Сеть Файстеля

В начале 1970-х годов, сознавая необходимость защиты уже электронной информации при передаче данных в сетях ЭВМ (особенно бизнес-транзакций, при осуществлении денежных переводов и передаче конфиденциальных финансовых данных), компания International Business Machines (она же известная во всем мире как IBM) приступила к выполнению собственной программы научных исследований, посвященных защите информации в электронных сетях, в том числе и криптографии. Так развитие одной передовой технологии повлекло за собой настоящую революцию в другой.

Поскольку в ряде университетов Соединенных Штатов (таких, как Станфордский университет и Массачусетский технологический институт) всегда существовал интерес к данной области исследования, IBM постаралась привлечь университетских специалистов к разработке методов защиты электронной информации. Это было отчасти вызвано и тем, что многие из специалистов этих университов активно сотрудничали с военными кругами и соответственно специалистами военной разведки — основными потребителями криптографических методов сокрытия и защиты информации. Поэтому университетские криптографы обладали несколько большими объемами информации о защите информации, нежели другие профессиональные математики и аналитики. Ведь военные специалисты в то время были единственными источниками достойной научной информации, посвященной криптографии, хорошо знавшими криптографию не только с теоретической, но и практической стороны.

Команду разработчиков фирмы IBM, приступившую к исследованию систем шифрования с симметричной схемой использования ключей, возглавил доктор Хорст Файстель, в то время уже ставший довольно известным криптографом.

Файстель до того момента уже успел тесно поработать с Клодом Шенноном в компании Bell Laboratories. Идеи Шеннона вдохновляли многих исследователей на оригинальные изобретения. Свидетельством тому является довольно большое количество патентов, зарегистрированных и принятых в середине 60-х годов Национальным патентным бюро США (United States Patent and Trademark Office). Файстель активно сотрудничал с Шенноном и не мог не заразиться его идеями. На его счету как минимум два патента и несколько революционных статей в области криптографии и криптоанализа.

Воплотить большую часть из своих идей в жизнь он смог с помощью возможностей, предоставленных ему компанией IBM, не жалевшей денег и специалистов на разработку новых методов защиты электронной информации. Передовая технология требовала инвестиций и отнюдь не гарантировала быстрой отдачи, требовалось время на разработку действительно эффективных средств защиты электронного документооборота — эффективной и стойкой к взлому шифросистемы и методов ее использования. Представители руководства IBM понимали это и в достаточной степени предоставили свободу разработчикам, поставив перед ними в общем-то нелегкую и нетривиальную задачу.

Надо признать, что результат оправдал ожидания. Им стала проведенная в исследовательской лаборатории IBM Watson Research Lab разработка новой ори-

гинальной архитектуры построения симметричных шифров на базе необратимых преобразований. Архитектура нового способа шифрования впоследствии была названа в классической литературе архитектурой Файстеля (на данный момент в русской и зарубежной криптографии существует более устоявшийся термин: сеть Файстеля или Feistel's network). Позднее, в соответствии с разработанными Хорстом принципами, был сконструирован шифр Люцифер (Lucifer) — первый серьезный блочный шифр, описание которого появилось в открытой литературе и вызвало новую волну интереса специалистов к криптографии в целом.

Построение сложных криптографически стойких, но обратимых преобразований представляет собой довольно трудоемкую задачу. Кроме того, практическая реализация обратимых преобразований обычно содержит неэффективные алгоритмы, что приличным образом сказывается на скорости шифрования. По этой причине Файстель решил не искать решение проблемы обратимого преобразования данных, а попытаться найти схему шифрования, в которой такие преобразования не участвовали бы вовсе.

Идея использования операции «исключающее ИЛИ» возникла из классических примеров систем шифрования, а именно из идеи использовать самый простой с технической точки зрения способ шифрования — гаммирование. Стойкость такого способа, как известно, зависит от свойств вырабатываемой гаммы. Следовательно, процесс выработки гаммы — двоичной последовательности, которую затем суммируют с открытым текстом, — является самым узким местом во всем способе.

Файстель разрешил проблему следующим образом. Изначально выбирается размер блока данных, который будет зашифрован за одну итерацию алгоритма шифрования. Обычно размер блока фиксирован и не изменяется во время работы алгоритма над открытым текстом. Выбрав достаточно большого размера блок данных, его делят, например, пополам и затем работают с каждой из половинок. Если размер левой половинки равен размеру правой, такую архитектуру называют классической или сбалансированной сетью Файстеля. Если же деление блока данных происходит не на равные части, то такой алгоритм называют разбалансированной сетью Файстеля.

Предложенная им схема шифрования легко может быть продемонстрирована с помощью схемы шифрования (рис. 4.1).

На изображенной схеме буквами L_i и R_i обозначены левая и правая половинки исходных данных на i-м шаге последовательного преобразования. Каждый такой целый шаг называют раундом шифрования. Функция гаммирования обозначена через F_i, поскольку на каждом раунде может быть использована своя собственная функция. Ключ также имеет индекс i, но уже в силу того, что исходный ключ k может быть преобразован некоторым образом (говорят, развернут) в последовательность итерационных ключей либо подключей, то есть ключей, которые используются непосредственно функцией гаммирования.

Как видно из схемы, сначала с помощью функции гаммирования вырабатывают гамма-последовательность, которая зависит от итерационного ключа k_i и правой половины данных. После этого левая половинка просто суммируется с полученной гаммой по модулю, например, 2. Затем левая и правая половинки меняют-

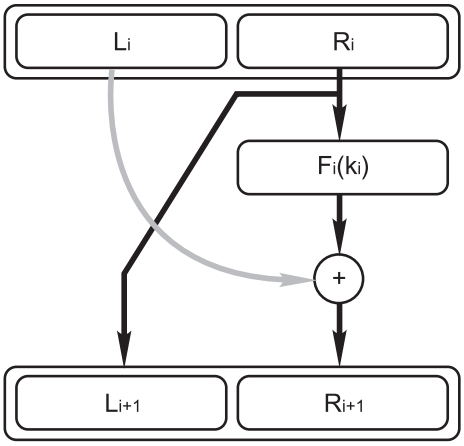

Рис. 4.1. Архитектура сети Файстеля

ся местами. На этом один цикл шифрования заканчивается. Поскольку за один раз обрабатывается только одна половина данных, желательно, чтобы число раундов было кратно двум. В таком случае есть уверенность, что каждая из половинок будет обработана одинаковое число раз. На схеме рис. 4.2 отображена сеть Файстеля с четным числом раундов.

Исходя из данного описания, преобразование данных одного раунда можно представить с помощью двух формул, выражающих новые значения левой и правой половинок блока шифруемых данных:

$$\begin{cases} L_{i+1} = R_i, \\ R_{i+1} = L_i \oplus F(R_i, k_i). \end{cases}$$

Архитектура разбалансированной сети Файстеля выглядит весьма похоже на архитектуру обычной сети и определяется таким же способом. Единственное, но весьма существенное отличие состоит в том, что, поскольку используется разбиение не на равные половинки блока, а на участки различной длины, функция гаммирования, обозначенная буквой «*F*», может зависеть не от всех битов исходного блока данных или иметь разные зависимости в разных раундах.

Собственно говоря, разбалансированную сеть Файстеля можно рассматривать как обобщение понятия сети Файстеля. Стойкость криптосистемы, построенной на основе сети Файстеля, зависит целиком от результата исполнения нелинейной функции гаммирования в нескольких итерациях. Поэтому для обеспечения достаточной надежности данные должны быть преобразованы с достаточно большим числом раундов, что позволяет достичь требуемых свойств рассеивания и полноты и соответственно стойкости шифра к дифференциальному и линейному криптоанализу (см. раздел «Современные методы криптоанализа»).

В большинстве шифров с архитектурой сети Файстеля используемая функция *F* в течение каждого раунда зависит только от одного из подключей, вырабатываемых из основного ключа шифра. Это свойство на самом деле не является основополагающим или положительно влияющим на стойкость шифра — в шиф-

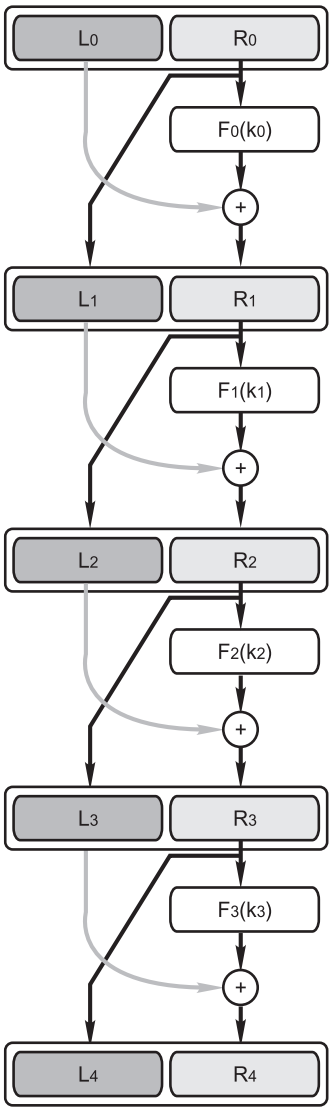

Рис. 4.2. Сеть Файстеля с несколькими раундами

ре Khufru, например, параметры функции F изменяются после зашифрования каждого следующего символа. Сеть с такого рода зависимостью функции гаммирования называют гетерогенной и гомогенной в противном случае. Применение гетерогенных сетей может значительно улучшить характеристики шифра, поскольку неравномерное изменение внутренних свойств сети в пределах допустимых границ делает изучение свойств шифра достаточно затруднительным занятием. Меняя размеры половинок и их влияние на выход шифра, можно добиться впечатляющих статистических результатов, поскольку существует очевидная зависимость не только между сложностью функции гаммирования, но и структурой используемой сети Файстеля.

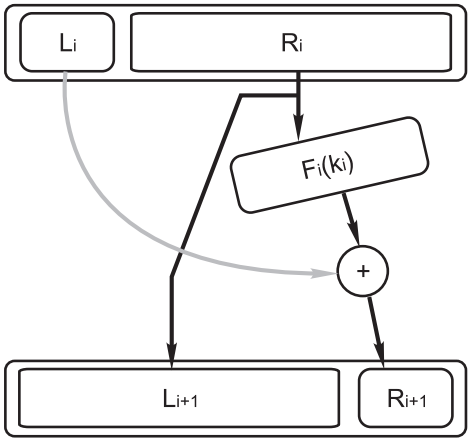

Рис. 4.3. Архитектура разбалансированной сети Файстеля

Целью построения блочных шифров является не только создание стойкого алгоритма защиты информации, но и такого, чтобы его реализация была достаточно дешевой, а время работы как можно более меньшим. Именно поэтому, легко реализуемые шифры на базе гомогенных сбалансированных сетей Файстеля применяются гораздо чаще и считаются своего рода «панацеей». Однако это вовсе не означает, что они являются более криптостойкими и надежными.

Естественным путем увеличения сложности анализа сетей Файстеля является следующий метод (кстати, использованный и в алгоритме в соответствии с ГОСТ 28147—89 — отечественным стандартом криптографического преобразования данных). Для того чтобы распространить влияние функции F в одном раунде на выход и функцию следующего раунда, к выходному значению F прибавляют по некоторому модулю значение итерационного подключа для текущего раунда и затем полученное значение подают на вход функции F следующего раунда шифрования. Такой способ организации сети ставит выходы последующих раундов шифров в прямую зависимость от предыдущих, что способствует организации лавинного эффекта и полноты.

Примером практической реализации сбалансированной гомогенной сети Файстеля может служить следующий исходный текст (см. листинг 4.3):

Листинг 4.3

```
uint16_t F_Gamma(uint16_t data_half, uin8_t key) {
    // используем какие-либо сложные преобразования
    // но желательно такие, чтобы их выполнение было как можно более быстрым
    return (data_half ^ ((uint16_t) key * 0xABCD1234);
}

uint32_t Feistel_Network(uint32_t data, uint8_t key, int rounds) {
uint16_t left, right, swap;

left = data & 0xFFFF;              // делим данные (размер 32 бита)
right = (data >> 16) & 0xFFFF;     // на половинки по 16 битов
```

```
for (int i = 0; i < rounds; i++) {
    swap = left ^ F_Gamma(right, key);    // готовим левую половинку
    left = right;                  // и меняем местами левую и правую
    right = swap;                  // половинки
}
return (left | ((uint32_t) right << 16));
}
```

Функция Feistel_Network() представляет собой реализацию сети Файстеля с произвольным числом раундов rounds. Ключ key представляет собой 8-битное значение, которое используется только как аргумент для передачи его функции гаммирования F_Gamma(). Обратите на это особое внимание, поскольку, очевидно, здесь таится первый способ оптимизации алгоритма.

Чтобы ускорить исполнение функции Feistel_Network(), мы можем внести тело функции F_Gamma() внутрь Feistel_Network(), устранив тем самым накладные расходы на передачу параметров и вызов подфункции. Другой очевидный способ ускорить процесс состоит в том, чтобы сделать число раундов постоянным (например, равным восьми или шестнадцати), а затем избавиться от цикла, расписав все итерации шифра последовательно, одну за другой. Выбор количества раундов объясняется нахождением некоторого своего рода компромисса между малой скоростью шифрования у шифра с большим количеством раундов и простотой вскрытия шифра с малым их числом. Этот показатель у шифра, построенного на сети Файстеля, колеблется от 8 до 32 раундов. Впрочем, никто не возбраняет применять и 64, и 128 раундов. Все упирается лишь в вычислительные мощности компьютерных систем. Единственное ограничение состоит в том, что количество раундов должно быть четным. В таком случае обе половинки — и левая, и правая — будут обработаны одинаковое число раз и, следовательно, не возникнет ситуации, когда одна половина данных (например, четная), зашифровываемых 16-раундовым алгоритмом, в действительности обработана восемью, а другая всего лишь семью итерациями алгоритма.

Все проделанные модификации для алгоритма с четырьмя раундами показаны в листинге 4.4. Подобная оптимизация обычно дает прирост производительности в 5—10 %, а ведь это еще оптимизация не алгоритма, как такового, а примитивная оптимизация его реализации.

Листинг 4.4

```
uint32_t Feistel_Network_Optimal(uint32_t data, uint8_t key, int rounds) {
    uint16_t left, right, swap;

    left = data & 0xFFFF;          // делим данные на половинки
    right = (data >> 16) & 0xFFFF;

    swap = left ^ right ^ ((uint16_t) key * 0xABCD1234); // 1-й раунд
    left = right;
    right = swap;

    swap = left ^ right ^ ((uint16_t) key * 0xABCD1234); // 2-й раунд
    left = right;
    right = swap;
```

```
    swap = left ^ right ^ ((uint16_t) key * 0xABCD1234); // 3-й раунд
    left = right;
    right = swap;

    swap = left ^ right ^ ((uint16_t) key * 0xABCD1234); // 4-й раунд
    left = right;
    right = swap;

    return (left | ((uint32_t) right << 16));
}
```

На сегодняшний день практически отсутствуют системы шифрования, построенные с помощью разбалансированных сетей Файстеля, а все популярные алгоритмы являются с этой точки зрения классическими. Их вид и функции шифрования можно успешно представить себе, исходя из предоставленных в листингах раздела примеров.