая на шифруемый текст последовательность зависит только от значения вектора инициализации и ключа шифрования. То есть шифрующую последовательность можно сгенерировать заранее и, например, использовать ее в периоды максимальной нагрузки на шифрующий компьютер или устройство, восполняя последовательность в фоновом режиме. Такая возможность весьма важна для серверных компонент распределенных систем, поскольку позволяет существенно увеличить их пиковую производительность.

Еще одно важное отличие режима OFB от остальных состоит в том, что при возникновении ошибки в одном бите шифртекста после расшифровывания возникает ошибка только в одном бите расшифрованных данных, тогда как в остальных режимах ошибочно расшифруются один или два блока.

Стоит сказать и о том, что в режимах OFB и CFB алгоритмы блочного симметричного шифрования можно использовать в качестве потокового шифра (см. разд. 1.1), установив L равным размеру символов потока (например, 1 или 8).