ЛЕКЦИЯ 3

БЛОЧНЫЕ ШИФРЫ. РЕЖИМЫ ШИФРОВАНИЯ

Клод Шенон в своем труде «Теория связи в секретных сиситемах» проклассифицировал ранее существовавшие шифры (моноалфавитные, полиалфавитные, транспозиционные шифры и шифры замены) и показал, что они ненадежны ко многим атакам. Например, атака открытым текстом позволяет вскрыть любой из этих шифров.

Шенон показал, что для создания надежного шифра нужно взять комбинацию шифра замены и транспозиционного шифра.

Идеальные шифры возможны, но очень непрактичны: для таких шифров длина ключа должны быть не меньше длины самого сообщения. Если использовать один и тот же ключ для двух блоков текста, то шифр перестает быть абсолютно надежным.

1. Шифр Lucifer

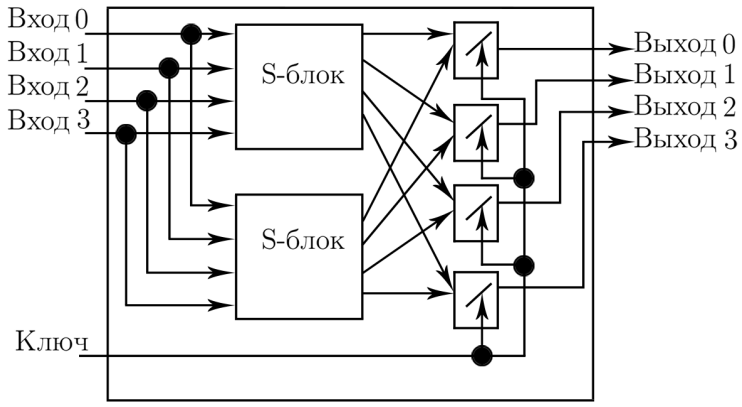

Рис. 3.1

Этот шифр является моноалфавитным.

Замены, которые даются схемой, можно прелставить в виде функции, но эту функ-

цию нельзя будет разложить до линейных членов (нелинейная таблица замен).

В зависимости от ключа выбирается одна или другая таблица замен (выбор конкретного s-блока).

S-блок устроен таким образом, что изменение одного бита на входе ведет к изменению примерно половины бит на выходе.

Теперь рассмотрим следующую версию шифра Lucifer.

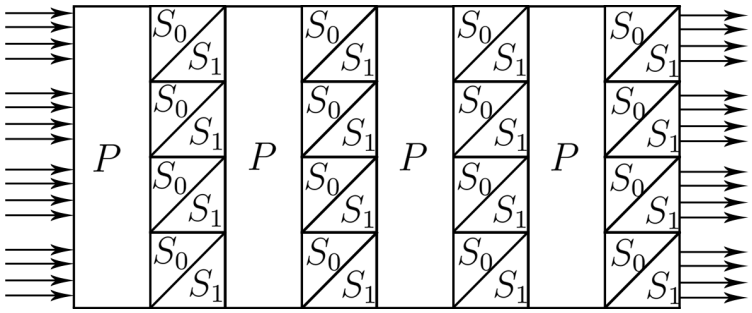

Рис. 3.2

Есть 16 бит на входе. Они сначала подаются на p-блок — блок перестановок (permutation), а затем на s-блок — блок замен (substitution). P-блок менял между собой отдельные биты, а s-блок выпонял нелинейную замену битов.

После четырех уровней такая структура шифра приводит к **лавинному эффекту** (avalanche effect) — влияние одного бита на входе на все биты на выходе.

2. Шифр DES (Digital Enscription Standart)

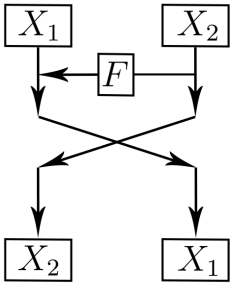

Рис. 3.3

Здесь применяются так называемые **ячейки Фейстеля** — это способы организовать блочный алгоритм шифрования таким образом, что криптографу нет нужды заботиться о восстановимости шифра.

Суть алгоритма состоит в следующем: текст разделяется на две половины. Затем к правой части применяется функция Фейстеля. Далее результат применения складывается побитово по модулю 2 с левой частью, после чего подставляется в правую часть следующей итерации, а правая часть подставляется в левую (без изменения).

После одной итерации в ячейке Фейстеля меняется только половина текста. Эти итерации (раунды) повторяются множество раз. Благодаря этому обеспечивается эффект надежного шифра.

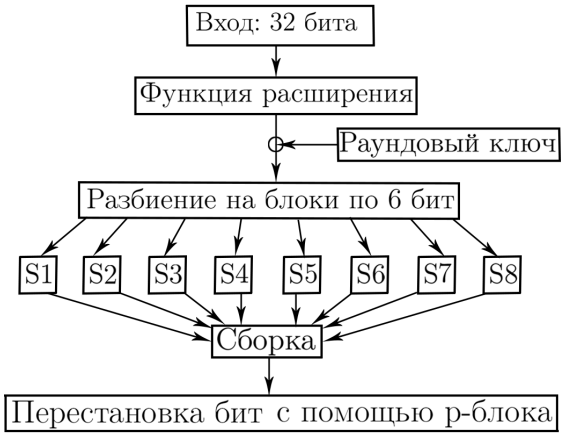

Рис. 3.4

На вход подается 32 бита, затем расширяется до 48 бит. Далее для каждой ячейки Фейстеля побитово прибавляется раундовый ключ (размером 48 бит). Потом разбиваем то, что получилось, на блоки по 6 бит, после чего каждый из восьми блоков подается на соответствующий s-блок. Там выполняется нелинейная замена, и потом применяется перестановка бит.

В 70-х годах компания IBM, которая разработала этот шифр, предполагала, что кодирование будет вестись 128-ю битами. Организация NSA, которой американское правительство отправило шифр на экспертизу, сократила 128 бит до 56 и предоставила

другие s-блоки. Как выяснилось через 20 лет, предложенные s-блоки оказались устойчивыми к дифференциальному и линейному криптоанализу, который появился только в 90-х годах.

Обратимость текста гарантируется конструкцией ячейки Фейстеля.

3. Другие шифры

На ячейке Фейстеля основано многио других шифров (например, **ГОСТ 28147-89**). В этом шифре каждый из блоков по 64 бита разбивается на блоки по 32 бита, затем происходит сложение с ключом по модулю 2^{32} (сложение с переносом, а не побитовое). После этого выполняется преобразование в s-блоках (8 s-блоков по 4 бита), а затем происходит циклический сдвиг на 11 бит. Этот сдвиг влияет сразу на два блока, причём по-разному, благодаря чему достигается лавинный эффект.

В этом шифре ключ имеет размер 256 бит.

Один из s-блоков DES взломали, благодаря чему надежность этого шифра заметно упала. Американское правительство объявило конкурс на создание нового стандарта шифрования. Так на свет появился AES — Advanced Encription Standart.

С помощью ячейки Фейстеля было создано большое количество шифров, в которых ячейка Фейстеля видоизменялась, копировалась и т. д.

К функции шифрования можно предъявить следующие **требования**:

1. Наличие лавинного эффекта
2. Правильность выбора s-блоков
3. Устойчивость к линейному и дифференциальному криптоанализу
4. Производительность алгоритмов
5. Простота анализа алгоритмов

4. Режимы шифрования

Суть статистического криптоанализа состоит в следующем: пусть есть несколько блоков данных, которые подвергаются замене на какие-то другие, причём функция замены одна и та же.

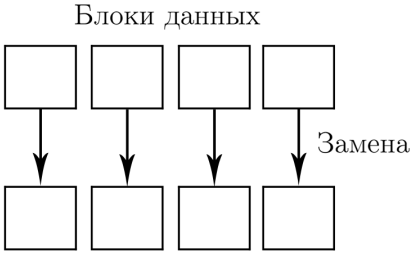

Рис. 3.5

Тогда, сравнивая два текста, можно попробовать восстановить функцию замены. Если функция замены сложна, то нужно передать фразу, которая содержит все буквы алфавита, и тогда шифр будет взломан.

4.1. Electrionic Codebook (ECB)

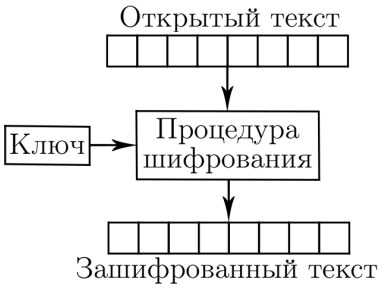

Рис. 3.6

Определенные слова (например, имена собственные) заменяются через другие слова в электронной книге. Есди у слова нет аналога в книге, то оно шифруется побуквенно.

Преимущество ECB состоит в том, что можно зашифровывать отдельные блоки независимо (на параллельных процессорах). Но также есть и минусы. Например, здесь допустима атака повтором (функция шифрования не защищает от повтора текста). Например, злоумышленник может заставить провести платеж в банке несколько раз. Но от этого можно защититься нумерацией операций.

Также остается актуальной проблема статистического криптоанализа. Статистические особенности исходных данных будут заметны на каком-то макроуровне (неустойчивость к атаке по словарю).

Основной недостаток этого режима шифрования — это отсутствие отслеживания потери блоков.

4.2. Cipher Block Chaining (CBC)

В этом режиме результат шифрования складывается с открытым текстом следующего блока **(шифр Вернама)**. В самом начале используется вектор инициализации, который можно менять от операции к операции.

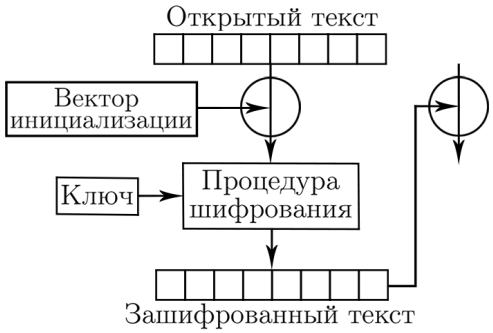

Рис. 3.7

У этого режима шифрования есть недостаток. Если хотя бы один бит при расшифровании повредится, то блок испортится целиком. Следующий блок тоже испортится, но там испортится только один бит. Все остальные биты у нас не будут затронуты. Таким образом, данный режим сцепления блоков относится к **самовосстанавливающимся**

режимам.

Преимущества данного режима: он самовосстанавливающийся на последующем блоке, устойчив к атаке по словарю, то есть устойчив к атаке по кодовой книге (если использовать 64 бита, то можно составить словарь из 2^{64} вариантов, и тогда шифр будет вскрыт).

Недостатки: распространяет ошибку на следующий блок, для очень крупных текстов возможно выделение особенностей текста (если используется 32 бита, то очень крупным можно считать 2^{64}).

Если каждый раз будем использовать одинаковый вектор инициализации, то одинаковый текст будет начинаться одинаково.

4.3. Cipher FeedBack (CFB)

Суть этого метода — обратная связь по шифротексту.

Здесь вектор инициализации складывается с результатом шифрования (раньше складывали с открытым текстом).

Предположим, что в блоке, который передали по радиоканалу, один бит испортился. Один бит испортится в открытом тексте, и полностью испортится следующий блок. Но на следующий это никак не распространится. То есть этот режим является самовосстанавливающимся (ошибка не распространяется).

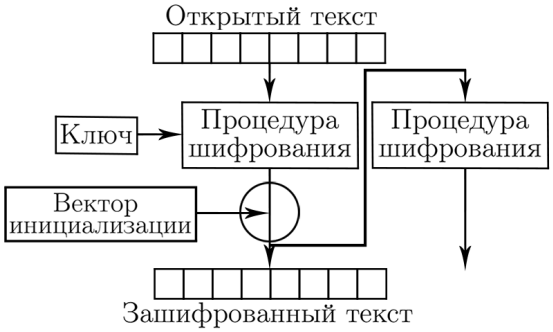

Рис. 3.8

Также возможно частичное распараллеливание. Можно шифровать параллельно несколько блоков, но конечный результат нужно сложить с предыдущим. Если раньше нельзя было начать шифрование следующего блока, пока на зашифруем предыдущий, потому что было необходимо сложить с результатом шифрования, то здесь можно предварительно зашифровать все нужные блоки, а потом сложить с результатами шифрования предыдущих.

4.4. Output Feedback (OFB)

Суть этого метода — обратная связь по выходу.

Здесь после шифрования результат шифрования подаётся на следующий вход, и только потом складывается (в CFB сначала происходило сложение и подача на следующий блок).

Недостатки: возможно, короткий цикл.

Преимущества: возможна предварительная генерация.

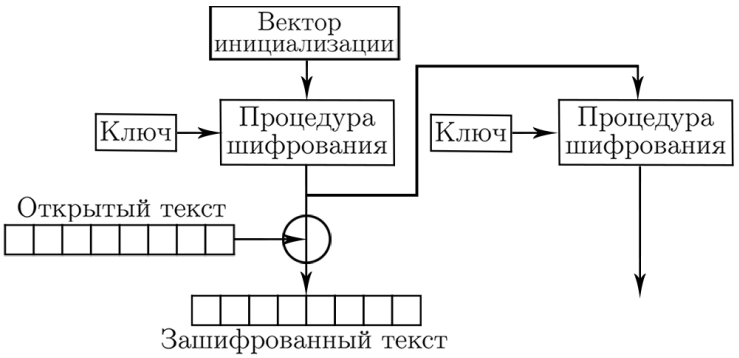

Рис. 3.9

Часть, связанная с шифровкой вектора инициализации, является самой вычислительно сложной, потому что здесь происходит всё шифрование. Именно этой частью загружен процессор. А потом сложение с открытым текстом происходит очень быстро, потому что это сложение по модулю 2, побитовое. Можно заранее сгенерировать битовый поток Г, который складывается с открытым текстом.

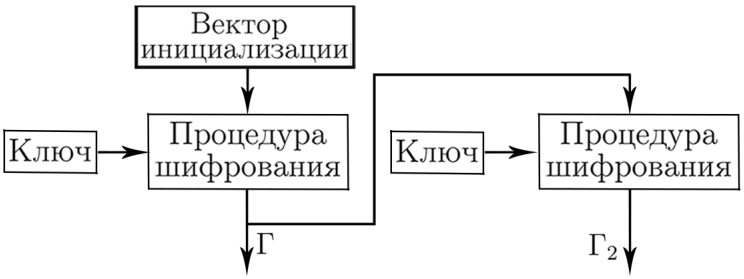

Рис. 3.10

Но, если всегда из Γ_2 получается Γ_3, то, если после Γ_3 получится Γ_1, то цикл будет состоять всего из 3-х итераций.

Если длина вектора 64 бита, то с вероятностью $\frac{1}{2^{64}}$ цикл прервётся. Можно грубо оценить длину такого цикла из соображений теории вероятности; его длина будет $\sqrt{2^{64}} = 2^{32}$. Длина этого цикла будет в 2 раза меньше, чем длина цикла на других режимах связи. Таким образом, возможность короткого цикла остается главным недостатком этого режима.

Но это устраняется улучшением — особым режимом сцепления блоков. На вход каждого блока, каждой функции шифрования будем подавать некий уникальный номер блока. И дальше шифруем и прибавляем открытый текст к результату шифрования . Благодаря этому получаем режим сцепления, называемый **счётчиком (Counter)**. Несмотря на то, что злоумышленник может знать, какие в шифровке цифры, т. к. он не знает ключа, он не знает, во что эти цифры будут зашифрованы. Очевидно, что одинаковый открытый текст даст разные блоки, потому что эти открытые тексты будут стоять на разных позициях, а значит, там будет разный результат шифрования и разные блоки на выходе.

Если злоумышленник проведёт атаку открытым текстом (в какой-то момент подставит второй блок, зная, что некий открытый текст соответствует некому закрытому

тексту), он сможет восстановить результат шифрования. Но функции шифрования DES (в предположении, что нет полного перебора) и GOST устроены таким образом, что даже, зная открытый текст и закрытый текст, нельзя получить ключ.

Хорошая функция шифрования можно назвать не только ту, в которой, не зная ключа, невозможно получить открытый текст, но также ту, в которой, не зная ключа, невозможно получить закрытый текст (нельзя зашифровать повторно). И еще ту, в которой, даже зная открытый и закрытый текст, нельзя получить ключ. Более того, чаще всего это намного сложнее.

Из преимуществ рассматриваемого режима можно отметить полную параллелизацию, т. к. результаты шифования предыдущих блоков не используются. Тем не менее, такая система устойчива к атаке по словарю и к атаке по открытому тексту.