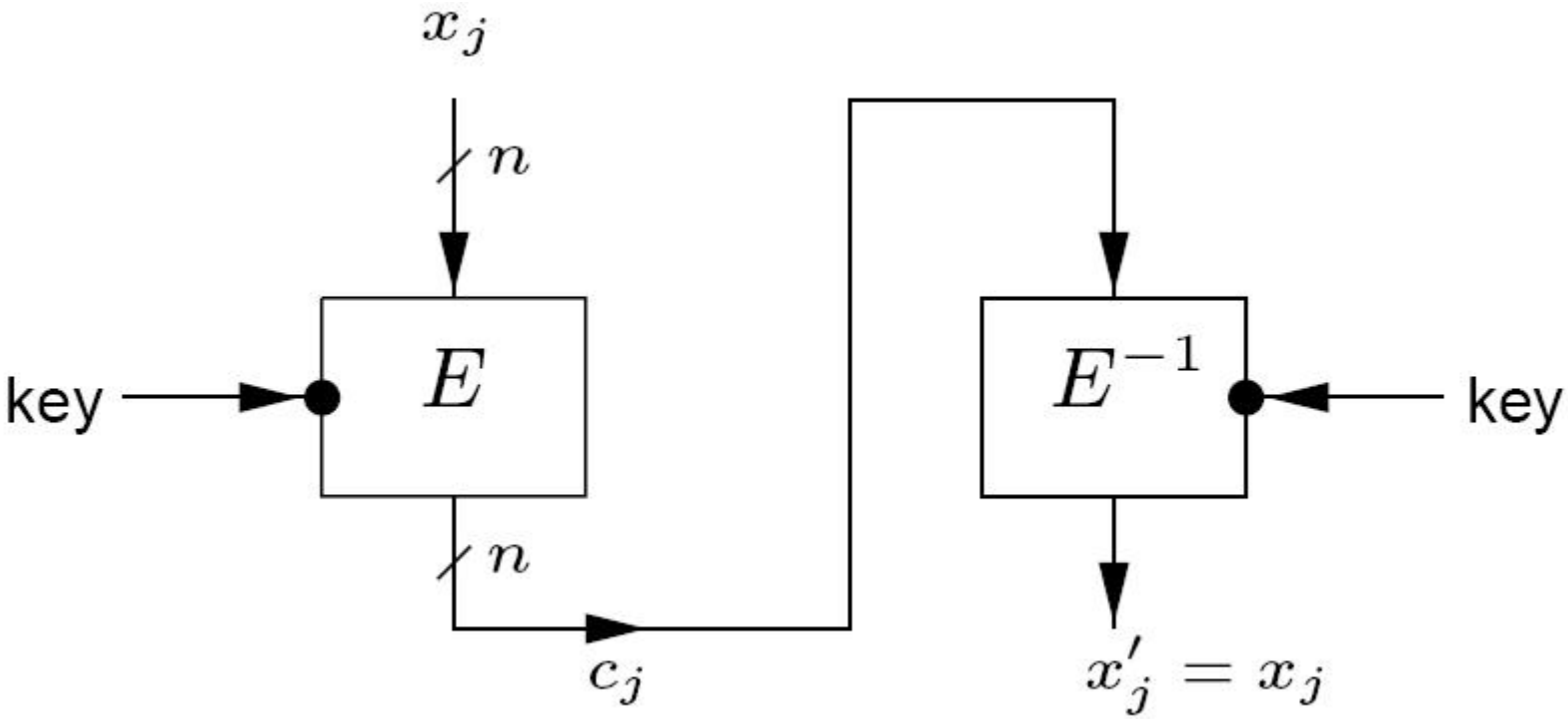
ИСПОЛЬЗОВАНИЕ БЛОКОВЫХ ШИФРОВ

Режимы шифрования

- Постановка задачи:
  - Предположим, что у нас есть хороший метод кодировать блоки длиной 128 бит (например AES).
  - Но сообщение длиннее. Что делать?
  - Мы сейчас будем рассматривать разные методы построения шифров для длинных сообщений из маленьких блоков.

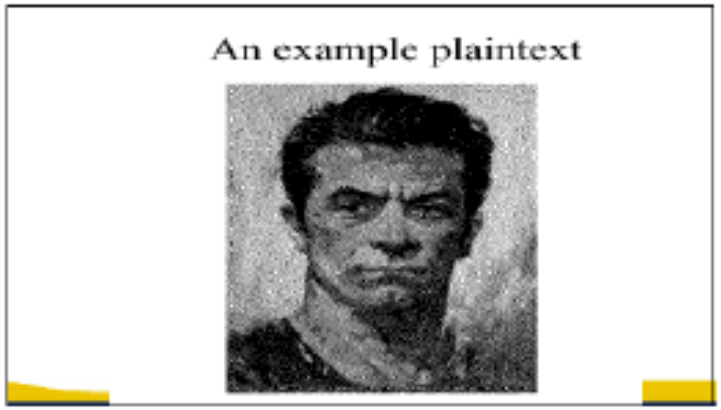
ECB

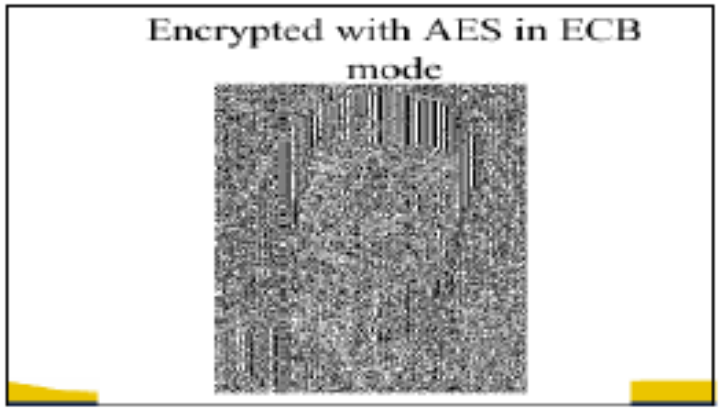
- Самое простое ECB (electronic codebook); блоки кодируются независимо друг от друга. Что плохо?



ECB: недостатки

- Одинаковые блоки переходят в одинаковый код.
- Блоки кодируются независимо; следовательно, их можно переставлять.
- Ошибка в одном блоке дальше этого блока не идет.

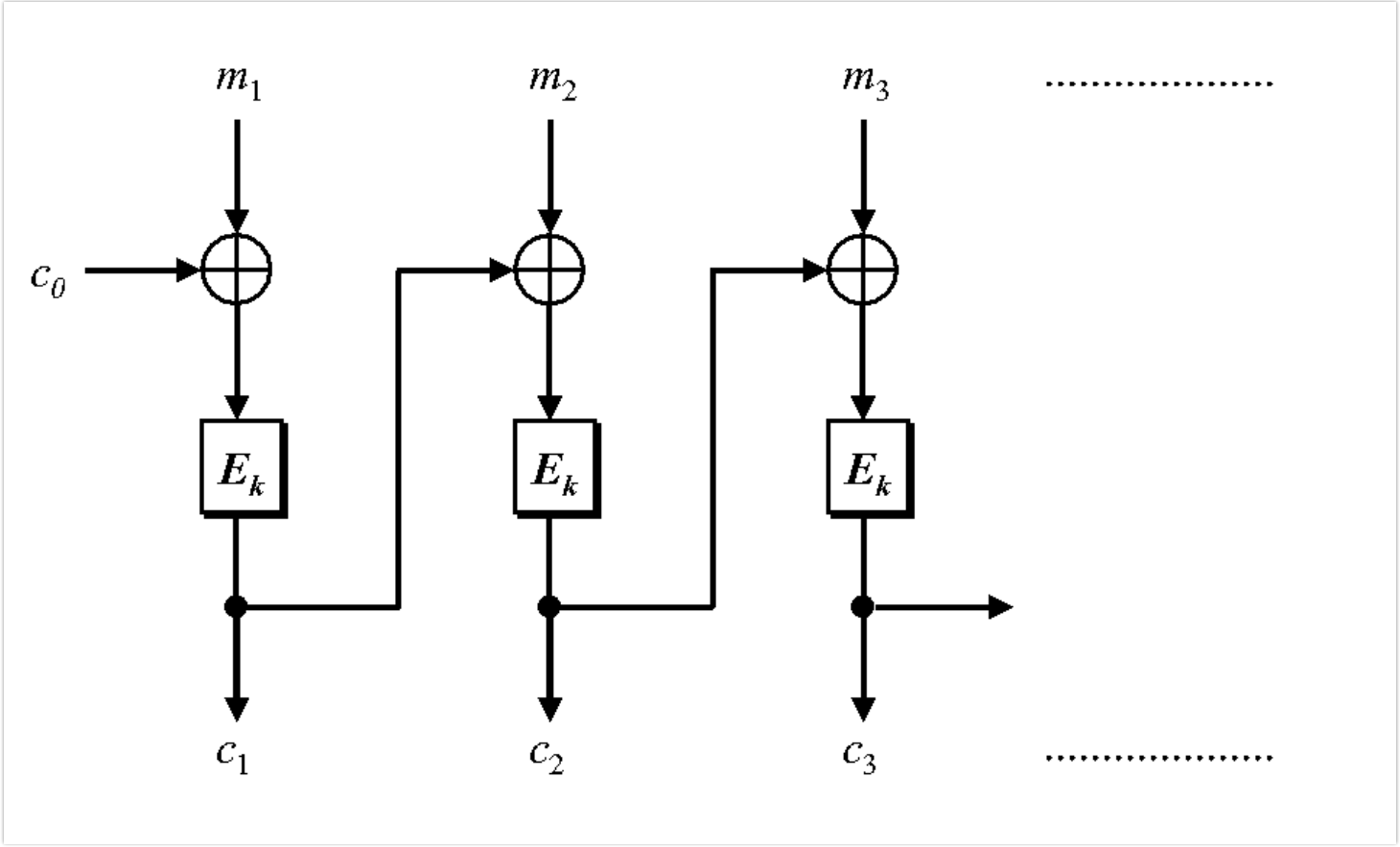




Не рекомендуется использовать!

CBC mode

- CBC (cipher block chaining): $c_0 = IV, c_j = E\left(k, m_j \oplus c_{j-1}\right)$



CBC

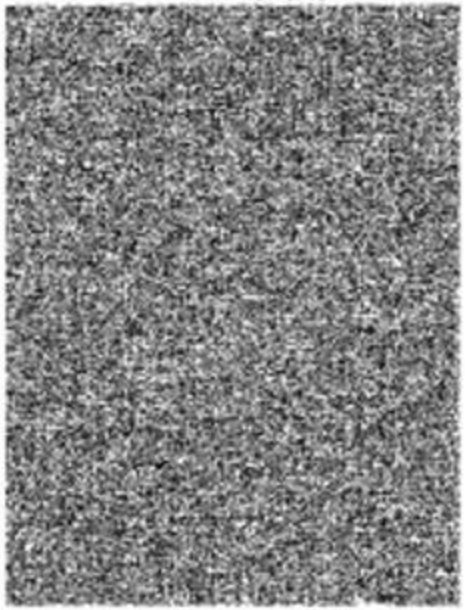
- Идентичные сообщения переходят в идентичные коды, только если IV тоже совпадает.
- Блок зависит от предыдущих, переставлять нельзя.
- Ошибка в блоке c_j вызывает ошибку в m_{j+1} на том же месте, дальше все OK; но m_j теряется полностью.
- Т.е. враг может подкорректировать m_{j+1} , но только ценой того, что m_j превратится в мусор.

CBC mode

- На примере изображения

An example plaintext

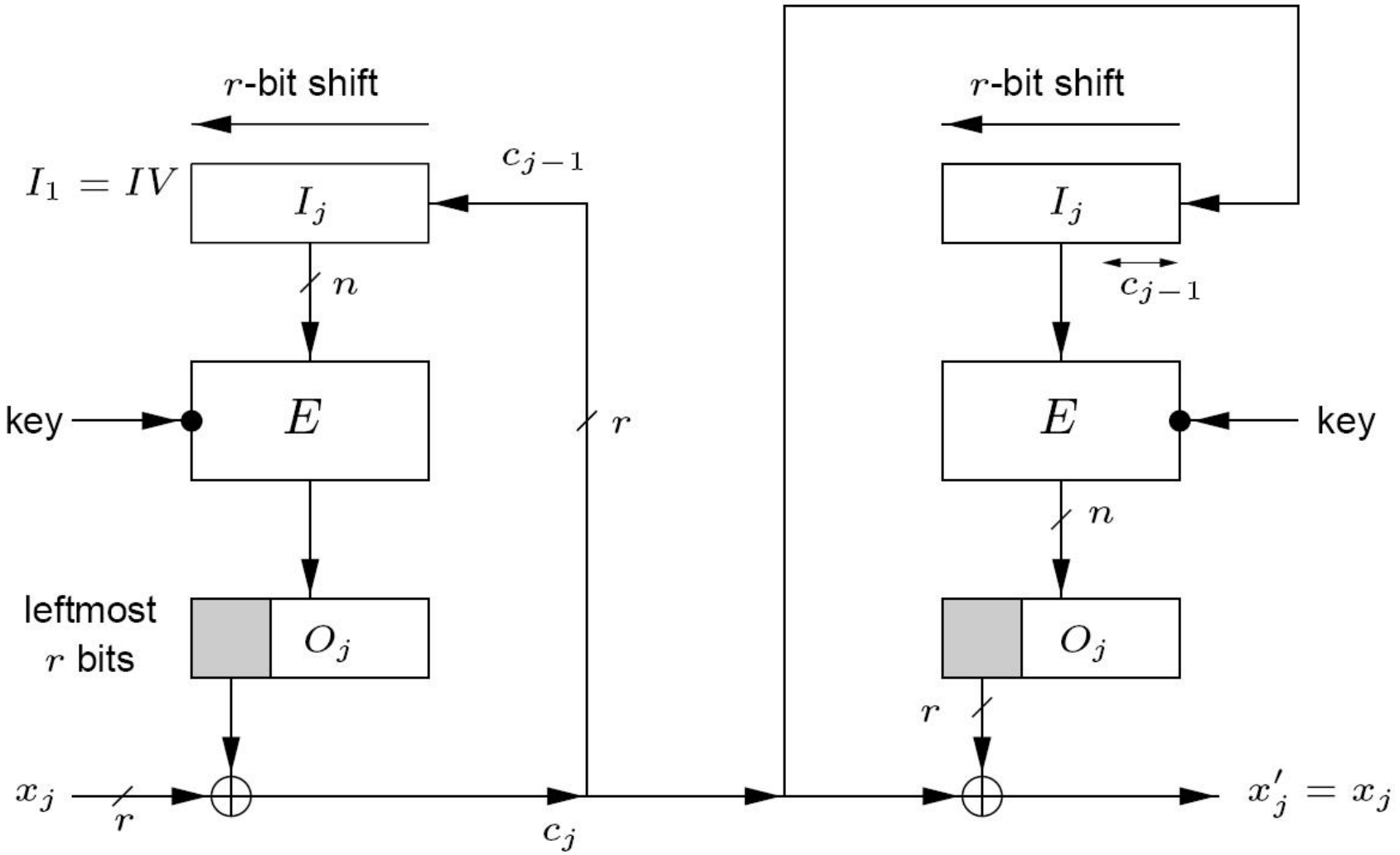
Encrypted with AES in CBC mode



CFB

- CFB (cipher feedback): когда нужно иногда отправлять сообщения размером меньше длины блока, скажем r бит. Тогда:
  - берем $l_0 = IV$,
  - считаем код $O_j = E(k, l_j)$,
  - берем t_j как r крайних слева битов O_j,
  - подсчитываем $c_j = t_j \oplus m_j$,
  - сдвигаем $l_{j+1} = 2^r \cdot l_j + c_j \ mod \ 2^n$

CBF


r-bit shift
$I_1 = IV$
I_j
c_{j-1}
n
key
E
r
leftmost
r bits
O_j
x_j
r
c_j
r-bit shift
I_j
c_{j-1}
E
key
n
O_j
r
$x'_j = x_j$

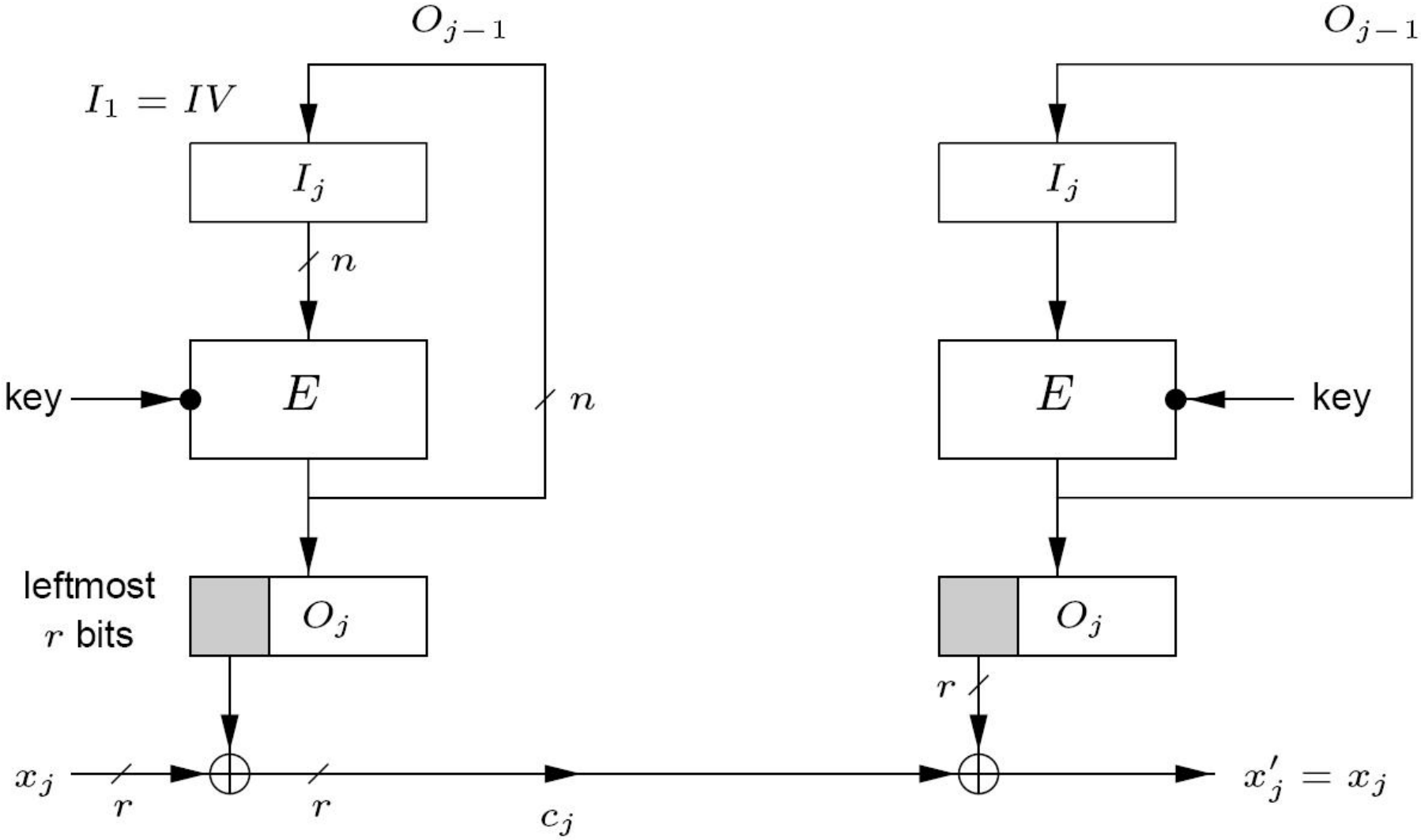
CBF:свойства режима

- Идентичные сообщения переходят в идентичные коды, только если IV тоже совпадает.
- Блок зависит от предыдущих, переставлять нельзя. Чтобы блок декодировался правильно, надо, чтобы предыдущие [n/r] были правильным.
- Ошибка в блоке c_j вызывает ошибку в [n/r] последующих блоках.
- Каждый запуск E выдает только r битов, а не n, как мог бы.

OFB

- OFB (output feedback): когда нехорошо, чтобы ошибка так далеко распространялась. Тогда:
  - берем $l_0 = IV$,
  - считаем код $O_j = E(k, l_j)$,
  - берем t_j как r крайних слева битов O_j,
  - подсчитываем $c_j = t_j \oplus m_j$,
  - берем следующий $l_{j+1} = O_j$.

OFB: схема


O_{j-1}
$I_1 = IV$
I_j
n
key
E
n
leftmost
r bits
O_j
x_j
r
r
c_j
O_{j-1}
I_j
E
key
O_j
r
$x'_j = x_j$

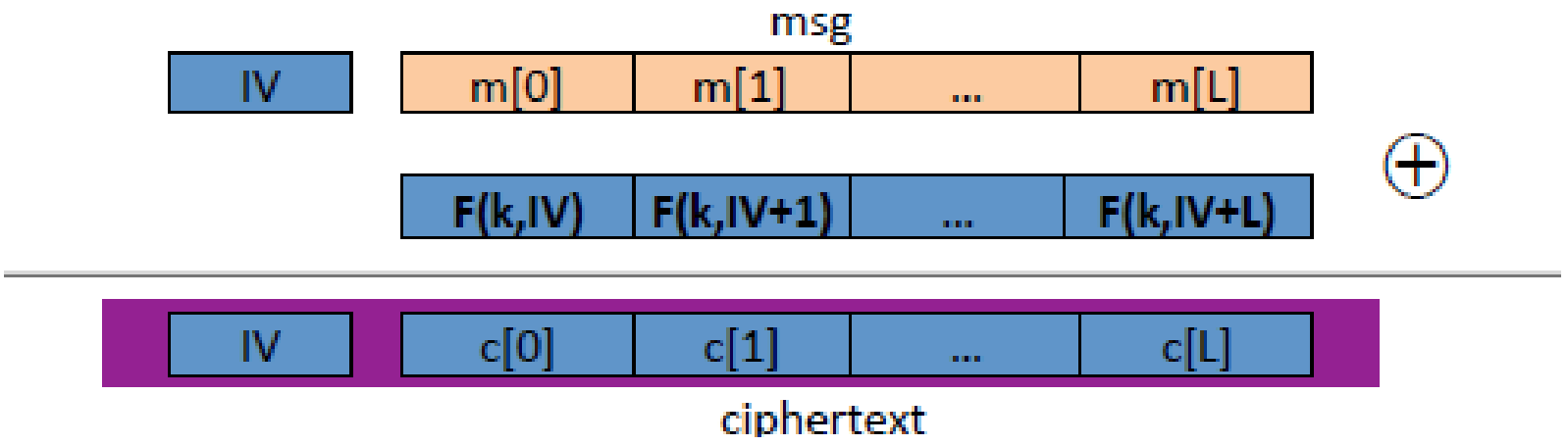
OFB: свойства

- Идентичные сообщения переходят в идентичные коды, только если IV тоже совпадает.
- Блок не зависит от предыдущих. Поэтому, в частности, нельзя использовать тот же ключ с тем же IV два раза.
- Ошибка в блоке c_j к другим ошибкам не приводит, но зато потерялась самосинхронизация.
- Каждый запуск E выдает только r битов, а не n, как мог
- бы; но зато теперь кодовый поток можно вычислять
- заранее.

CTR mode

- CTR (counter): когда необходимо, чтобы вычисления можно было производить параллельно. Тогда:
  - берем $l_0 = IV$,
  - считаем код $t_j = E(k, l_j)$,
  - подсчитываем $c_j = t_j \oplus m_j$,
  - берем следующий $l_{j+1} = l_j + 1$.

CTR: схема



CTR: свойства

- Идентичные сообщения переходят в идентичные коды, только если IV тоже совпадает.
- Блок не зависит от предыдущих. Поэтому, в частности, нельзя использовать тот же ключ с тем же IV два раза.
- Ошибка в блоке c_j к другим ошибкам не приводит, но зато потерялась самосинхронизация.
- Кодовый поток можно вычислять заранее или кодировать блоки параллельно.

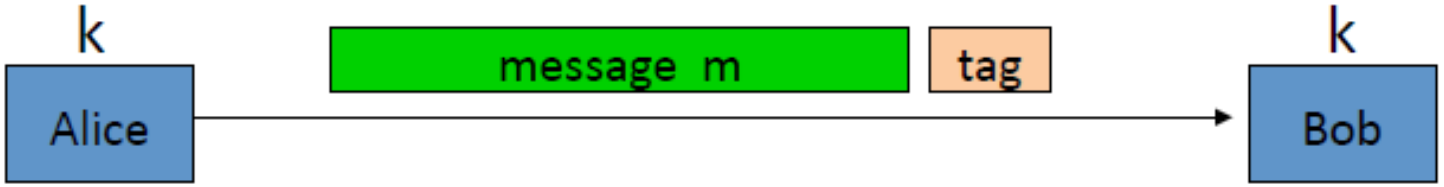
КОНТРОЛЬ ЦЕЛОСТНОСТИ

Message Auth. Codes

Целостность сообщения

- Цель: **целостность**, не конфиденциальность
- Примеры:
  - Сохранение на диск исполняемых файлов
  - Защита рекламных банеров

Целостность сообщения: MAC



Вычисление tag:

$$tag \leftarrow S(k, m)$$

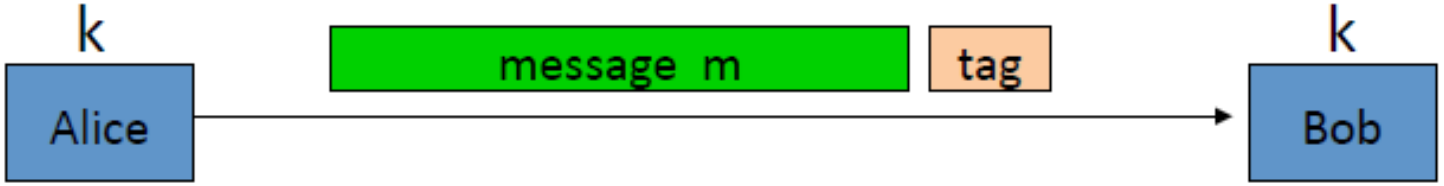
Проверка tag:

$$\mathrm{V}(k, m, tag) = \begin{cases} \text{да} \\ \text{нет} \end{cases}$$

Определение: MAC I=(S,V) определенный на множествах (K,M,T) это пара алгоритмов:

- S(k, m) возвращает tag
- V(k, m, tag) возвращает “да” или «нет»

Целостность требует обмена секретными ключами



Вычисление tag:

$$tag \leftarrow crc(m)$$

Проверка tag:

$$\mathrm{V}(m, tag) = \begin{cases} \text{да} \\ \text{нет} \end{cases}$$

- Атакующий может легко пересчитать m и значение CRC с нему
- CRC разработан для обнаружения **случайных** ошибок, но не вредоносных воздействий

Безопасность MAC

- Возможности атакующего:
  - Атака по выбранному сообщению:

$$m_1, m_2, m_3, \ldots, m_q \rightarrow t_i = S(k, m_i)$$

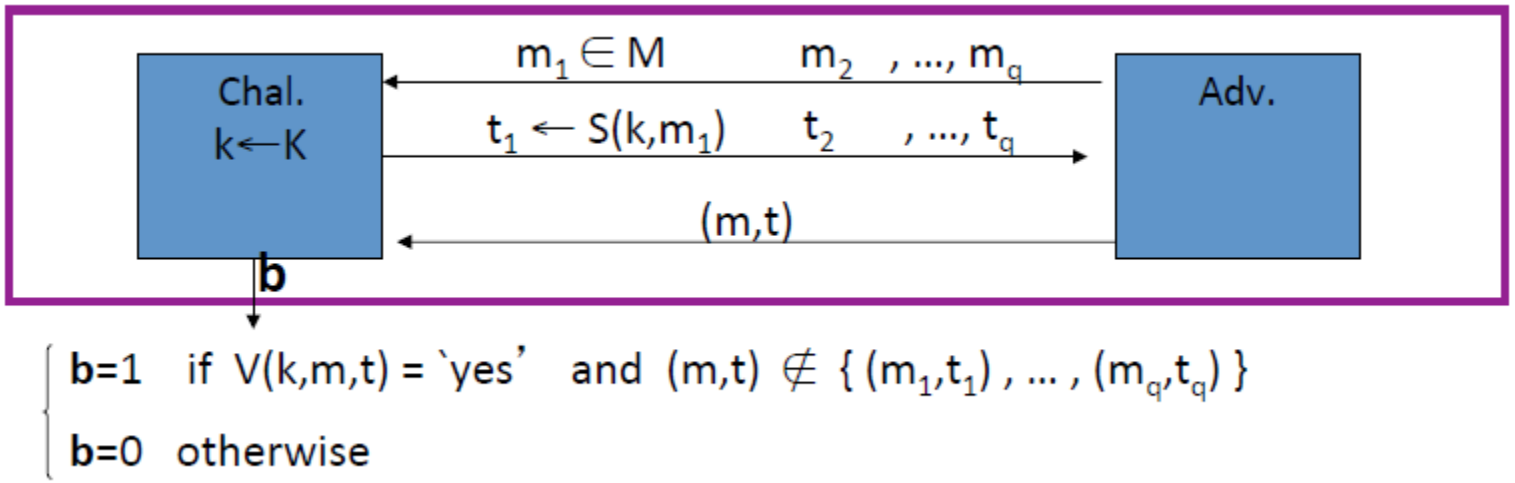
- Цель атакующего: **существующий полог**
  - Найти **любую** новую пару: сообщение/tag (m,t):

$$(m, t) \notin \{(m_1, t_1), (m_2, t_2), \ldots, (m_q, t_q)\}$$

- Атакующий не может построить проверку для нового сообщения
- Атакующий не может построить новую пару (m, t') для известной (m, t)

Безопасность MAC

- Для функции MAC I=(S,V) и атакующего A определим игру



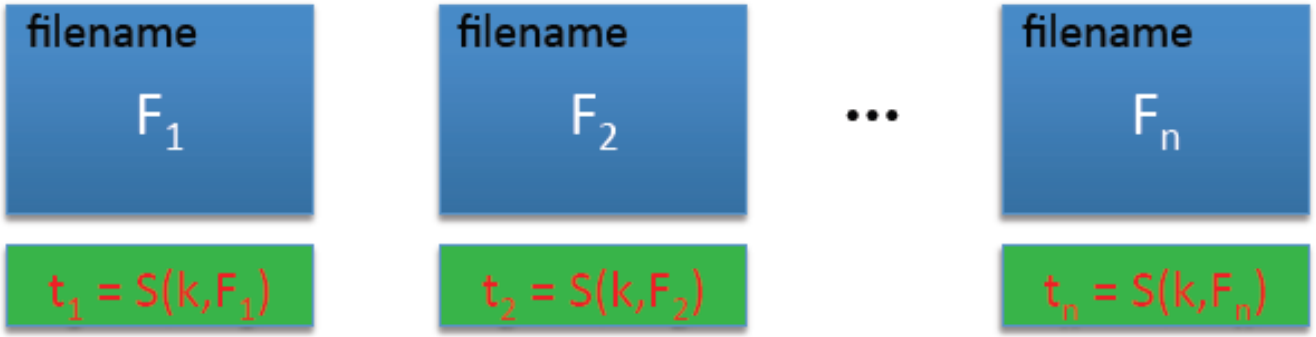
- Определение: функция MAC I=(S,V) является криптографически стойкой, если для всех «эффективных» A: $Adv_{MAC}[A, I] = \Pr\{b = 1\}$ -- neg

- Пусть задан MAC I=(S,V)
- Предположим, что атакующий способен найти пару $m_0 \neq m_1 : S(k, m_0) = S(k, m_1)$ для ½ k из K
- Может ли этот MAC быть безопасным?
  - Да, так как атакующий не может построить соответствующий tag
  - Нет, так как к функции может быть применена атака по выбранному сообщению
  - Все зависит от свойств MAC

- Пусть задан MAC I=(S,V)
- Предположим, что $S(k, m\ \ )$ всегда имеет длину 5 бит.
- Может ли этот MAC быть безопасным?
  - Нет, так как атакующий сможет просто угадать tag
  - Все зависит от свойств MAC
  - Да, так как атакующий не может построить tag для любого сообщения

Пример: Защита файловой системы

- При установке системы на диск

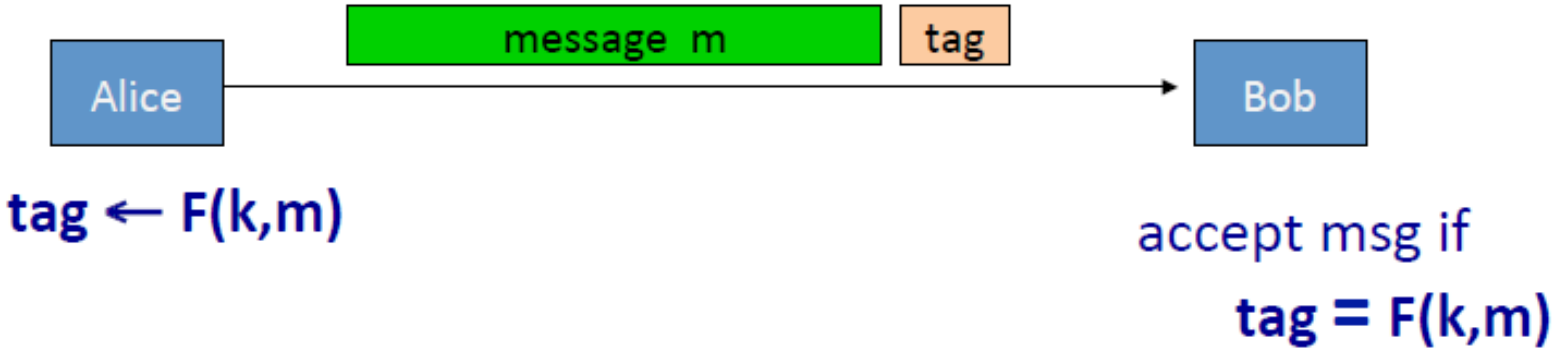


k вычисляют по паролю пользователя

- В случае инфицирования системы и искажения файлов
- Пользователь может перезагрузиться в чистой системе и ввести пароль
  - Тогда, MAC → все искаженные файлы будут обнаружены

Стойкая PRF → Стойкий MAC

- Определим MAC $I_F = (S, V)$ на основе стойкой PRF $F: K \times X \to Y$:
  - $S(k, m) = F(k, m)$
  - $V(k, m, t)$: возвращает «да», если $t = F(k, m)$, иначе «нет»



Пример

- Пусть задана стойкая PRF $F: K \times X \rightarrow Y: Y = \{0,1\}^{10}$
- Будет ли MAC, определенный на основе F?
  - Да, так как F – стойкая PRF
  - Нет, так как атакующий сможет просто угадать tag из-за малой мощности Y
  - Все зависит от свойств F

Безопасность

- Теорема:
- Если $F: K \times X \rightarrow Y$ криптографически стойкая PRF и $1/|Y|$ пренебрежимо мала (т.е. |Y| должна быть велика), тогда I_F стойкий MAC

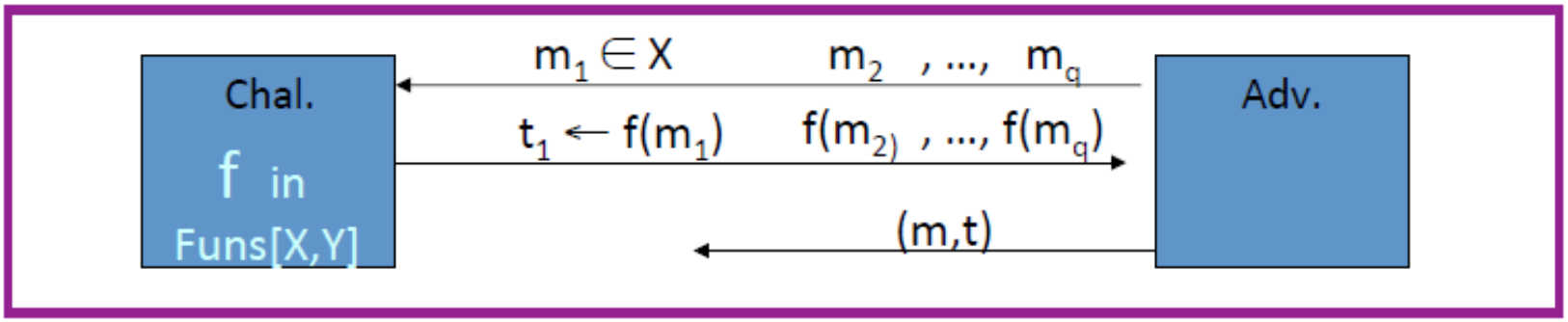
- В частности для любого эфф. MAC атакующего A существует эфф. атакующий B для PRF F:

$$Adv_{MAC}[A, I_F] \leq Adv_{PRF}[B, F] + 1/|Y|$$

⇒ I_F остается стойкой до тех пор пока |Y| велика

Доказательство

- Предположим, что f: X→Y -- настоящая случайная функция
- Тогда атакующий MAC A должен будет выиграть следующую игру:



- A выиграет, если $t = f(m)$ and $m \notin \{ m_1, \ldots, m_q \}$

$\Rightarrow$ Pr[A wins] = 1/|Y|

Примеры

- AES: MAC для 16 байт
- Основной вопрос: как сделать Small-MAC → Big-MAC
- Подходы:
  - CBC-MAC(банковские протоколы)
  - HMAC(Интернет)
- Оба обеспечивают решение

Укорочение MAC

- Лемма: Пусть $F: K \times X \longrightarrow \{0,1\}^n$ стойкая PRF, тогда

$$F_t(k,m) = F(k,m)[1\ldots t] \quad \text{for all} \quad 1 \le t \le n$$

- Если I(S,V) – n-битный MAC, основанный на стойкой PRF, тогда укороченный MAC, основанный на F_t, будет стойким пока t достаточно велико($1/2^t$ -- мало)

Построение MAC из PRF

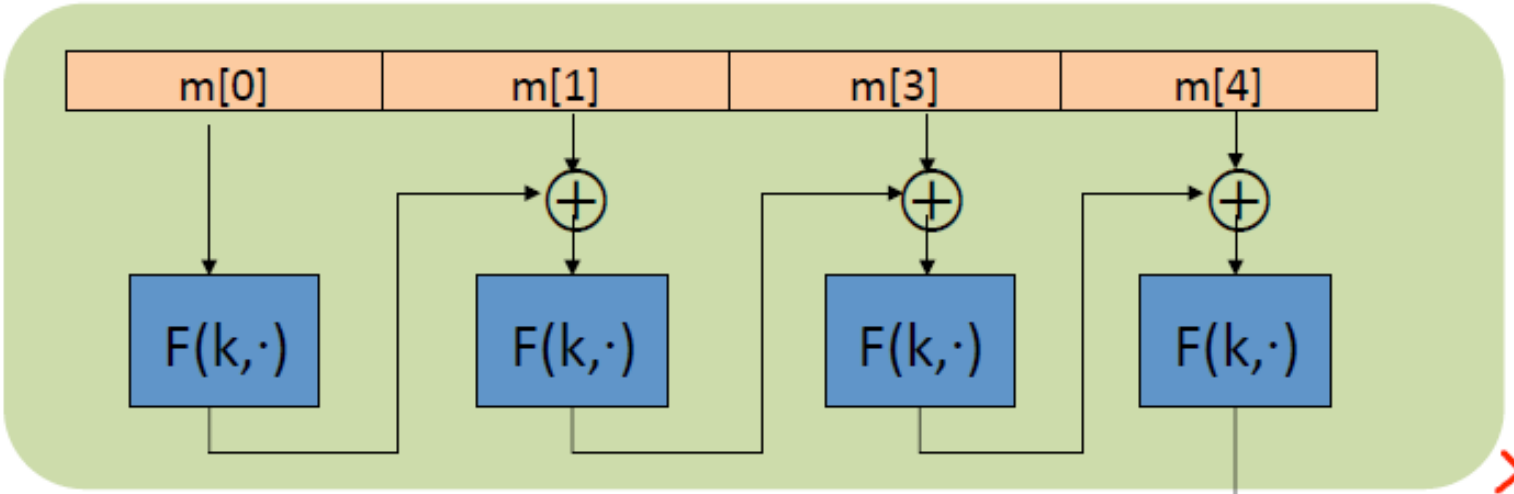
- Стойкая PRF → Стойкий MAC

- Цель: На основе PRF для коротких входных сообщений построить PRF для длинных сообщений

- Будем считать $X = \{0,1\}^n$

CBC-MAC

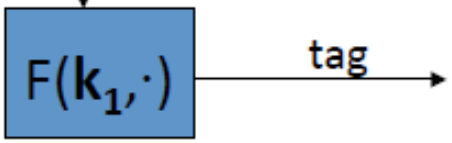
raw CBC



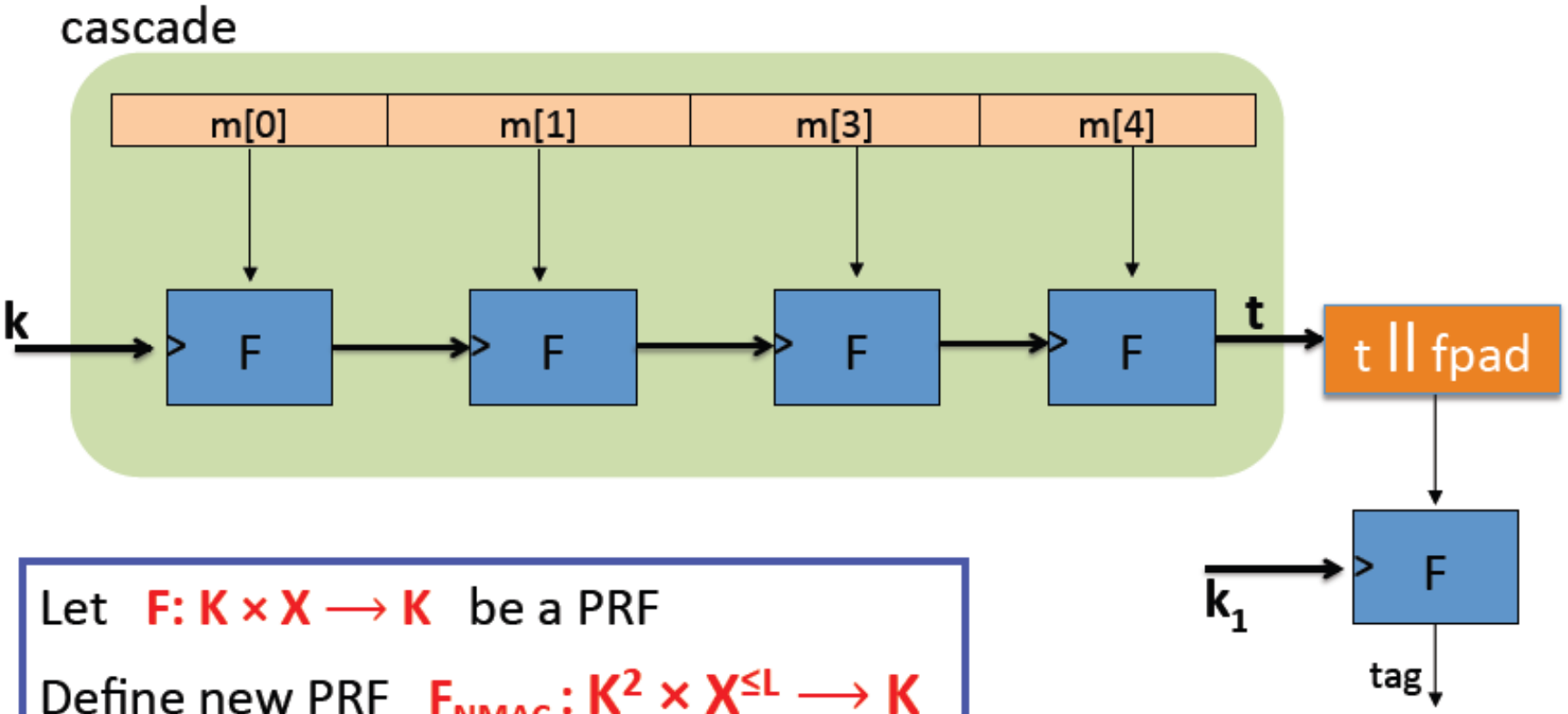
$X^{\leq L} = \bigcup_{i=1}^{L} X^i$

Let $F: K \times X \longrightarrow X$ be a PRP

Define new PRF $F_{ECBC}: K^2 \times X^{\leq L} \longrightarrow X$



NMAC



Let $F: K \times X \longrightarrow K$ be a PRF

Define new PRF $F_{NMAC}: K^2 \times X^{\leq L} \longrightarrow K$