ПРАКТИЧЕСКАЯ КРИПТОЛОГИЯ
ЛЕКЦИЯ 5
Специальность: 6.170101 – Бсіт
Лектор: Сушко С.А.

БЛОЧНЫЕ ШИФРЫ

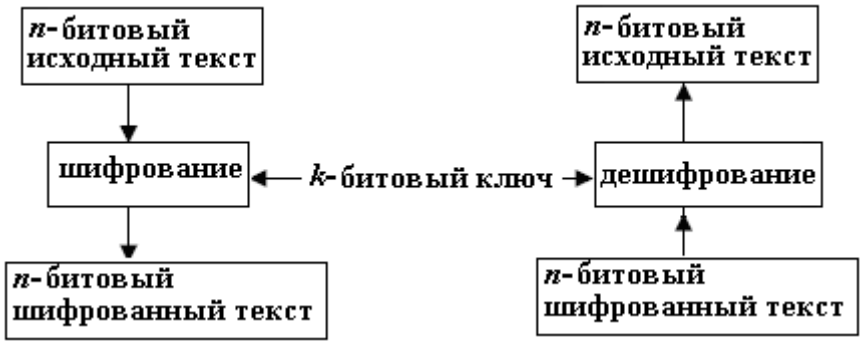
§1. ПРИНЦИПЫ ПОСТРОЕНИЯ БЛОЧНЫХ ШИФРОВ

Современный блочный шифр – это шифр с симметричным ключом, разбивающий перед шифрованием открытый текст на n-битовые блоки и далее шифрующий сообщение блоками, т.е.

$$y = E_k(x) \qquad \text{и} \qquad x = D_k(y),$$

где x, y – блоки открытого и шифрованного текстов; k – секретный ключ шифра; E, D – функции шифрования и дешифрования (другими словами, блоковый алгоритм – это алгоритм простой замены блоков текста фиксированной длины).

Алгоритмы дешифрования и шифрования – инверсные, оба работают на одном и том же секретном ключе.



Общая идея шифрования и дешифрования
с помощью блокового шифра

Если сообщение содержит менее n битов, то его дополняют дополнительными данными (чаще нулями) до n-битового размера; если текст имеет больше, чем n бит, то его делят на n-битовые блоки.

Чаще всего блочные шифры обрабатывают блоки длиной $n = 64, 128, 256, 512$ бит.

Пример. Сколько дополнительных бит надо добавить к сообщению длиной 100 символов, если кодирование одного символа требует 8 бит и блочный шифр работает с блоками длиной 64 бита?

Р е ш е н и е. Закодированное сообщение из 100 символов содержит 800 бит. Исходный текст должен делиться без остатка на 64. Обозначим L и l – длина сообщения и длина дополнения

$$L + l \equiv 0 (\mathrm{mod}\, 64) \Rightarrow l \equiv -800 (\mathrm{mod}\, 64) \equiv 32 (\mathrm{mod}\, 64).$$

$\Rightarrow$ в конце открытого текста надо добавить 32 дополнительных бита. Общая длина 832 бита или 13 блоков по 64 бита. Алгоритм шифрования будет работать 13 раз, чтобы создать 13 блоков шифрованного текста.

Блочный шифр можно спроектировать так, чтобы он действовал и как шифр подстановки, и как шифр перестановки:

1) если шифр спроектирован как шифр подстановки, каждый бит открытого текста может быть заменен на 0 или 1 $\Rightarrow$ исходный текст и шифротекст могут иметь различное число единиц.

2) если шифр спроектирован как шифр перестановки, то биты открытого текста только меняются местами.

В любом случае, число возможных n-битовых открытых текстов равно числу шифротекстов и равно 2^n (*так как каждый из n битов блока может быть равен 0 или 1*).

Современные блоковые шифры спроектированы как шифры подстановки, потому что использование только перестановки (сохранение числа единиц или нулей) делает шифр уязвимым к методу полного перебора (из-за сохранения числа единиц или нулей). Проиллюстрируем это примерами.

Пример. Пусть блочный шифр шифрует блок размером $n = 64$ бита. Зашифрованный текст содержит 10 единиц. Сколько проб при полном переборе должен выполнить криптоаналитик, чтобы получить открытый текст, соответствующий перехваченному шифротексту, если: а) шифр спроектирован как шифр подстановки; б) шифр спроектирован как шифр перестановки.

Р е ш е н и е. а) при подстановке криптоаналитик не знает, сколько единиц находится в открытом тексте и должен перепроверить все возможные 2^{64} блока, чтобы найти один, который имеет смысл. Если криптоаналитик проверял бы 1 миллиард блоков за секунду, то для полного перебора ему потребовалось бы время $t = \frac{2^{64}}{10^9}$ с.

$$\ln t = 64\ln 2 - 9\ln 10 = 23,46;$$

$$t \approx e^{23,5} \approx 13710347350\,\text{с} \approx 3808429\,\text{час} \approx 158684\,\text{дня} \approx 435\,\text{лет}.$$

б) при перестановке криптоаналитик знает, что в открытом тексте есть точно 10 единиц. Он проводит атаку полного перебора, используя только те 64-битовые блоки, которые имеют точно 10 единиц. Общее число таких блоков: $C_{64}^{10} \approx 151 \cdot 10^9$. Все их можно проверить за ≈ 151с < 3 мин.

§2. ОПЕРАТОРЫ, ИСПОЛЬЗУЕМЫЕ ДЛЯ ПОСТРОЕНИЯ БЛОЧНЫХ ШИФРОВ

Согласно работам Шеннона к современным блочным шифрам выдвигают следующие требования:

- ***рассеивание*** информации – распространение влияния одного знака открытого текста на почти все знаки шифротексты. Это затрудняет статистический анализ и не позволяет восстановить ключ по частям;
- ***перемешивание*** информации для усложнения зависимости между ключом и шифрованным текстом.

Схемы, решающие такие задачи, называют SP***-сетями*** (от первых букв англ. substitution – подстановка и permutation – перестановка).

Основные операторы блоковых шифров:

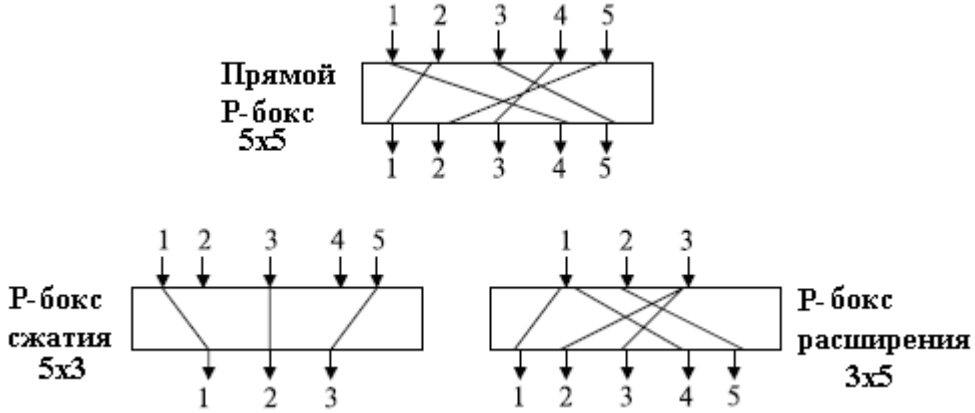
1) операторы перестановки, называемые P-боксами;
2) операторы подстановки, называемые S-боксами;
3) операция исключающего ИЛИ;
4) циклический сдвиг;
5) замена;
6) разбиение и объединение блока.

Комбинация этих операторов в современных шифрах позволяет рассеять и перемешать информацию.

1). P-боксы

P***-бокс*** (***блок перестановки***) подобен традиционному шифру перестановки (переставляемые символы – биты). Возможны 3 типа P -боксов:

- ***прямые*** P***-боксы*** (простая перестановка символов, n входов и n выходов, всего возможно $n!$ отображений);
- P***-боксы расширения***
- P***-боксы сжатия***



Три типа P-боксов

Прямой P-бокс обычно задают с помощью таблицы перестановок, в которой номера ячеек задают номера битов входящего блока, а числа записанные в ячейки указывают новые позиции бита. Например, по таблице перестановок прямого P-бокса 8×8, шифрующего 8-битовый блок, 7-ый бит становится 1-ым, 4-ый бит – 2-ым и.т.д.

7	4	8	1	3	5	2	6

Прямой P-бокс – обратимый, т.е. его можно использовать для шифрования и дешифрования. Этапы построения обратной перестановки следующие:

Домашнее задание: составить таблицу перестановки для прямого P-бокса 8×8, которая перемещает два средних бита (биты 4 и 5) во входном блоке к двум крайним битам (биты 1 и 8) выходного слова. Относительные позиции других битов не изменяются.

Ответ: [4 1 2 3 6 7 8 5].

P-бокс сжатия – это P-бокс с n входами и m выходами, где $m<n$. Некоторые из информационных входов блокированы и не связаны с выходом. Задаются таблицей перестановки с m ячейками (входы), в которых позиции битов от 1 до n, (биты, которые блокированы в таблице отсутствуют). Пример таблицы перестановки для P-бокса сжатия 8×4 (биты с номерами 2, 4, 6, 8 блокированы).

7	3	1	5

P-бокс расширения – это P-бокс с n входами и m выходами, где $m>n$. Некоторые из входов связаны больше чем с одним выходом. Правила перестановки бит указываются в таблице. Пример таблицы перестановки для P-бокса расширения 12×16. Биты с номерами 1, 3, 9 и 12 соединены с двумя выходами.

01	09	10	11	12	01	02	03	04	05	06	07	08	09	12	03

P-боксы сжатия и расширения необратимы.

P-бокс сжатия не является обратным к P-боксу расширения и наоборот. Это означает, что если мы используем P-бокс сжатия для шифрования, мы не сможем использовать P-бокс расширения для

дешифрования и наоборот. Однако, есть шифры, которые применяют P-боксы сжатия или расширения.

2). S-боксы

S-бокс (блок подстановки) – это миниатюрный шифр подстановки. На вход в S-бокс может подаваться n-битовый блок, а на выходе выйти уже m- битовый блок, где не всегда $m = n$. (говорят бокс с n входами и m выходами).

Обозначим в S-боксе с n входами и m выходами входы через $x_1, x_2, ..., x_n$, а выходы – через $y_1, y_2, ..., y_m$. Связь между входами и выходами задает система уравнений

$$\begin{cases} y_1 = f_1(x_1, x_2, ..., x_n), \\ y_2 = f_2(x_1, x_2, ..., x_n), \\ ..... \\ y_m = f_m(x_1, x_2, ..., x_n) \end{cases}$$

S-боксы делятся на линейные и нелинейные. В ***линейном S-боксе*** эту связь можно записать в виде линейных соотношений

$$\begin{cases} y_1 = a_{11}x_1 \oplus a_{12}x_2 \oplus ... \oplus a_{1n}x_n, \\ y_2 = a_{21}x_1 \oplus a_{22}x_2 \oplus ... \oplus a_{2n}x_n, \\ ..... \\ y_m = a_{m1}x_1 \oplus a_{m2}x_2 \oplus ... \oplus a_{mn}x_n \end{cases}$$

В ***нелинейном S-боксе*** линейные соотношения для каждого выхода задать нельзя.

Пример. В S-боксе с тремя входами и двумя выходами мы имеем

$$\begin{cases} y_1 = x_1 \oplus x_2 \oplus x_3, \\ y_2 = x_1 \end{cases}$$

S-бокс линеен, может быть представлен в матричном виде:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

Пример. В S-боксе с тремя входами и двумя выходами мы имеем

$$\begin{cases} y_1 = x_1^3 \oplus x_2, \\ y_2 = x_1 \oplus x_1x_2 \oplus x_3 \end{cases}$$

(умножение и сложение проводится в GF(2)). S-бокс нелинеен, так как нет линейных соотношений между входами и выходами.

Пример S-бокса размера 3×2. Первый бит входа определяет строку, два следующих бита входа определяют столбец. Два бита на выходе – это значение на пересечении выбранных строки и столбца.

Самый левый бит входа ↓	00	01	10	11	← Самые правые биты входа
0	00	10	01	11	
1	10	00	11	01	

В обратимом S-боксе число входных битов должно быть равным числу бит выхода.

3. Операция XOR (операция исключающее ИЛИ)

Операция XOR – побитовое сложение, сложения по модулю 2.

Свойства операции XOR:

1. Замкнутость: если x, y – n-битовые слова, то $x \oplus y = z$, где z – n-битовое слово.
2. Ассоциативность: $x \oplus (y \oplus z) = (x \oplus y) \oplus z$.
3. Коммутативность: $x \oplus y = y \oplus x$.
4. Существование нулевого элемента с условием $x \oplus (00...0) = x$.
5. Существование инверсии – операция XOR слова с самим собой дает нулевой элемент $x \oplus x = (00...0)$.

Тесно связаны с XOR две операции:

- операция дополнения – инвертирование каждого бита в блоке: 0 на 1, а 1 на 0. Если $\overline{x}$ – дополнение x, то $x \oplus \overline{x} = (11...1)$ и $x \oplus (11...1) = \overline{x}$.
- операция инверсии, по которой из $y = x \oplus k$ следует $x = y \oplus k$, где k – ключ.

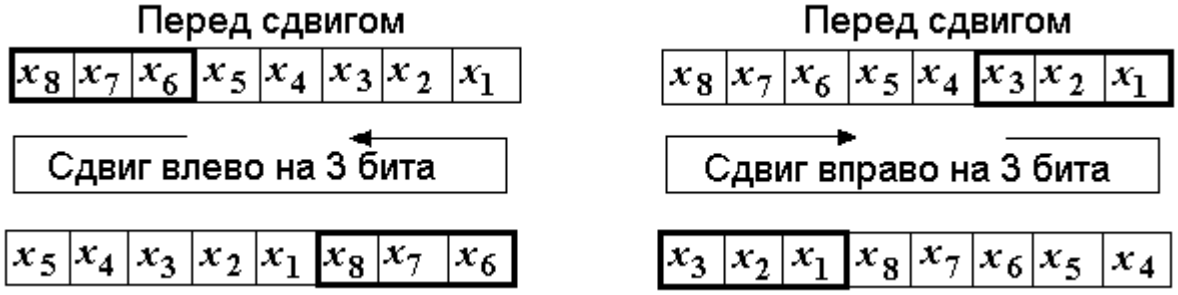
5. Циклический сдвиг

Циклический правый сдвиг сдвигает каждый бит в n-битовом слове на k позиций вправо; самые правые k бит справа удаляются и становятся крайними левыми. Аналогично работает левый сдвиг.

Циклический левый сдвиг – результат инверсии правого сдвига. Если один из них используется для шифрования, другой может применяться для дешифрования.

Свойства циклического сдвига:

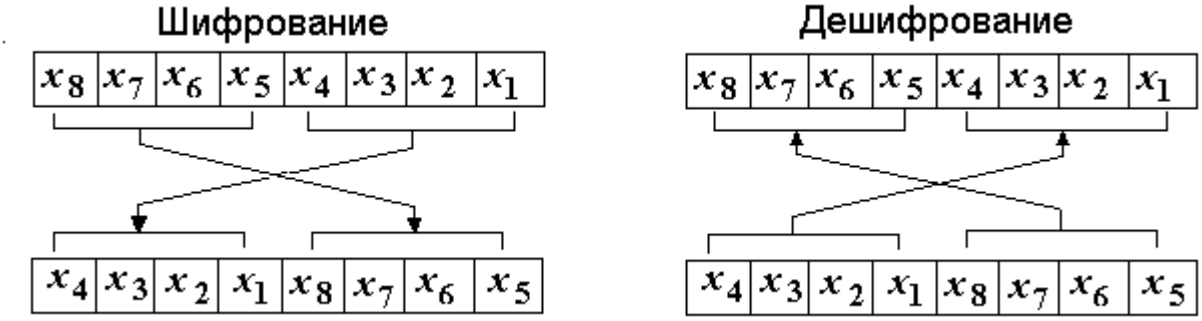
1) смещение по модулю n. Другими словами, если $k = 0$ или $k = n$, никакого смещения не происходит. Если $k > n$, то входная информация сдвигается на $k \bmod n$ бит.
2) если смещение делается неоднократно, то вновь может появиться исходное n-битовое слово (сдвиг является групповой операцией).



Циклический сдвига 8- битового слова

6. Замена

Замена – частный случай операции циклического сдвига на $k = n/2$ битов (здесь n – четное). Сдвиг влево на $n/2$ – то же самое, что сдвиг на $n/2$ вправо, $\Rightarrow$ операция является обратимой.



Операция замена в 8 битовом а слове

6. Разбиение и объединение

Разбиение разделяет n-битовое слово пополам, создавая два слова равной длины. ***Объединение*** связывает два слова равной длины, чтобы создать n-битовое слово. Эти две операции инверсны друг другу: если одна используется для шифрования, то другая – для дешифрования.

Комплекс, который объединяет подстановку, перестановку и другие рассмотренные операторы, называется ***составным***.

§3. РАУНДЫ БЛОЧНЫХ ШИФРОВ

Современные блочные шифры рассеивают и перемешивают информацию, обрабатывая данные с помощью комбинации P-боксов, S-боксов и других операторов.

Однократное применение такой комбинации называется ***раундом*** (***циклом***) блокового шифра. В N-раундном шифре, чтобы создать зашифрованный текст, исходный текст шифруется N раз; соответственно, зашифрованный текст расшифровывается N раз. Текст, возникающий между двумя раундами, называется ***промежуточным.***

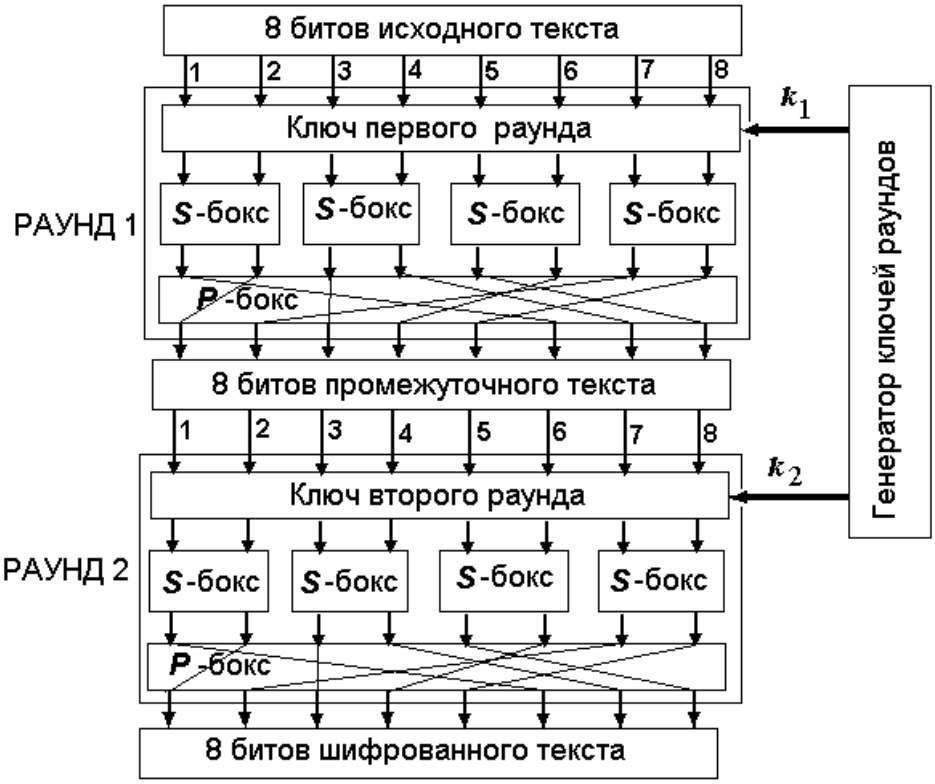
Чтобы улучшить рассеивание и перемешивание информации, в современных блоковых шифрах используют крупные блоки данных, и большое число S-боксов и большое число раундов.

Для каждого раунда из основного секретного ключа k шифра с помощью ***алгоритма разворачивания ключа*** генерируется ***подключ раунда*** k_i ($i-$номер раунда). Алгоритмы для генерации раундовых подключей называются еще ***ключевым расписанием***. Поскольку при расшифровании подключи используют в обратном порядке по сравнению с шифрованием, то шифротекст расшифровывается успешно, только если алгоритм построения подключей обратимый.

Пример составного шифра с 2 раундами (см.рис.).

В каждом раунде выполнены 3 преобразования:

- 8-битовый открытый текст смешивается с 8-битовым ключом с помощью операции XOR, чтобы скрыть биты (говорят "отбелить" текст (whiting)).
- блок разбивается на 4 группы по 2 бита, которые подаются в четыре S-бокса. Значения битов изменяются.
- Выходы S-бокса поступают в P-бокс, где биты переставлены так, чтобы в следующем раунде результат каждого блока поступил на разные входы.

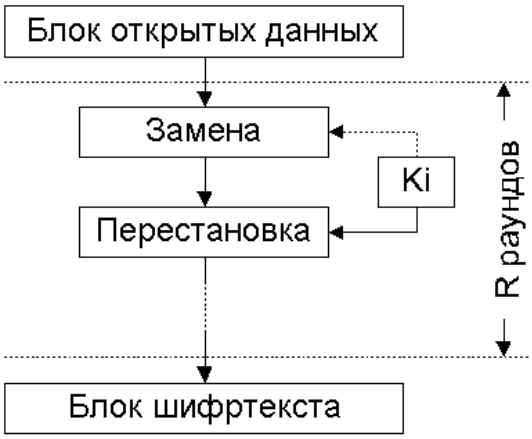


§4. ОБЩИЕ ПРИНЦИПЫ ПОСТРОЕНИЯ БЛОКОВЫХ ШИФРОВ

По виду повторяющегося преобразования, используемого в раундах, блоковые алгоритмы шифрования делят на несколько категорий:

1. ***Сеть Фейстеля***, использующая многократно повторяющуюся структуру, называемую ***ячейкой Фейстеля***. Допускается использование необратимых операторов. При переходе от одной ячейки к другой меняется раундовый ключ. Операции зашифрования/расшифрования при определённой доработке совпадают, требуя только использования ключей в обратном порядке. Шифрование при помощи данной конструкции легко реализуется как на программном уровне, так и на аппаратном, что обеспечивает широкие возможности применения (например, DES, ГОСТ 28147-89, RC5, BLOWFISH, TEA, CAST-128 и т.д.).
2. ***Подстановочно-перестановочная сеть*** (***SP-сеть*** - Substitution-permutation network). В отличие от сети Фейстеля, SP-сети обрабатывают за один раунд целиком шифруемый блок. Обработка данных сводится, в основном, к заменам и перестановкам, зависящим от ключа. Упрощенная схема показана на рисунке. Впрочем, такие операции характерны и для других видов алгоритмов шифрования, поэтому, на мой взгляд, название "подстановочно-перестановочная сеть" является

достаточно условным. SP-сети распространены существенно реже, чем сети Фейстеля; в качестве примера SP-сетей можно привести алгоритмы SERPENT или SAFER+ .



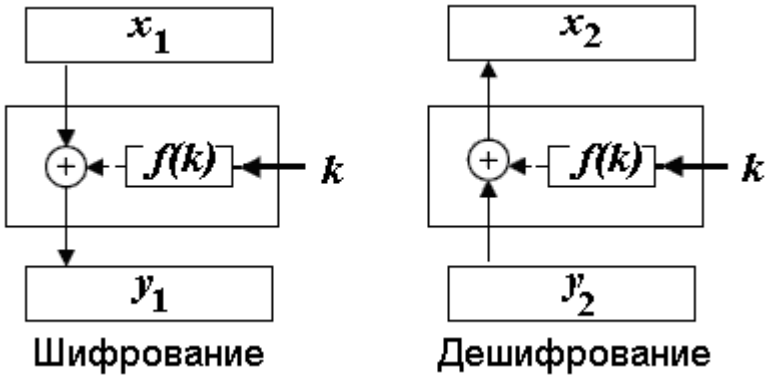
3. ***Шифры со структурой квадрат*** основанные только на обратимых операторах. Пример – шифр AES. Для структуры "квадрат" характерно представление шифруемого блока данных в виде двумерного байтового массива. Криптографические преобразования могут выполняться над отдельными байтами массива, а также над его строками или столбцами. Структура алгоритма получила свое название от алгоритма Square, который был разработан в 1996 году Винсентом Риджменом (Vincent Rijmen) и Джоан Деймен (Joan Daemen) – будущими авторами алгоритма Rijndael, также имеющего Square-подобную структуру и ставшего новым стандартом шифрования США AES после победы на открытом конкурсе. Другой пример – алгоритм SHARK (более ранняя разработка Риджмена и Деймен) и CRYPTON, разработанный южнокорейским криптологом Че Хун Лим из Южной Кореи в 1998 г.

Строгие границы между описанными выше структурами не определены, поэтому достаточно часто встречаются алгоритмы, причисляемые различными экспертами к разным типам структур. Например, алгоритм CAST-256 относится его автором к SP-сети, а многими экспертами называется расширенной сетью Фейстеля.

§5. СЕТЬ ФЕЙСТЕЛЯ

Шифр (сеть) Фейстеля базируется на необратимых операторах и использует один и тот же модуль при расшифровании и зашифровании.

Возникает вопрос: если алгоритмы шифрования и расшифрования содержат необратимые операторы, то как зашифрованный текст инвертировать в открытый? Фейстель показал, что действие необратимого оператора в алгоритме шифрования можно нейтрализовать в алгоритме расшифрования с помощью операции XOR.



Покажем это. Пусть при зашифровании ключ поступает на вход необратимой функции $f(k)$, значения которой далее складываются по модулю 2 (операция XOR) с исходным текстом. В результате этих действий появлется шифротекст. Пусть открытому тексту x_1 отвечает шифрованный текст y_1, и при любом изменении в ходе зашифрования возникает шифрованный текст y_2, который при расшифровании дает открытый текст x_2. Надо показать, что из равенства $y_2 = y_1$, следует равенство $x_1 = x_2$.

Зашифрование: $y_1 = x_1 \oplus f(k)$.

Расшифрование:

$$x_2 = y_2 \oplus f(k) \underset{\uparrow y_2 = y_1}{=} y_1 \oplus f(k) \underset{\uparrow y_1 = x_1 \oplus f(k)}{=} x_1 \oplus f(k) \oplus f(k) = x_1$$

Следовательно, хотя в шифре использована необратимая функция, алгоритмы зашифрования и расшифрования могут быть инверсными.

Пример. Открытый текст и его шифротекст имеют длину 4 бита, длина ключа 3 бита. Функция извлекает 1-ый и 3-ий биты ключа, интерпретирует их как десятичное число, возводит его в квадрат и интерпретирует результат как 4-битовую последовательность. Найти результат зашифрования и расшифрования, если первоначальный исходный текст – 0111, ключ –101.

Р е ш е н и е: первый и третий биты ключа – это $11_2 = 3_{10}$; $3^2 = 9_{10} = 1001_2$ (индекс указывает систему исчисления).

Зашифрование: $y = x \oplus f(k) = 0111 \oplus 1001 = 1110$.

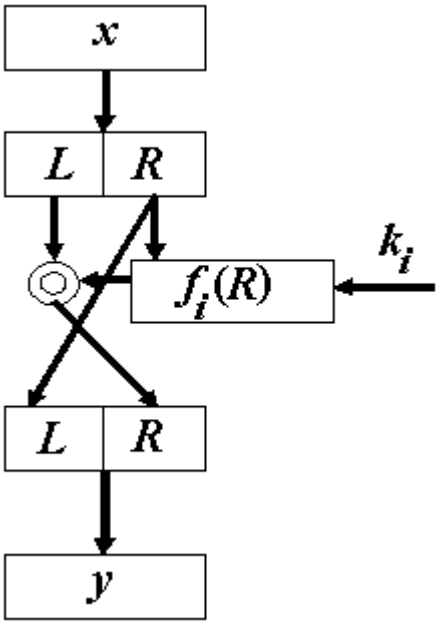
Расшифрование: $x = y \oplus f(k) = 1110 \oplus 1001 = 0111$

Функция $f(k)$ – необратима, но операция XOR позволяет использовать ее и в алгоритмах зашифрования, и в алгоритмах расшифрования.

Структура i-го раунда шифрования блока сети Фейстеля

Блок текста x длиной N разобьем на два полублока: L (левый старший) и R (правый младший):

$$x = \{L; R\}, \quad |x| = N, \quad |L| = |R| = N/2.$$



На вход раундовой функции подадим правый полублок R и раундовий подключ k_i и применим к ним операцию XOR. Для подготовки к следующему $(i+1)$-му раунду меняем местами левый и правый полублоки местами (говорят образуется ***петля Фейстеля***). Процес преобразования i-го раунда сети Фейстеля выглядит так:

$$\begin{cases} L_i = R_{i-1}; \\ R_i = L_{i-1} \oplus f(R_{i-1}; k_i). \end{cases}$$

Так как

$$L_{i-1} \oplus f(R_{i-1}; k_i) \oplus f(R_{i-1}; k_i) = L_{i-1},$$

то сеть Фейстеля гарантирована обратима с учетом возможности восстановления исходных данных в каждом раунде.

Свойства сети Фейстеля:

1^0. Сеть Фейстеля можно сконструировать так, что для зашифрования и расшифрования будет использоваться один и тот же алгоритм – отличие между этими операциями будет состоять лишь в порядке применения раундовых ключей; такое свойство алгоритма наиболее полезно при его аппаратной реализации или на платформах с ограниченными ресурсами.

2^0. В одном раунде меняется только половина битов блока $\Rightarrow$ для стойкости шифра надо увеличивать число раундов.

Замечание: Подстановочно-перестановочные сети используют лишь обратимые операторы: S-боксы должны иметь равное число входов и выходов, чтобы быть совместимыми. Не позволяется сжатие или расширение P-боксов, нет необходимости делить исходный текст на две половины.

Для получения криптопреобразования, обладающего хорошими криптографическими свойствами, функция усложнения f, используемая в раунде, реализуется в виде композиции элементарных преобразований, называемых слоями функции усложнения (функцию f называют еще ***раундовой***, ***цикловой***). Конструктивные слои функции

усложнения имеют следующие назначения: подмешивание раундовых ключей; перемешивание входных блоков; реализацию сложной нелинейной зависимости между знаками ключа, входного и выходного блоков.

Раундовая функция должна удовлетворять ряду условий:

- цикловая функция должна быть обратимой (однако помнить, что в схеме Фейстеля функция усложнения в принципе может не удовлетворять этому требованию, так как обратимость преобразования обеспечивается за счет использования операции XOR);
- раундовая функция должна быть нелинейной;
- перемешивающие слои раундовой функции должны реализовывать связи между входными и выходными битами S-боксов таким образом, чтобы каждый S-бокс удовлетворял критериям лавинного эффекта (см. лекцию по DES), а совокупность входных битов каждого S-бокса зависела от выходов нескольких S-боксов предыдущего раунда;
- раундовая функция должна обладать свойствами, затрудняющими применение методов дифференциального и линейного криптоанализа, т.е. раундовая функция должна иметь минимальную корреляцию между разностью открытых текстов и соответствующих криптограмм.