КОНТРОЛЬ ЦЕЛОСТНОСТИ

Лекция 16

Контроль целостности

- Процесс контроля целостности обеспечивается введением в передаваемую информацию избыточности. Это достигается добавлением к сообщению некоторой проверочной комбинации. Такая комбинация вычисляется согласно определенным алгоритмам и играет роль индикатора, с помощью которого проверяется целостность сообщения. Именно этот момент дает возможность проверить, были ли изменены данные третьей стороной. Вероятность того, что данные были изменены, служит мерой имитостойкости шифра.
- Дополнительную избыточную информацию, вносимую в сообщение, называют имитовставкой. Вырабатываться имитовставка может как до начала, так и одновременно с шифрованием сообщения.

Криптографические примитивы

Основные свойства сообщения, обеспечиваемые криптографической защитой:

конфиденциальность – невозможность ознакомления с сообщением посторонних лиц;

аутентичность – включает в себя **аутентичность отправителя**, **целостность сообщения** и **невозможность отрицания авторства**.

Проверка целостности сообщения гарантирует невозможность его модификации из-за подмены злоумышленником или случайного искажения.

Защитить передаваемую информацию можно путём добавления к ней некоторого контрольного поля - контрольной суммы. В зависимости от того, на основе какой информации и по каким правилам вырабатывается эта сумма, различают алгоритмы вычисления хеш-функций, имитовставки и электронной цифровой подписи (ЭЦП).

Хеш-функции

Хеш-функция – это преобразование, отображающее множество битовых строк произвольной длины на множество битовых строк фиксированной длины.

Криптографически стойкой называют хеш-функцию, удовлетворяющую следующим свойствам:

- вычислительно сложно найти текст, который при хешировании выдаст заданный хеш;
- вычислительно сложно найти коллизию – т.е. пару исходных текстов, которые дадут одинаковое значение хеша.

Под словосочетанием «вычислительно сложно» подразумевается, что для выполнения вычислений за разумное время необходима вычислительная мощность и/или объём оперативной памяти, превосходящий реально доступный на данном уровне развития вычислительной техники.

Недостатком использования хеш-функции (бесключевой) для защиты информации является тот факт, что значение хеша зависит только от передаваемого сообщения. То есть, злоумышленник может изменить сообщение, и затем заново вычислить значение хеша, введя принимающую сторону в заблуждение.

Схема вычисления ХЭШ-функции

Схема вычисления значения h(M) хэш-функции h для сообщения M обычно включает в себя:

- алгоритм вычисления шаговой функции хэширования g;
- итеративную процедуру вычисления хэш-функции И.

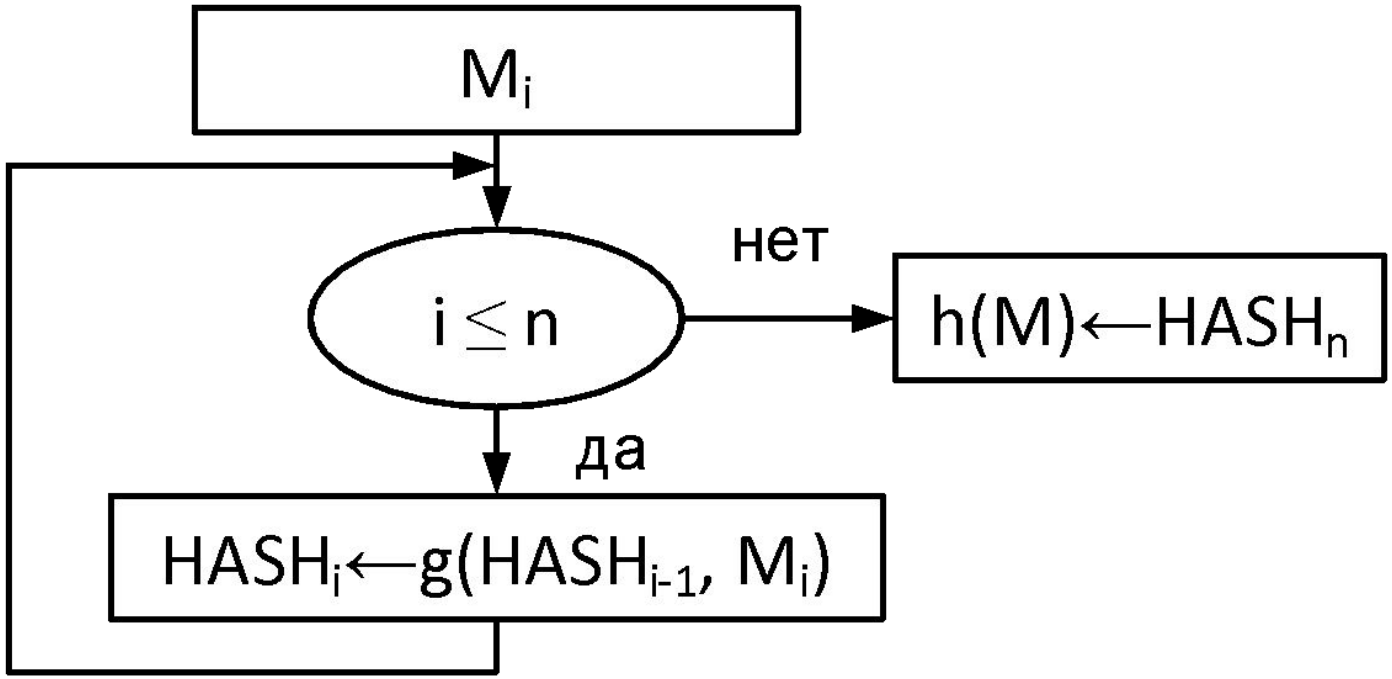


Схема вычисления значения хэш-функции для сообщения $M = M_1 M_2 ... Mn$.

Схема алгоритма SHA-1 (1)

Хэш –функцией в криптографии называется преобразование информации, переводящее строку битов произвольной длины в строку битов фиксированной длины.

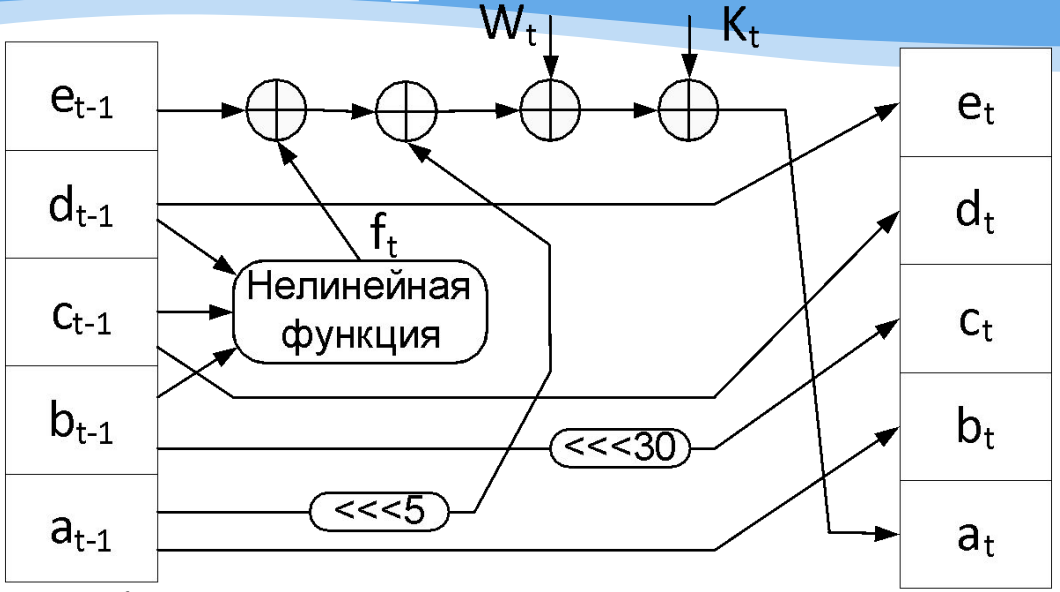
Известные хэш-функции: MD2, MD4 и MD5, SHA, SHS, ГОСТ Р 34.11-94 и другие.

Secure Hash Algorithm SHA-1

Прежде всего исходное сообщение M дополняют так, чтобы оно стало кратным 512 битам. Дополнительная набивка выполняется следующим образом: сначала добавляется 1, затем следует столько нулей, сколько необходимо для получения сообщения, которое на 64 бита короче, чем кратное 512, и, наконец, добавляют 64-битовое представление длины исходного сообщения. Инициализируются пять 32-битных переменных в виде:

A = 0 x 6 7 4 5 2 3 0 1
B = 0 x E F C D A B 8 9
C = 0 x 9 8 B A D C F E
D = 0 x 1 0 3 2 5 4 7 6
E = 0 x C 3 D 2 E 1 F 0

Схема алгоритма SHA-1 (2)



Алгоритм имеет следующий набор нелинейных функций:

$$f_t(X,Y,Z) = (X \wedge Y) \vee ((\neg X) \wedge Z) \quad t = 0 \ldots 19,$$

$$f_t(X,Y,Z) = X \oplus Y \oplus Z \quad t = 20 \ldots 39,$$

$$f_t(X,Y,Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z) \quad t = 40 \ldots 59,$$

$$f_t(X,Y,Z) = X \oplus Y \oplus Z \quad t = 60 \ldots 79$$

где t – номер операции.

В алгоритме используются четыре константы:

$$K_t = 0\text{x}\ 5A827999 \quad t = 0 \ldots 19,$$

$$K_t = 0\text{x}\ 6ED9EBA1 \quad t = 20 \ldots 39,$$

$$K_t = 0\text{x}\ 8F1BBCDC \quad t = 40 \ldots 59,$$

$$K_t = 0\text{x}\ CA62C1D6 \quad t = 60 \ldots 79$$

Схема алгоритма SHA-1 (3)

Блок сообщения преобразуется из шестнадцати 32-битовых слов (M0….M15) в восемьдесят 32-битных слов(W0…W79) с помощью следующего алгоритма:

Wt =Mt для t = 0…15

$W_t = (W_{t-3}\ W_{t-8}\ W_{t-14}\ W_{t-16}) << 1$ для t = 16…79 (*)

где t – номер операции (для t = 1…80); <<< S – циклический сдвиг влево на S бит.

A, B, C, D, E присваиваются a, b, c, d, e. Работает схема в течении главного цикла, во время которого вычисляется текущее значение хэш-функции. Этот главный цикл состоит из четырёх циклов, а каждый цикл из 20 микроциклов, т.е. всего используется 80 микроциклов для вычисления текущего значения хэш-функции. При этом каждый цикл функции ft имеет указанное значение, значение kt определено. Информация блока составляет 512 бит разбиваемых на шестнадцать 32-битных слов Mt. В первых 16 микроциклах берутся эти значения Mt, а в последних – по формуле (*). Полученные значения at, bt, ct, dt, et, суммируются с A, B, C, D, E и становятся A, B, C, D, E и т. д. В итоге получаем 160 битовое значение хэш-функции.

С учетом введенных обозначений главный цикл из восьмидесяти операций программно можно описать так:

```
FOR t = 0 до 79
TEMP = (a<<<5)+ft(b,c,d)+e+Wt+Kt
E=d
D=c
C=(b<<<30)
B=a
A=TEMP
```

7 алгоритмов семейства хэш-функций MD 4 (1)

Все представленные 7 алгоритмов – обобщение более раннего и простого алгоритма MD4.

1. MD4 – 3 раунда по 16 шагов в каждом, выходная строка – 128 бит.
2. MD5 – 4 раунда по 16 шагов в каждом, выходная строка – 128 бит.
3. SHA-1 – 4 раунда по 20 шагов в каждом, выходная строка – 160 бит.
4. RIPEMD – 160 – 5 раундов по 16 шагов в каждом, выходная строка – 160 бит.

7 алгоритмов семейства хэш-функций MD 4 (2)

5-7 алгоритмы семейства MD 4 относятся к SHA-2

5. SHA-256 – 64 раунда по одному шагу, выходная строка – 256 бит.
6. SHA-512 – 80 раундов по одному шагу, выходная строка – 512 бит.
7. SHA-384 – 80 раундов по одному шагу, выходная строка – урезана до 384 бит.

ГОСТ Р34.11-94

Параметрами хэш-функции ГОСТ Р34.11-94 являются 256- битный стартовый, вектор хэширования *GOST0 = gost0\\gost1\\gost2\\gost3*, на выбор которого ограничения не накладываются, и блок подстановки, используемый в алгоритме шифрования ГОСТ 28147-89.

Вычисление хэш-функции h:

Вход: M=M_1 M_2...Mn (n блоков по 256 бит)

Алгоритм: для всех i=0, …, n $GOST_i \leftarrow g(GOST_{i-1}, M_i)$

Выход: h(M)= $GOST_n$

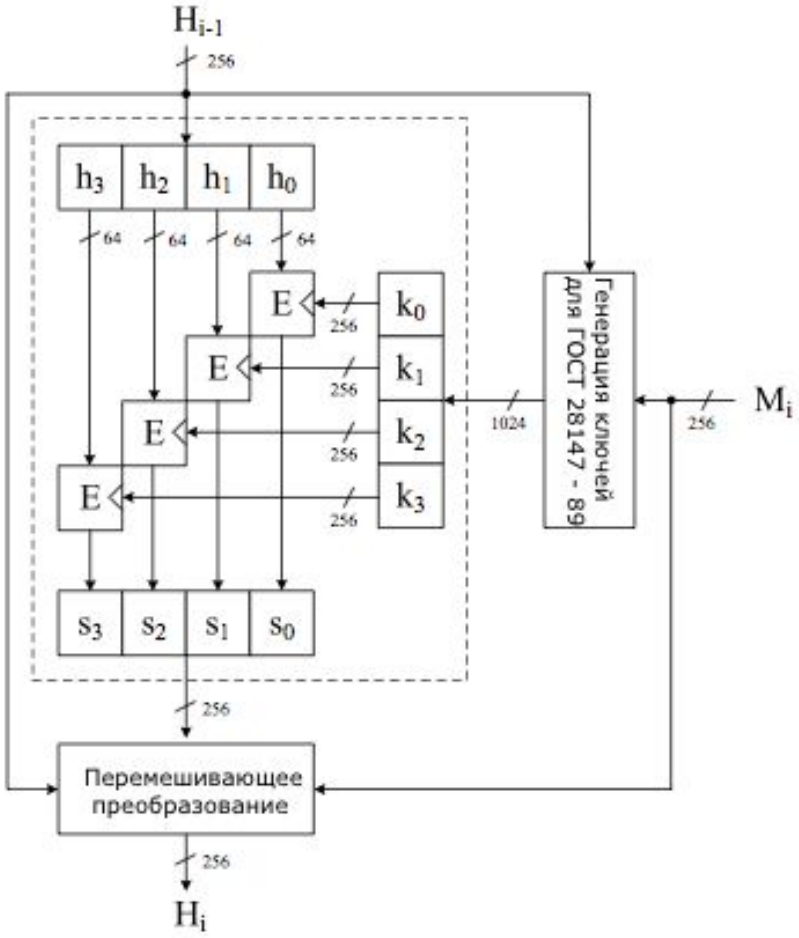
При вычислении шаговой функции хэширования используется накопитель *H,* содержащий четыре 64-битных слова H_0,..., H_3. Исходным заполнением накопителя H являются слова стартового вектора хэширования $GOST_0$.

Алгоритм вычисления шаговой функции хэширования *g* для *256-битного* блока M_i включает в себя три этапа:

- генерацию четырех 256-битных ключей

$$K_1^{(i)}, K_2^{(i)}, K_3^{(i)}, K_4^{(i)}.$$

Структура функции сжатия ГОСТ Р34.11 – 94



ГОСТ Р34.11-94 (2)

- зашифрование заполнения накопителя H на этих ключах с помощью алгоритма ГОСТ 28147-89;
- перемешивание результата зашифрования.

На первом этапе блок M_i,- рассматривается как вектор m над полем F_2. С помощью четырех различных невырожденных аффинных над F_2 преобразований из этого вектора вырабатываются четыре 256-битных ключа

$$K_1^{(i)}, K_2^{(i)}, K_3^{(i)}, K_4^{(i)} : K_j^{(i)} = A_j m + c_j, j = 1, ..., 4.$$

где A_j - блочная матрица, c_j - вектор сдвига.

На втором этапе каждое из четырех 64-битных слов H_0, H_1, H_2, H_3 накопителя H зашифровывается в режиме простой замены на соответствующем ключе $K^{(I)}_J$. Результатом зашифрования является вектор s = (s_1, s_2, s_3, s_4), где:

$$s_j = E_{K_j^{(i)}}(H_j), \quad 1 \le j \le 4.$$

На третьем этапе выполняется перемешивание, представляющее собой композицию невырожденных линейных над F_2 преобразований, применяемую к векторам h, m, s, где h - представленное в виде вектора содержимое накопителя H. Результат перемешивания заносится в накопитель H и является текущим значением $GOST_i$ шаговой функции хэширования g для 256-битного блока M.

После обработки блока M_n итоговым сжатым образом сообщения будет 256-битная строка из четырех слов.

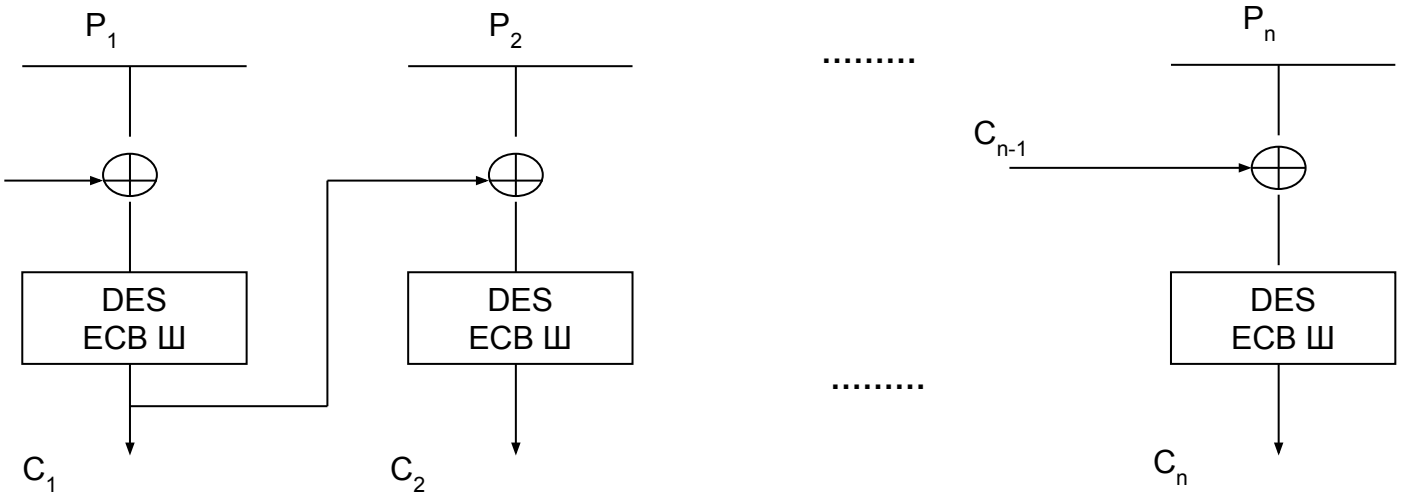
Алгоритмы выработки имитовставки

Для защиты от атак подмены сообщения и вычисления нового значения хеша применяются алгоритмы выработки имитовставки. В англоязычной литературе такие алгоритмы называют **MAC** – Message Authenfication Code – **код аутентификации сообщения**. Такие алгоритмы похожи на алгоритмы хеширования, с той разницей, что имитовставка зависит не только от исходного текста, но и от ключа. Злоумышленник, не знающий ключа, не сможет рассчитать имитовставку для изменённого сообщения. Таким образом, **имитовставка** может обеспечить все необходимые свойства сообщений – аутентификацию отправителя, целостность, невозможность отрицания авторства.

К недостаткам имитовставки можно отнести использование одного и того же ключа для выработки и для проверки имитовставки, что порождает проблемы распределения ключей, присущие всем методам симметричной криптографии.

Режим сцепления блоков

Зашифрование



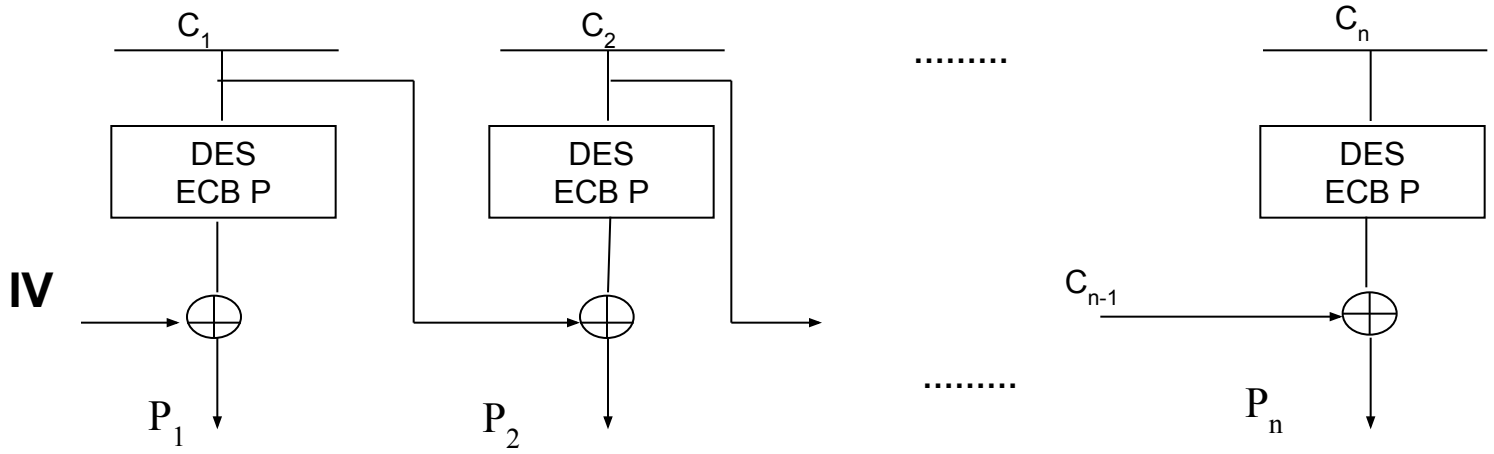
В режиме сцепления блоков CBC (Cipfer Block Chaining) шифротекст для каждого блока Ci получается путем зашифрования по базовому алгоритму, например, DES ECB, суммы по mod 2 соответствующего блока открытого текста и шифротекста предыдущего блока.IV – начальный вектор. DES ECB Ш – шифрование в режиме ECB.Для получения шифротекста первого блока используется начальный вектор IV вместо шифротекста предыдущего блока.

В канал связи для расшифрования направляются блоки шифротекста C1…..Cn и инициализирующий вектор IV.

Последний блок зависит от ключа, вектора IV и каждого бита открытого текста. Этот блок называется кодом аутентификации сообщения.

Режим сцепления блоков

Расшифрование



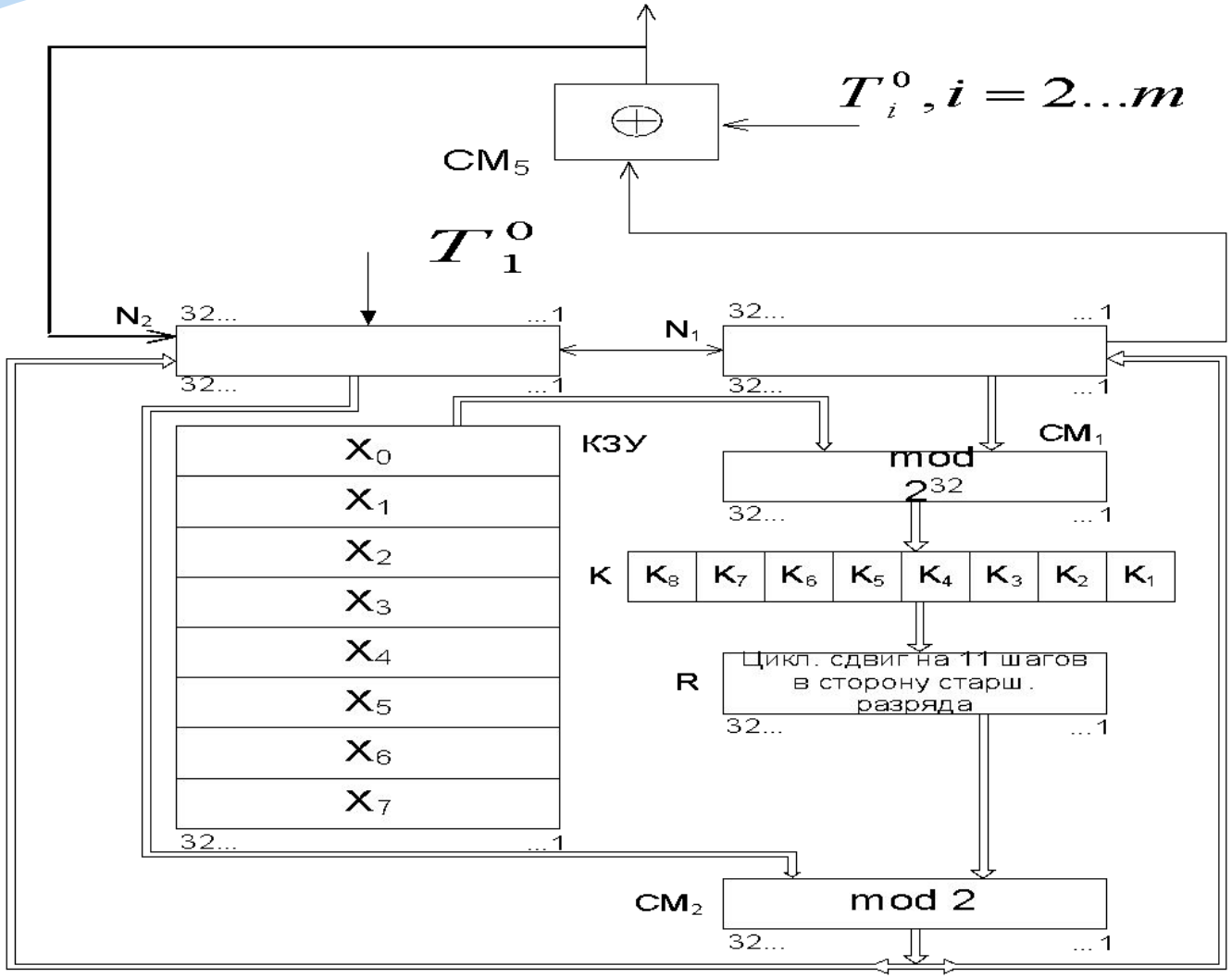
В режиме сцепления блоков CBC открытый текст для каждого блока Pi получается путем расшифрования по базовому алгоритму DES ECB P суммы по mod 2 соответствующего блока шифротекста текста и шифротекста предыдущего блока.

IV – начальный вектор. DES ECB P – расшифрование в режиме ECB.

Для получения открытого текста первого блока используется начальный вектор IV вместо шифротекста предыдущего блока.

$$C_i = DES(C_{i-1} \oplus M_i), \qquad M_i = DES^{-1}(C_i) \oplus C_{i-1}$$

Режим выработки имитовставки ГОСТ 28147-89



$И_р$ – имитовставка, вырабатываемая по определенному правилу

Алгоритмы ЭЦП

- Для решения проблем MAC, связанных с трудностями распределения ключей в симметричных системах, применяются алгоритмы электронной цифровой подписи (ЭЦП). Эти алгоритмы являются важной частью современных информационных систем,
- В алгоритмах ЭЦП используется не один ключ, а ключевая пара. Один из ключей в этой паре называется открытым, второй – закрытым (секретным, личным). Открытый ключ известен всем участникам, желающим получать сообщения от данного отправителя и проверять его подписи. Закрытый ключ известен только отправителю сообщения.
- Формирование ЭЦП осуществляется отправителем на основе сообщения и закрытого ключа. Проверка ЭЦП осуществляется получателем с помощью сообщения, открытого ключа отправителя и цифровой подписи. Таким образом, обеспечивается аутентификация, целостность и неотрицание авторства – сформировать верную подпись для сообщения может только отправитель, знающий закрытый ключ.

Недостатки ЭЦП

- Такие системы позволяют обеспечить безопасную коммуникацию даже тогда, когда нет возможности распределить ключи по безопасному каналу. Однако схемы ЭЦП не свободны от недостатков. Как правило, вычисление подписи требует б'ольших затрат времени, чем формирование имитовставки. Генерация ключевой пары занимает ещё больше времени. Кроме того, необходимо контролировать передаваемые открытые ключи, чтобы злоумышленник не мог, перехватив их, организовать атаку «человек посередине».

Дополнение

Имитовставка, является функцией сообщения x, $=f(x)$. Она может служить для целей аутентификации сообщения и проверки его целостности. Поэтому имитовставки можно разделить на два класса:

код проверки целостности сообщения (MDC, англ. modification detection code), для проверки целостности данных (но не аутентификации), вычисляется путем хэширования сообщения;

код аутентификации сообщения (MAC, англ. message authentication code), для защиты данных от фальсификации, вычисляется с помощью хэширования сообщения с использованием секретного ключа.