Режимы выполнения алгоритмов симметричного шифрования

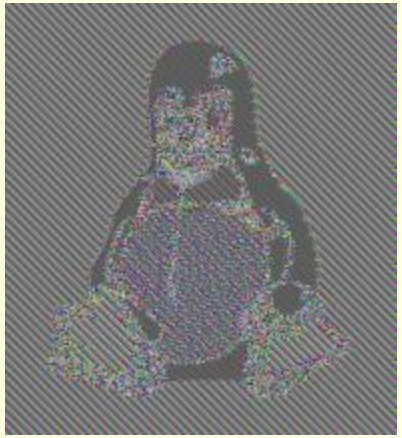
- *Режим ECB*
- *Режим CBC*
- *Режим PCBC*
- *Режим CFB*
- *Режим OFB*
- *Режим CTR*
- *Режим RD*
- *Режим RD с проверкой целостности*

Режим ECB

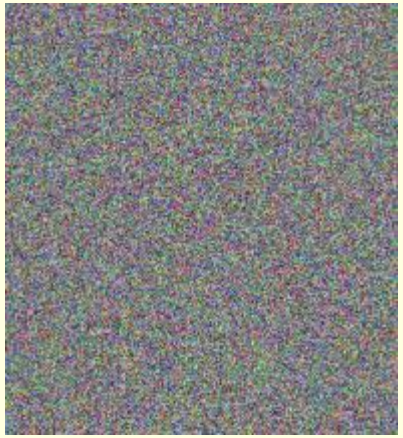
- В ГОСТ 28147—89 этот режим называется **режимом простой замены**.
- Electronic Codebook – каждый блок незашифрованного сообщения шифруется независимо от остальных блоков, с применением одного и того же ключа шифрования. Типичное применение – передача небольшого значения (например, криптографического ключа), длина которого сопоставима с длиной блока.
- Существенным недостатком ECB является то, что один и тот же блок незашифрованного сообщения, появляющийся более одного раза в сообщении, всегда имеет один и тот же зашифрованный вид. Вследствие этого для больших сообщений режим ECB считается небезопасным. Если сообщение имеет много одинаковых блоков, то при криптоанализе данная особенность может быть использована.

Режим ECB

Открытый текст в виде изображения



Криптограмма, полученная шифрованием в режиме ECB. На изображении видны черты исходного изображения



Криптограмма, полученная шифрованием в режиме, отличном от ECB. Изображение представляет собой псевдослучайную последовательность пикселей

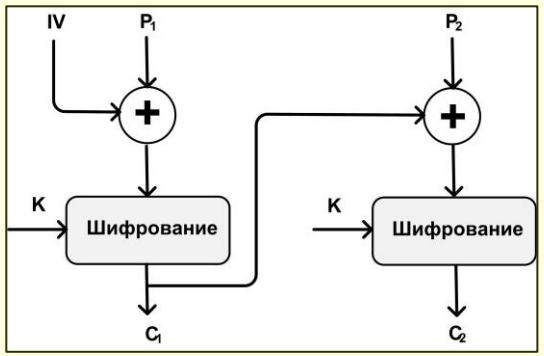
Режим ECB

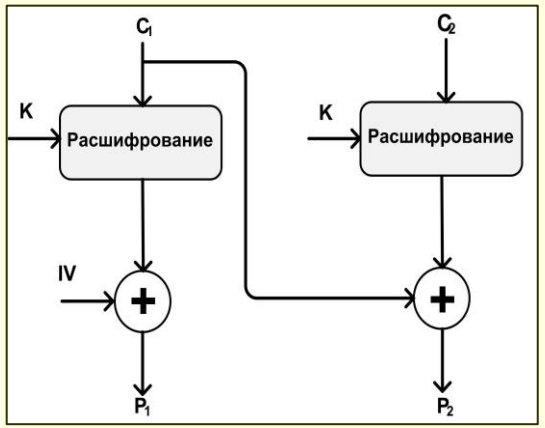
- **Достоинства ECB:**

- Постоянная скорость обработки блоков (скорость определяется эффективностью реализации шифра);
- Возможно распараллеливание вычислений (так как блоки не связаны между собой).

Режим CBC

- Cipher Block Chaining – на вход алгоритму шифрования подается результат операции **XOR** предыдущего блока зашифрованного сообщения и следующего блока незашифрованного сообщения. Типичное применение – шифрование больших сообщений. $C_i = E_k [P_i \oplus C_{i-1}]$





Для получения первого блока зашифрованного сообщения используется инициализационный вектор (IV), для которого выполняется операция **XOR** с первым блоком незашифрованного сообщения. При расшифровании выполняется операция **XOR** IV с выходом алгоритма расшифрования для получения первого блока незашифрованного текста.

IV должен быть известен как отправителю, так и получателю. Для максимальной безопасности IV должен быть защищен так же, как ключ.

Режим CBC

- **Недостатки CBC:**

- Возможность определения начала изменения данных по изменению шифротекста (если сравнить шифротексты двух сообщений с одним и тем же ключом, то номер первого блока, в котором шифротексты различаются, будет соответствовать номеру первого блока, в котором различаются исходные сообщения);
- Невозможность распараллеливания шифрования (поскольку для шифрования каждого *i*-го блока требуется блок, зашифрованный на предыдущем шаге (блоки связаны между собой)).

Режим CBC

- **Достоинства CBC:**

- Постоянная скорость обработки блоков (скорость определяется эффективностью реализации шифра; время выполнения операции «xor» пренебрежимо мало);
- Скрытие статистических особенностей, характерных для режима ECB (поскольку каждый блок открытого текста «смешивается» с блоком шифротекста, полученным на предыдущем шаге шифрования);

Режим PCBC (Propagating Cipher Block Chaining)

- Усовершенствованный режим **распространяющегося сцепления блоков шифра** (Propagating Cipher Block Chaining, PCBC) похож на CBC за исключением того, что для предыдущего блока открытого текста и предыдущего блока шифртекста выполняется операция XOR с текущим блоком открытого текста перед шифрованием.

$$C_i = E_k [P_i \oplus P_{i-1} \oplus C_{i-1}]$$

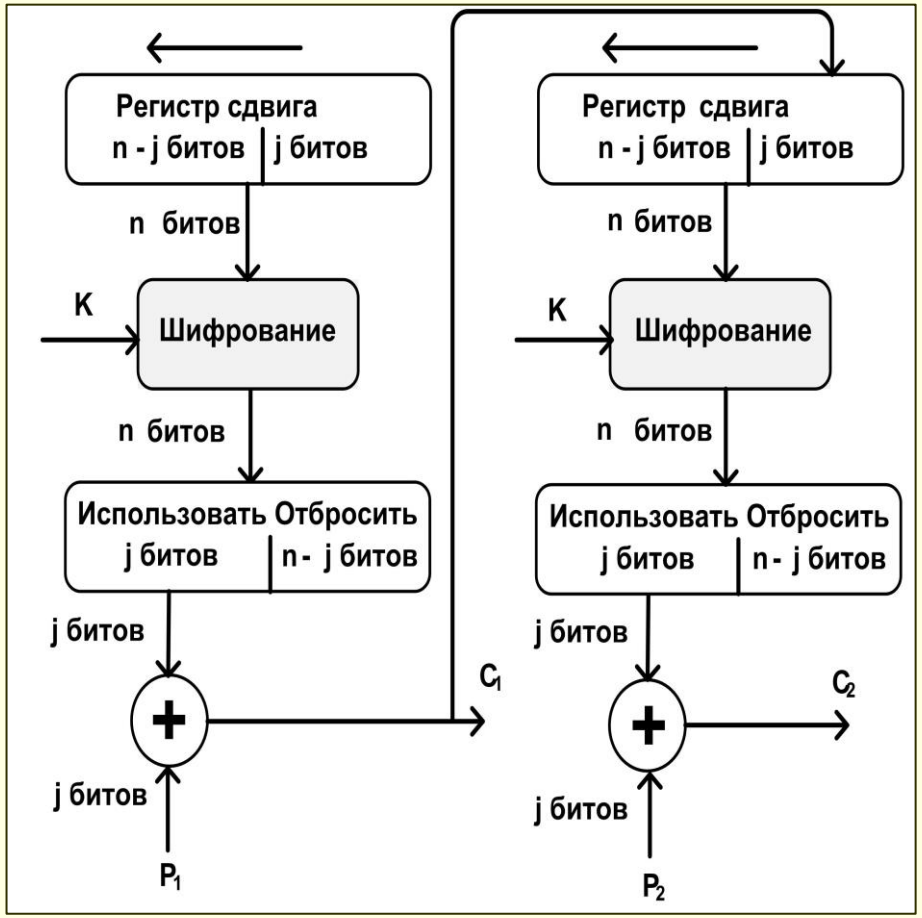
Режим CFB

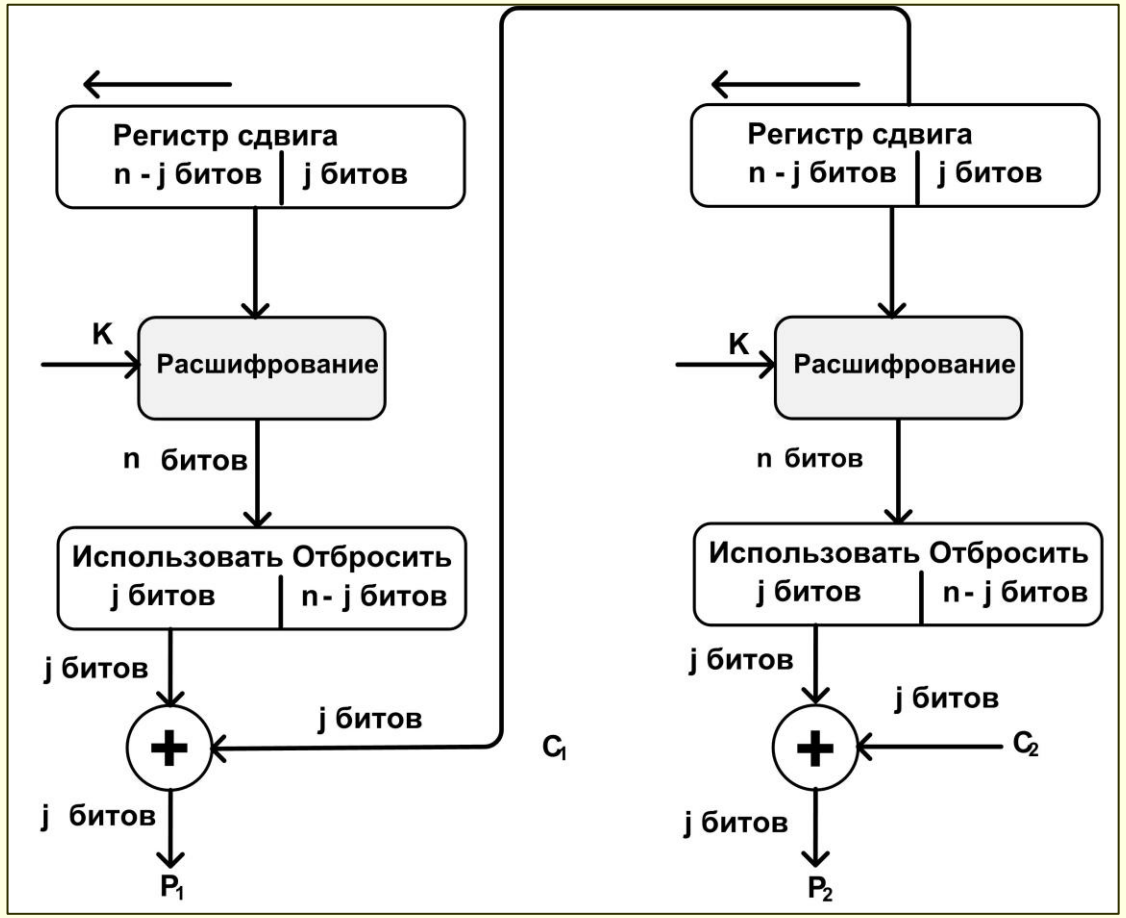
- Cipher Feedback – при каждом вызове алгоритма обрабатывается **J** битов входного значения. Предыдущий зашифрованный блок используется в качестве входа в алгоритм; к **J** битам выхода алгоритма и следующему незашифрованному блоку из **J** битов применяется операция **XOR**, результатом которой является следующий зашифрованный блок из **J** битов. Типичное применение – шифрование потока небольших сообщений в режиме реального времени, когда нецелесообразно ждать завершения формирования всего сообщения.
  - Блочный алгоритм предназначен для шифрования блоков определенной длины. Однако можно преобразовать блочный алгоритм в поточный алгоритм шифрования, используя последние два режима. Поточный алгоритм шифрования устраняет необходимость разбивать сообщение на целое число блоков достаточно большой длины. Таким образом, если передается поток символов, каждый символ может шифроваться и передаваться сразу, с использованием символьно ориентированного режима блочного алгоритма шифрования.
  - Одним из преимуществ такого режима блочного алгоритма шифрования является то, что зашифрованное сообщение будет той же длины, что и исходное.

Режим CFB

- Будем считать, что блок данных, используемый для передачи, состоит из **J** бит; обычным значением является **J=8**. Как и в режиме CBC, здесь используется операция **XOR** для предыдущего блока зашифрованного текста и следующего блока незашифрованного текста. Таким образом, любой блок зашифрованного текста является функцией от всего предыдущего незашифрованного текста.

Режим CFB

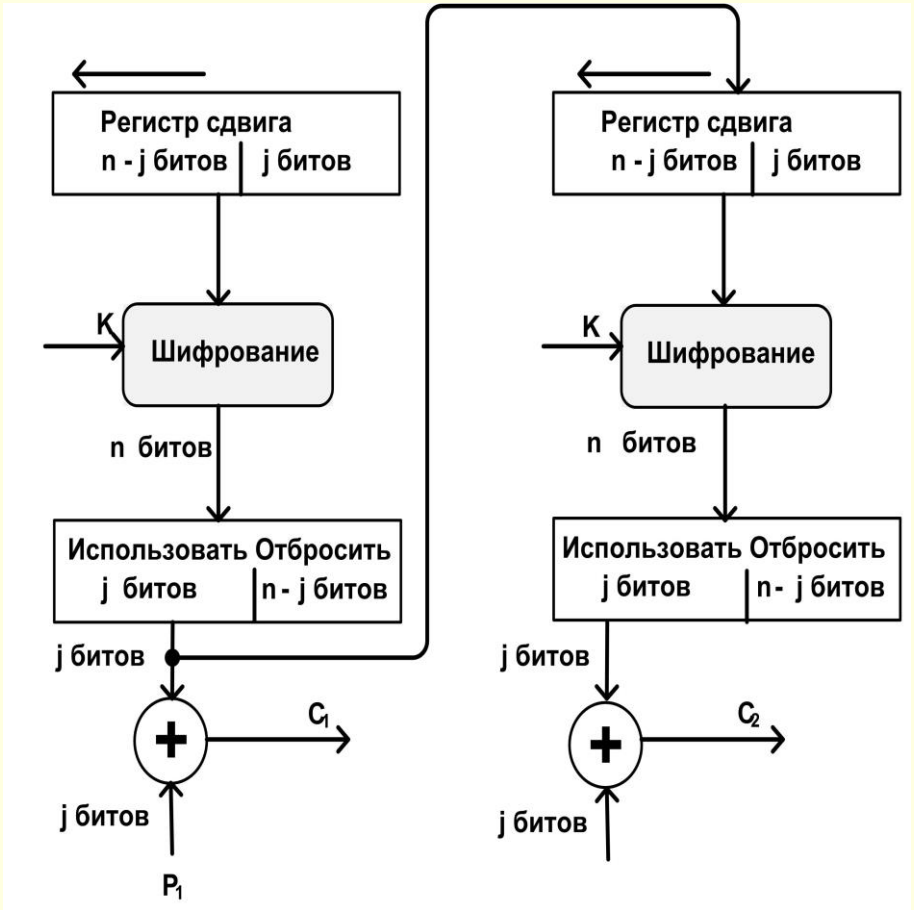

Регистр сдвига
n - j битов
j битов
n битов
K
Шифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
C1
j битов
P1
Регистр сдвига
n - j битов
j битов
n битов
K
Шифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
C2
j битов
P2

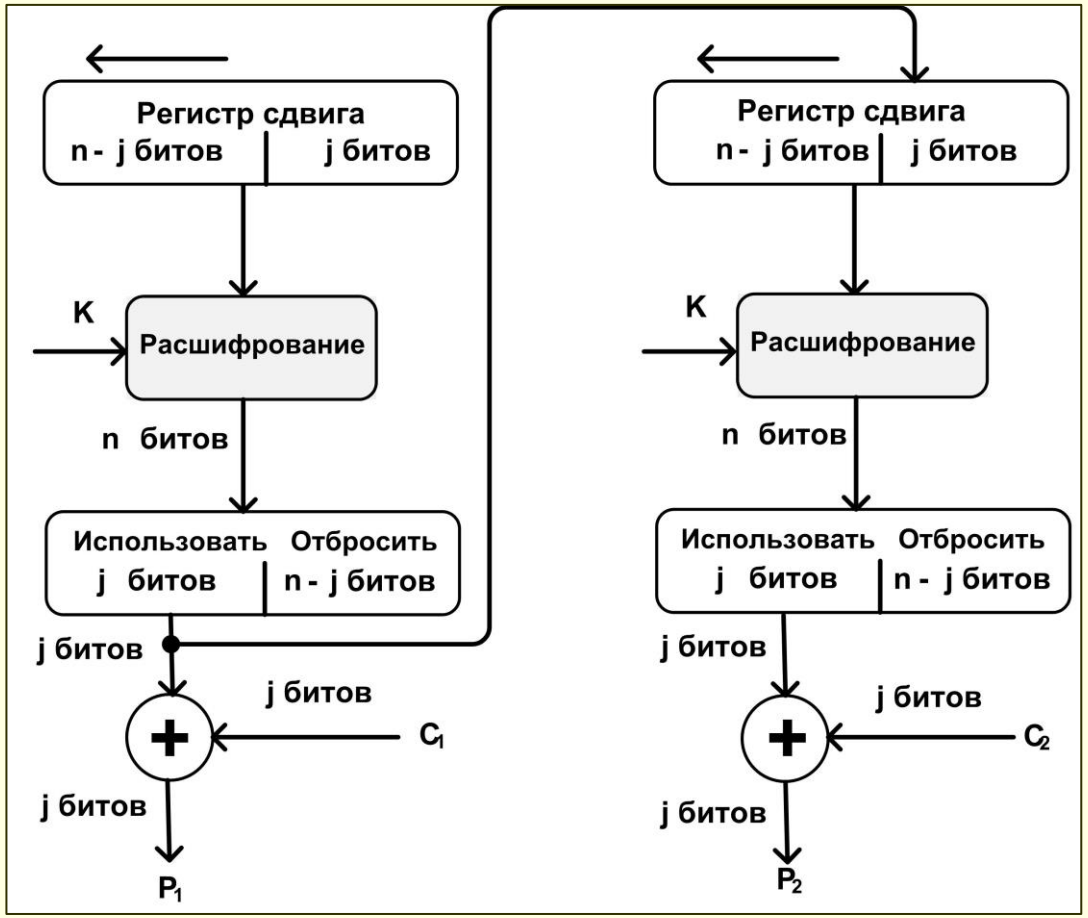

Регистр сдвига
n - j битов
j битов
K
Расшифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
j битов
C1
j битов
P1
Регистр сдвига
n - j битов
j битов
K
Расшифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
j битов
C2
j битов
P2

Режим OFB

- Output Feedback – аналогичен CFB, за исключением того, что на вход алгоритма при шифровании следующего блока подается результат шифрования предыдущего блока; только после этого выполняется операция **XOR** с очередными **J** битами незашифрованного сообщения. Типичное применение – потоко ориентированная передача по зашумленному каналу, в котором возможны потери пакетов.
- Основное преимущество режима OFB состоит в том, что если при передаче произошла ошибка, то она не распространяется на следующие зашифрованные блоки, и тем самым сохраняется возможность расшифрования последующих блоков. Например, если появляется ошибочный бит в C_i, то это приведет только к невозможности расшифрования этого блока и получения P_i. Дальнейшая последовательность блоков будет расшифрована корректно. При использовании режима CFB C_i подается в качестве входа в регистр и, следовательно, является причиной последующего искажения потока.
- Недостаток OFB в том, что он более уязвим к атакам модификации потока сообщений, чем CFB.

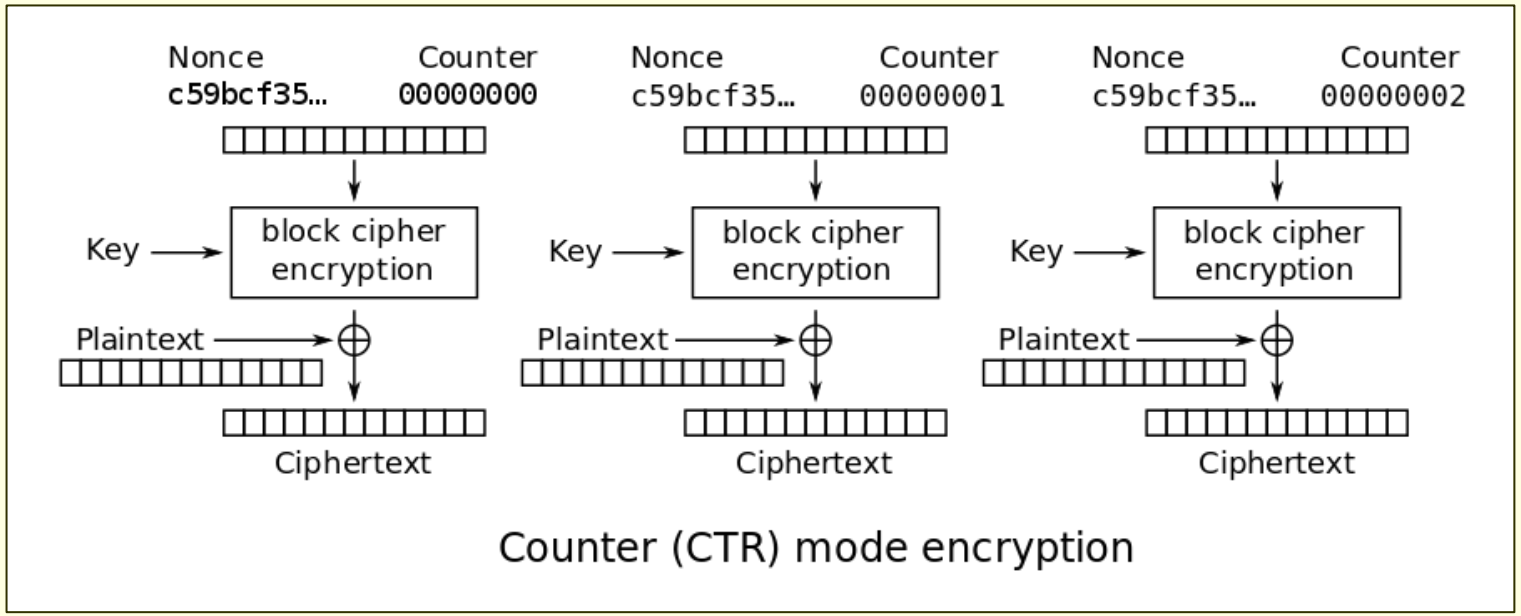
Режим OFB


Регистр сдвига
n - j битов
j битов
K
Шифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
C1
j битов
P1
Регистр сдвига
n - j битов
j битов
K
Шифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
C2
j битов


Регистр сдвига
n - j битов
j битов
K
Расшифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
j битов
C1
j битов
P1
Регистр сдвига
n - j битов
j битов
K
Расшифрование
n битов
Использовать
j битов
Отбросить
n - j битов
j битов
j битов
C2
j битов
P2

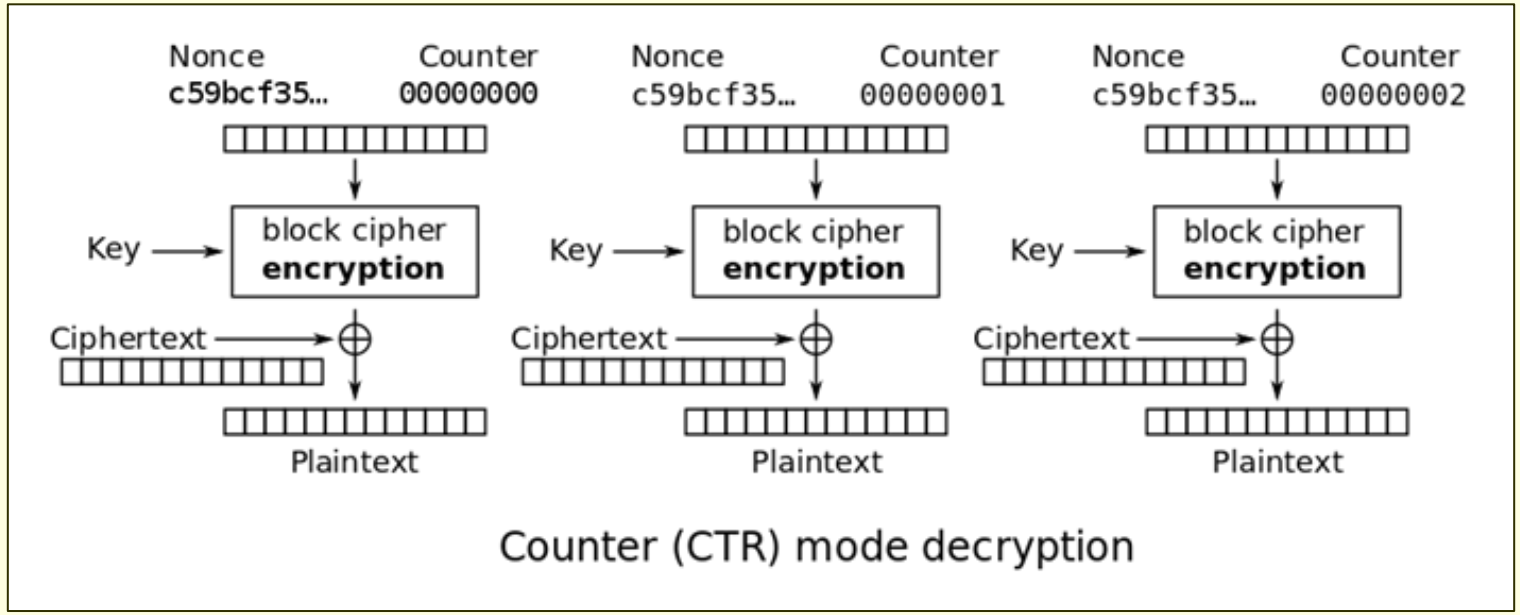
Режим CTR (Counter mode)

- **Режим счётчика (counter mode, CTR)** означает, что на вход соответствующего алгоритма блочного шифрования подаётся значение некоторого счётчика, который изменяется для каждого следующего блока. Режим делает из блочного шифра потоковый, то есть генерирует последовательность, к которой применяется операция XOR с текстом сообщения.
- **Шифрование в режиме CTR**



Режим CTR (Counter mode)

- **Расшифрование в режиме CTR**



$$P_i = C_i \oplus E_k(Ctr_i); i = 1, 2, \ldots, m$$

Режим CTR (Counter mode)

- Ctr_i — значение счётчика для i-го блока.
- Значения счётчика должны быть уникальными для каждого блока открытого текста, шифруемого на данном ключе (в противном случае блоки шифротекста, зашифрованные с помощью одинаковых значений счётчика, оказываются под угрозой). Это требование удовлетворяется следующим образом.

Режим CTR (Counter mode)

- Во-первых, значения счётчика для шифрования блоков одного сообщения получаются из начального значения счётчика использованием функции приращения. Чтобы обеспечивать случайность, величина приращения может зависеть от номера блока. Стандартная функция приращения может быть применена как ко всему блоку счётчика, так и к его части. Пусть значение счётчика является блоком из b битов, а функцию приращения применяем к m младшим разрядам.
- | — конкатенация; L_i — младшие m битов; M_i — старшие b-m битов. Уникальность значений счётчика обеспечивается для всех блоков сообщения при условии, что $n \leq 2^m$. Где n — количество блоков, на которое разбивается сообщение.

$$L_{i+1} = L_i + 1 \mod 2^m$$

$$Ctr_{i+1} = M_i | L_{i+1}$$

Режим CTR (Counter mode)

- Во-вторых, начальные значения счётчика для каждого сообщения выбираются таким образом, чтобы обеспечить уникальность всех используемых значений счётчика. Этого можно достичь разными способами. Например, если сообщения шифруются последовательно, то в качестве начального значения счётчика для данного сообщения можно использовать результат применения функции приращения к последнему значению счётчика предыдущего сообщения. При этом если функция приращения использует m битов, общее количество блоков открытого текста не должно превышать 2^m. Другой подход предлагает разбить двоичное представление счётчика на две части. Старшие разряды назначаются одноразовым номером сообщения, а к оставшимся будет применяться функция приращения.
- В результате отсутствия обратной связи алгоритмы шифрования и расшифровки в режиме CTR могут выполняться параллельно. Более того, большие объёмы вычислений, связанные с шифрованием значений счётчика, могут быть выполнены заранее, до того, как открытый текст или шифротекст окажутся доступными. Это обеспечивает режиму CTR преимущество перед режимами CFB и OFB.

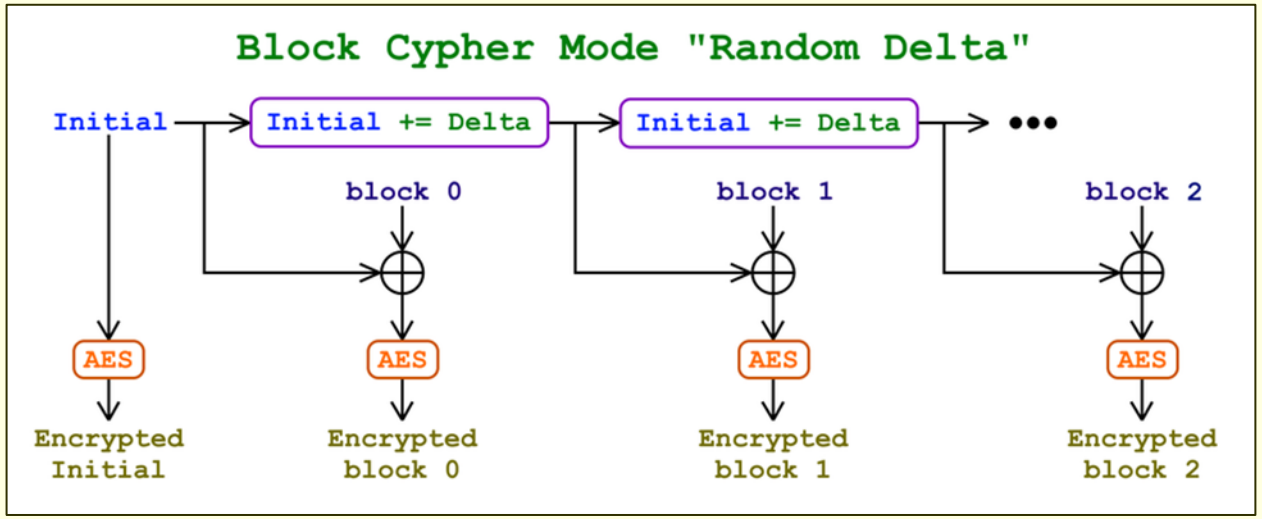
Режим RD (Random Delta)

- Режим Random Delta используется для устранения предсказуемости изменения счётчика в режиме CTR. Берётся случайный Initialization Vector (например, с помощью любого генератора всевдослучайных чисел). Его младшие 8 байт считаются случайной дельтой — Random Delta (RD):

Initialization Vector для Random Delta

Режим RD (Random Delta)

- Initial (Initialization Vector) шифруется и передаётся в начале сообщения. Блок 0 перед шифрованием XOR-ится с Initial. Для каждого последующего блока величина Initial увеличивается на Delta (в беззнаковом целочисленном представлении — uint128 += uint64):

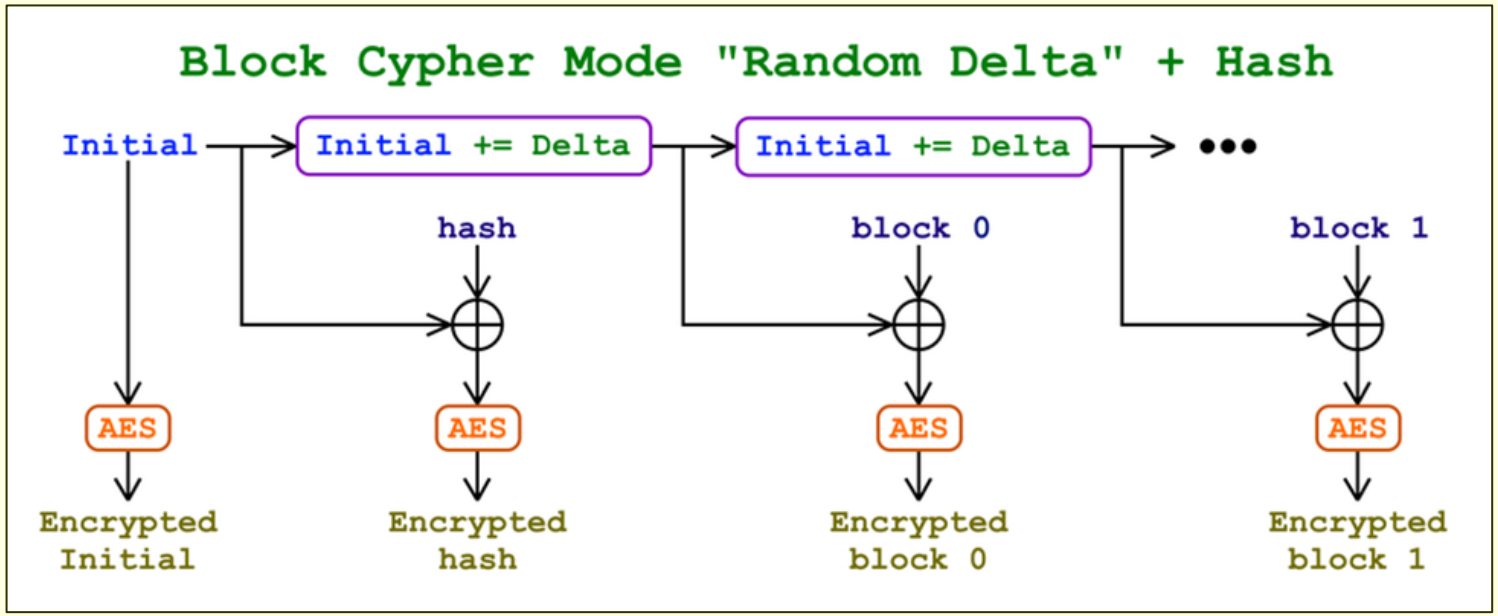


Режим RD (Random Delta)

- Таким образом устраняется предсказуемость изменения счётчика в режиме CTR. Если там дельта — всегда единица, здесь дельта — случайное число, одно из 2^{64}. Злоумышленнику оно, как и Initial, не известно.
- В CTR для преобразования открытого текста в шифротекст используется XOR. В Random Delta для преобразования открытого текста в шифротекст используется AES.

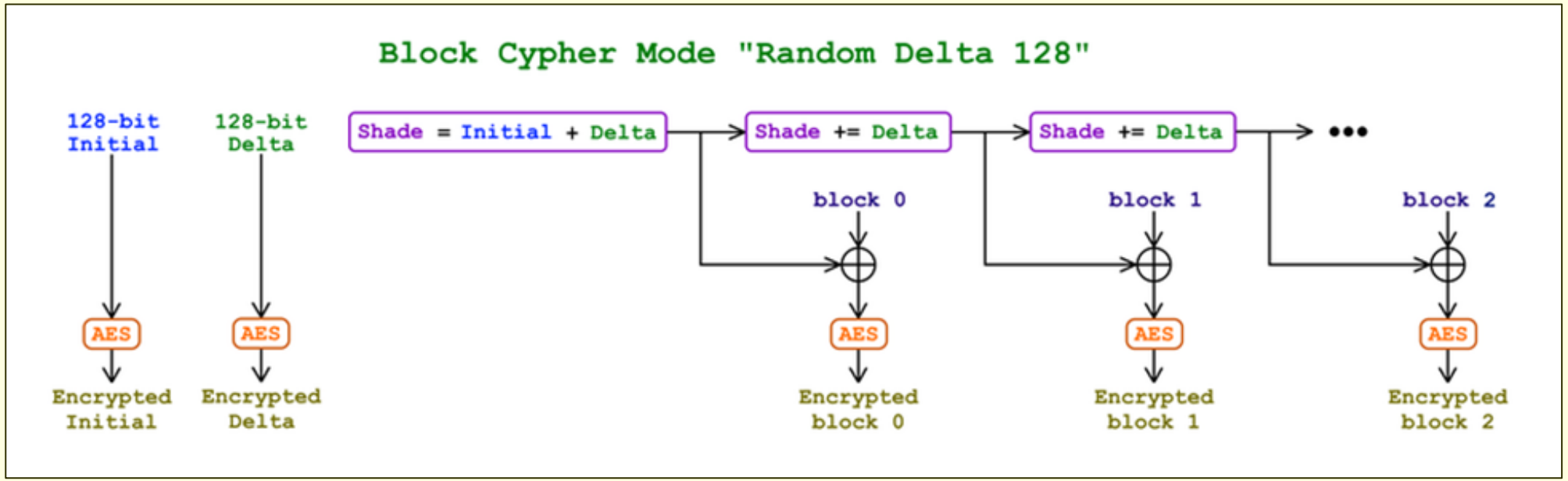
Режим RD с проверкой целостности

- Для проверки целостности в последовательность блоков может быть добавлен хеш.



Режим «Random Delta 128»

- Режим «Random Delta 128» отличается использованием отдельных 128-битных Initial и Delta.



Создание случайных чисел

- *Требования к случайным числам*
- *Криптографически созданные случайные числа*

Создание случайных чисел

Использование случайных чисел

1. Схемы взаимной аутентификации. В большинстве сценариев аутентификации и распределения ключа для предотвращения атак повтора (replay-атак) используются случайные числа, которые в этом случае называются nonce (number only once – число, используемое только один раз). Применение действительно случайных чисел в качестве nonce не дает противнику возможности вычислить или угадать nonce.
2. Ключ сессии, созданный центром распределения ключей (Key Distribution Center – KDC) или кем-либо из участников.

Создание случайных чисел

Требования к случайным числам

Двумя основными требованиями к последовательности случайных чисел являются случайность и непредсказуемость.

1. **Случайность**

   - Обычно при создании последовательности псевдослучайных чисел предполагается, что данная последовательность чисел должна быть случайной в определенном статистическом смысле. Следующие два критерия используются для доказательства того, что последовательность чисел является случайной:
   - *Однородное распределение:* распределение чисел в последовательности должно быть однородным; это означает, что частота появления каждого числа должна быть приблизительно одинаковой.
   - *Независимость:* ни одно значение в последовательности не должно зависеть от других.

Создание случайных чисел

- Хотя существуют тесты, показывающие, что последовательность чисел соответствует некоторому распределению, такому как однородное распределение, теста для «доказательства» независимости нет. Тем не менее, можно подобрать набор тестов для доказательства того, что последовательность является зависимой. Общая стратегия предполагает применение набора таких тестов до тех пор, пока не будет уверенности, что независимость существует.

Создание случайных чисел

2. **Непредсказуемость**

- В приложениях, таких как взаимная аутентификация и генерация ключа сессии, нет жесткого требования, чтобы последовательность чисел была статистически случайной, но члены последовательности должны быть непредсказуемы. При действительно случайной последовательности каждое число статистически не зависит от остальных чисел и, следовательно, непредсказуемо. Однако последовательность случайных чисел на практике должна создаваться некоторым алгоритмом. В этом случае необходимо, чтобы нельзя было предугадать следующие элементы последовательности, зная предыдущие элементы и используемый алгоритм.

Создание случайных чисел

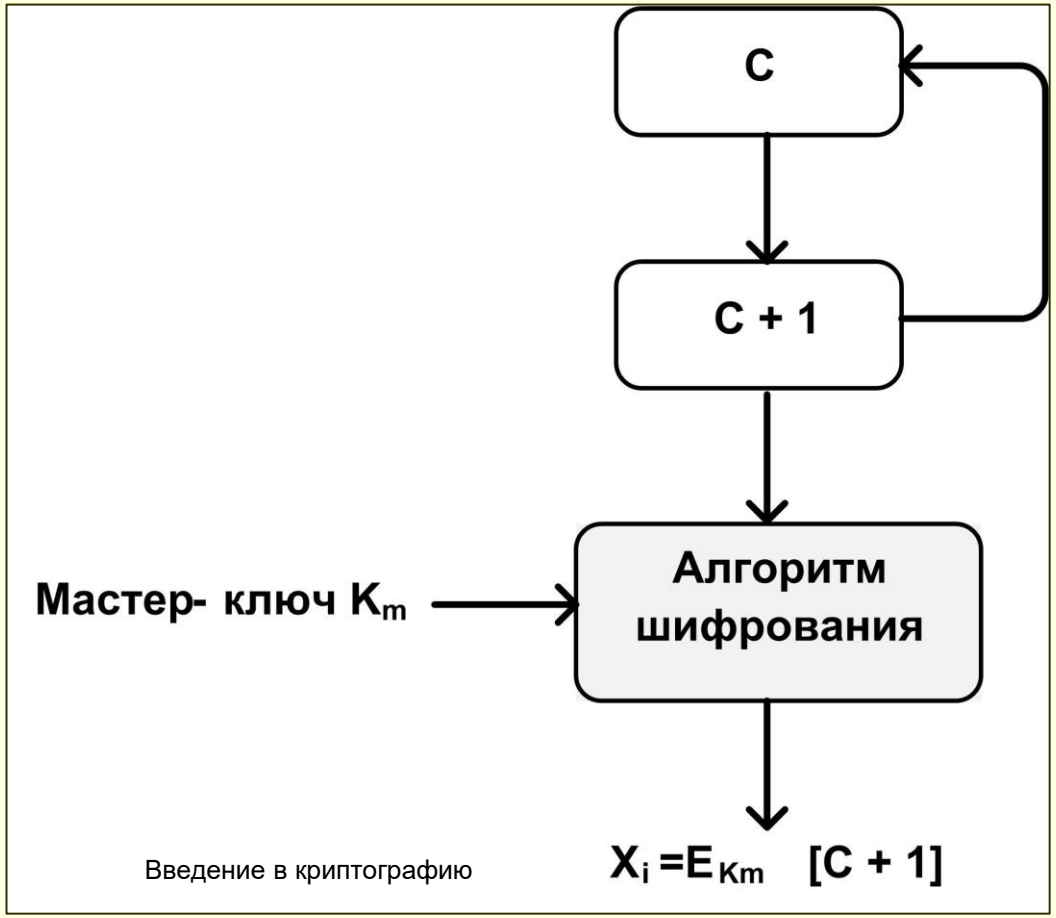
Источники случайных чисел

- Физические генераторы шумов, такие как детекторы событий ионизирующей радиации, газовые разрядные трубки и имеющий течь конденсатор могут быть такими источниками. Однако эти устройства в приложениях сетевой безопасности имеют ограниченное применение.
- Для создания случайных чисел приложения используют специальные алгоритмы. Эти алгоритмы детерминированы и, следовательно, создают последовательность чисел, которая не является статистически случайной. Тем не менее, если алгоритм хороший, полученная последовательность будет проходить много тестов на случайность. Такие числа часто называют псевдослучайными числами.

Создание случайных чисел

Генераторы псевдослучайных чисел

- **Циклическое шифрование**



Создание случайных чисел

Режим Output Feedback DES

Режим OFB DES может применяться для генерации ключа, аналогично тому, как он используется для потокового шифрования. Заметим, что выходом каждой стадии шифрования является 64-битное значение, из которого только левые `j` битов подаются обратно для шифрования. 64-битные выходы составляют последовательность псевдослучайных чисел с хорошими статистическими свойствами.

Создание случайных чисел

Генератор псевдослучайных чисел ANSI X9.17

DT_i – значение даты и времени создания i-го псевдослучайного числа.

V_i – начальное значение при создании i-го псевдослучайного числа.

R_i – i-ое псевдослучайное число.

K_1, K_2 – ключи, используемые на каждой стадии.

$$R_i = EDE_{K1,K2}\,[EDE_{K1,K2}\,[DT_i] \oplus V_i]$$
$$V_{i+1} = EDE_{K1,K2}\,[EDE_{K1,K2}\,[DT_i] \oplus R_i]$$

Даже если псевдослучайное число R_i будет скомпрометировано, вычислить V_{i+1} из R_i невозможно, и, следовательно, невозможно найти следующее псевдослучайное значение R_{i+1}.