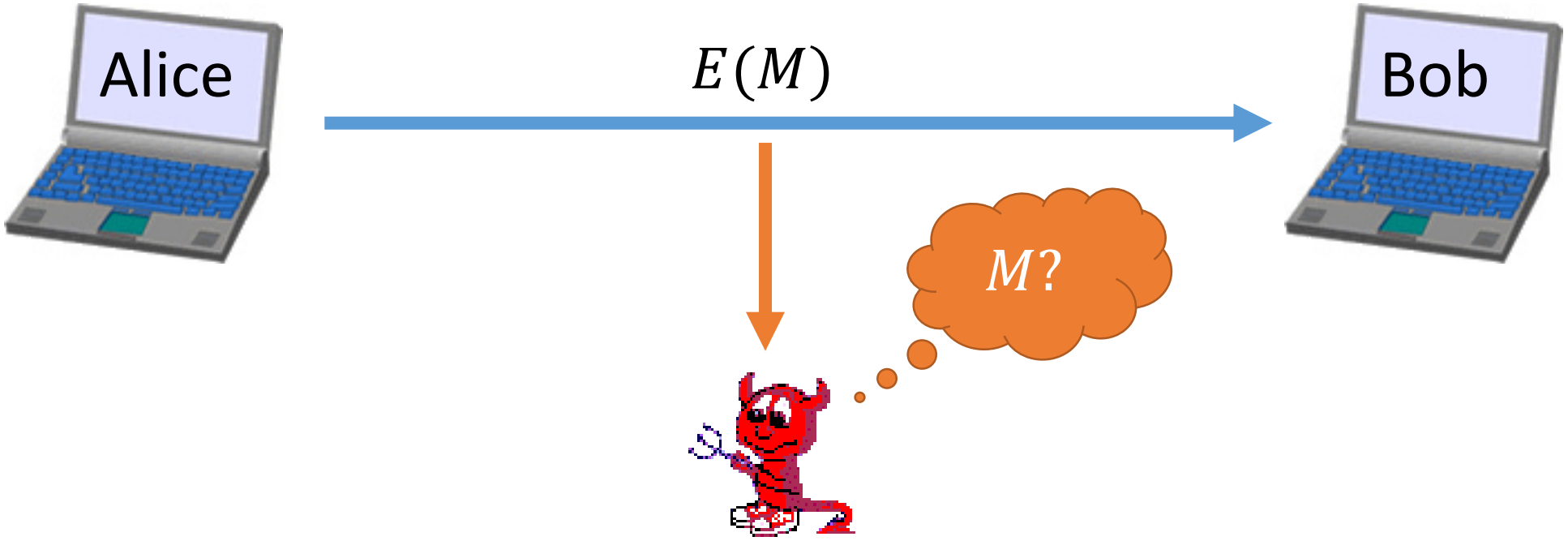
Коды аутентичности сообщений

Макаров Артём

МИФИ 2020

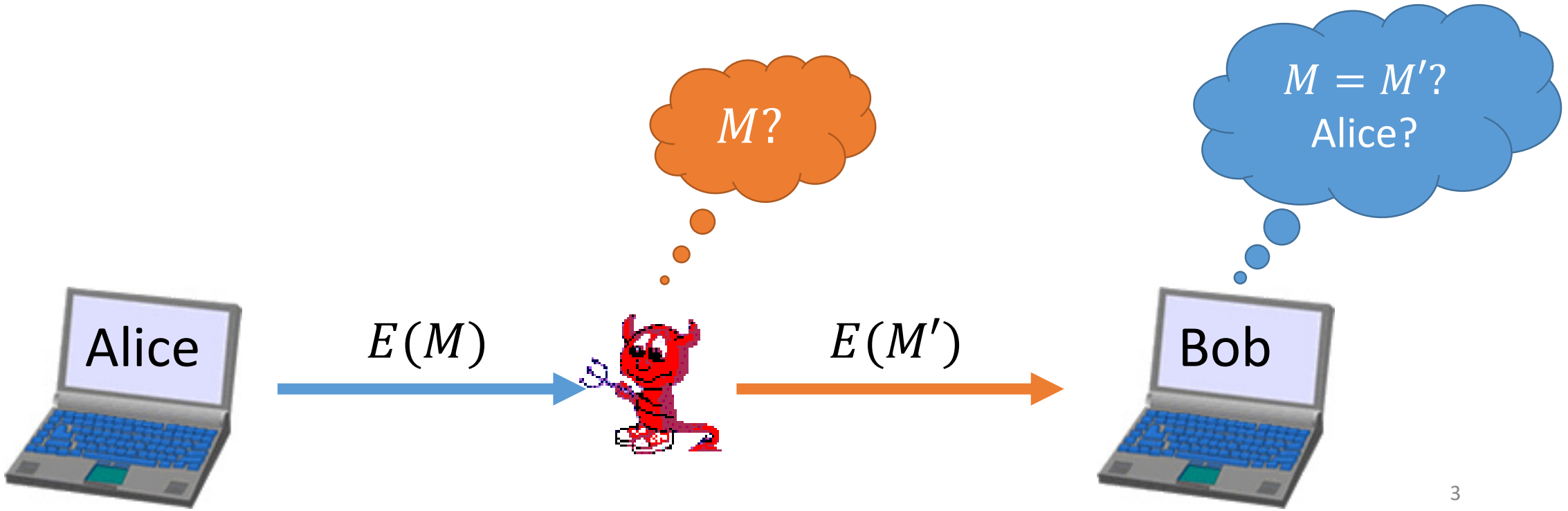
Защита от пассивного противника

- До этого мы рассматривали защиту информации от пассивного противника – противника, который не изменяет сообщения в канале информации



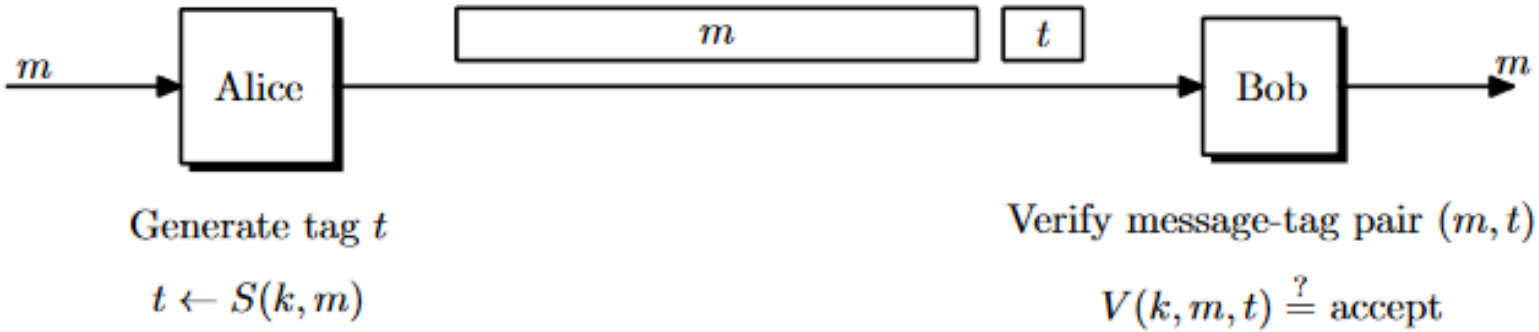
Защита от активного противника

В общем случае задача более сложная – защита от активного противника, который может подменять, изменять и передавать собственные сообщения в канале связи



Целостность сообщений

- Задача – обеспечить целостность сообщений m при передаче
- Обеспечиваем только **целостность**, сообщения предполагаются открытыми
- Основная идея – создать небольшую по длине величину t (tag, метка) на основе сообщения, и передать данную величину вместе с сообщением: (m, t). На стороне получателя величина t' вычисляется для полученного сообщения m' и производится сравнение $t = t'$. В случае равенства полагается, что целостность сообщения не нарушена.



Целостность сообщений

- В данной лекции рассматриваем только защиту целостности
- В дальнейшем в лекциях будем говорить и об обеспечении целостности и конфиденциальности (аутентифицированное шифрование)
- … но даже только обеспечение целостности имеет реальные приложения.
  - Пример – открытое распространение новостей об итогах торгов на бирже. Новости не являются секретными, но мы хотим удостоверится, что была обеспечена их целостность (т.е. их не подменили при передаче). Заметим, что порядок сообщений может быть обеспечен, при обеспечении целостности их нумерации (т.е. защищаем не только целостность сообщений, но их id).
  - Пример – обеспечение целостности дистрибутивов бесплатного программного обеспечения

Обеспечение целостности

- Как построить алгоритм обеспечения целостности?
- Очевидно он должен зависеть от сообщения
- Необходимо использование секретного ключа, неизвестного противнику, так как иначе противник может подменить сообщение и вычислить для него новый tag
- **ВАЖНО** CRC32 и другие помехоустойчивые коды не подходят для решения указанной нами задачи. Задача циклических кодов – обеспечение целостности при защите от случайных изменений, вызванных передачей по каналу связи. Мы пытаемся защититься от преднамеренных изменений, внесённых противником, который может вычислить и CRC32 для произвольных сообщений. Более того, для CRC32 возможно эффективное построение коллизий.

Определение MAC

Введём определение кода аутентичности сообщения (MAC, message authentication code, имитовставка).

MAC на (K, M, T) называется пара эффективных алгоритмов $I = (S, V)$. S – алгоритм выработки MAC, V – алгоритм проверки MAC. Пусть M – множество сообщений, K – множество ключей, T – множество кодов аутентичности (меток). Тогда для $m \in M, t \in T, k \in K$

- $S: K \times M \to T$ - вероятностный алгоритм, вычисляющий $t \leftarrow^R S(k, m)$
- $V: K \times M \times T \to \{0,1\}$ – детерминированный алгоритм, вычисляющий результат проверки $r \leftarrow V(k, m, t)$.
- Свойство корректности - $\Pr\left[V\left(k, m, S(k, m)\right) = 1\right] = 1$

Детерминированный MAC

- Если функция S – детерминированная, то для любой такой функции мы можем ввести функцию

$$V(k, m, t) = \begin{cases} 1, S(k, m) = t \\ 0, S(k, m) \neq t \end{cases}$$

Очевидно, что полученный MAC обладает свойством корректности и называется детерминированным MAC. Т.е. для фиксированного ключа он выдает одинаковый код аутентичности для одинаковых сообщений.

- Если функция S – рандоминизованная то MAC называется рандомизированным.

Стойкий MAC

Введём понятие стойкости MAC.

Возможности противника – выбор сообщений для получения MAC для них

Цель противника – получения новой верной пары сообщение-MAC

Стойкий MAC – MAC не позволяющий противнику получить такую пару

Построение MAC на основе PRF

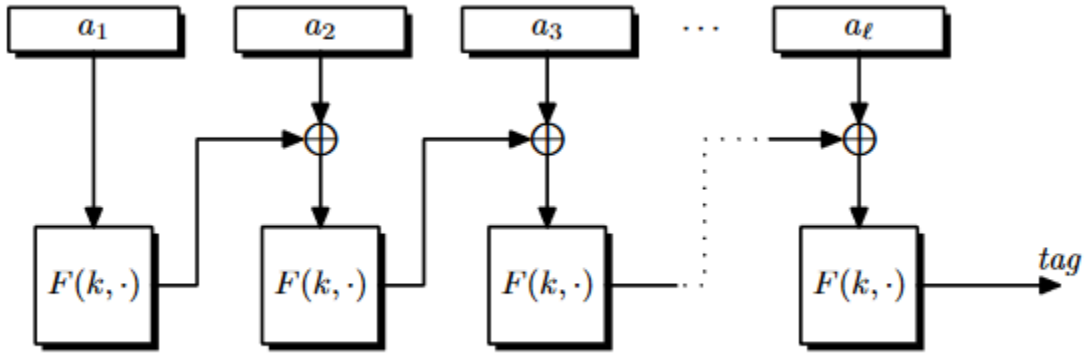
Любая стойкая PRF с суперполиномиальной областью значений является стойким MAC.

Проблема – рассмотренные ранее PRF имеют фиксированных вход (например размер блока в случае блочного шифра). Мы же ходим получать MAC для сообщений произвольной длины.

Хотим получить аналог «режимов шифрования» для коротких PRF, позволяющих вычислять MAC для произвольных сообщений

Беспрификсные PRF

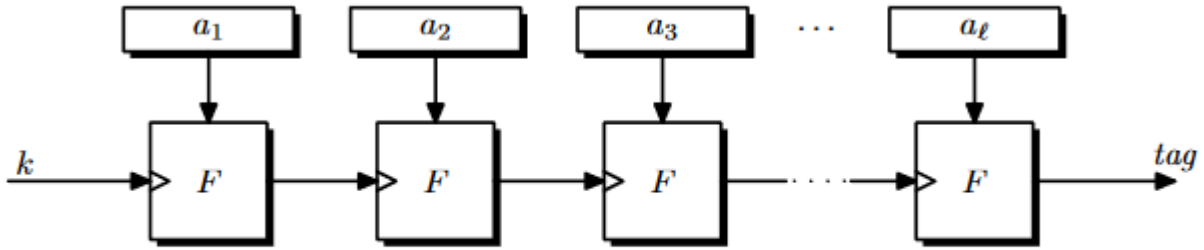
PRF $F_{CBC}(k, m)$ – цепочка CBC с использованием PRF. В качестве значение используется последний элемент цепочки.



(a) The CBC construction $F_{\text{CBC}}(k, m)$

Беспрификсные PRF

PRF $F^*(k, m)$ – каскадная конструкция. Выход каждой итерации PRF используется к качестве ключа в следующей итерации PRF.



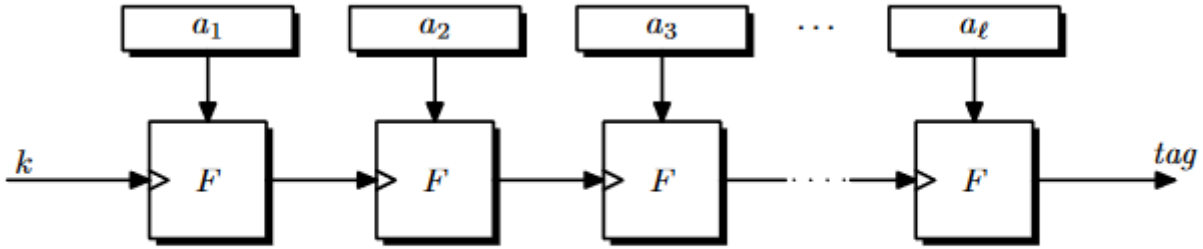
(b) The cascade construction $F^*(k, m)$

Figure 6.3: Two prefix-free secure PRFs

Атака на F^* MAC

Пусть F^* MAC на основе беспрификсной PRF.

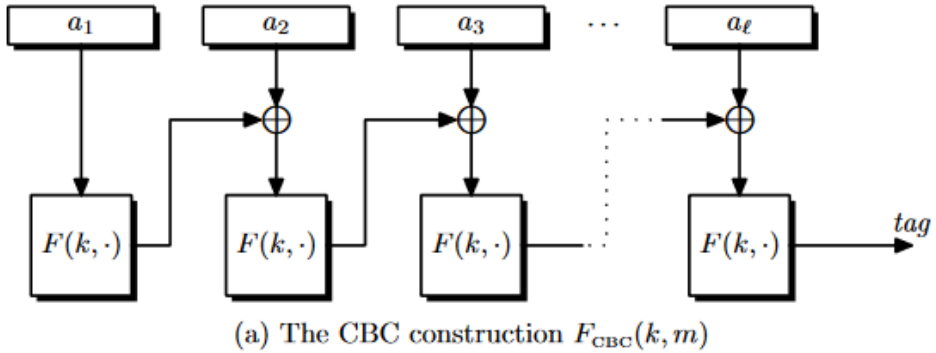
Для фиксированного сообщения $m \in X^{\leq l}$ MAC $t = F^*(k, m)$ и произвольного сообщения m' можно получить : $t' = F^*(k, m||m')$ без знания ключа, т.е. возможно осуществить атаку на MAC.



(b) The cascade construction $F^*(k, m)$

Figure 6.3: Two prefix-free secure PRFs

Атака на F_{CBC} MAC



(a) The CBC construction $F_{\text{CBC}}(k, m)$

Пусть F_{CBC} MAC на основе CBC. Построим атаку.

- Выберем произвольный $a_1 \in X$
- Запросим MAC t для сообщения a_1
- Вычислим $a_2 = a_1 \oplus t$. Тогда t является корректным MAC для сообщения (a_1, a_2)

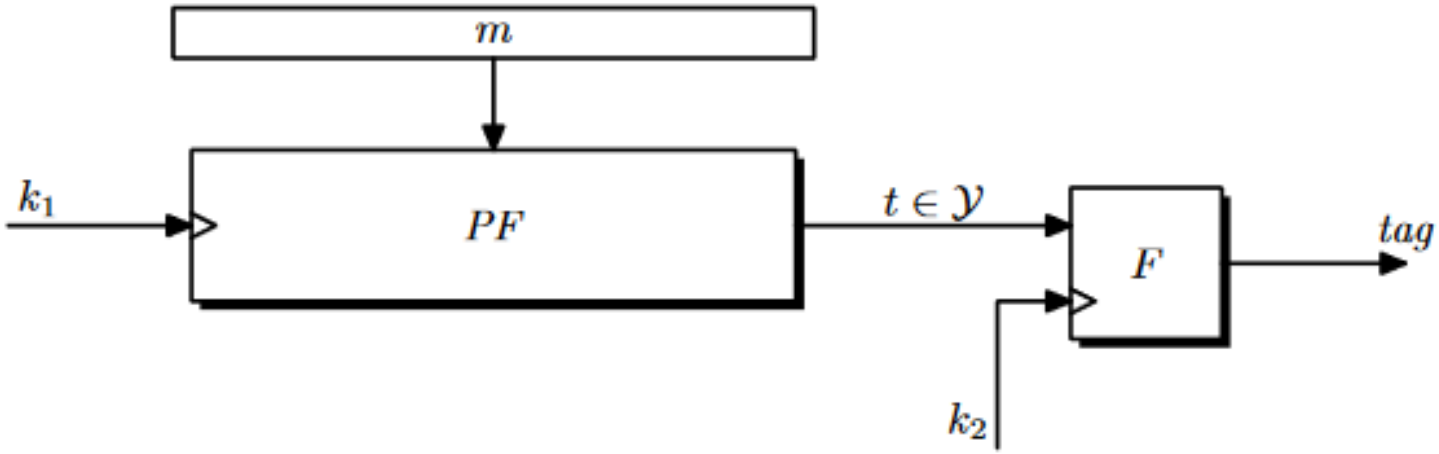
$$t = F(k, a_1), a_1 = F(k, a_1) \oplus a_2$$

$$F_{CBC}\big(k, (a_1, a_2)\big) = F(k, F(k, a_1) \oplus a_2) = F(k, a_1) = t$$

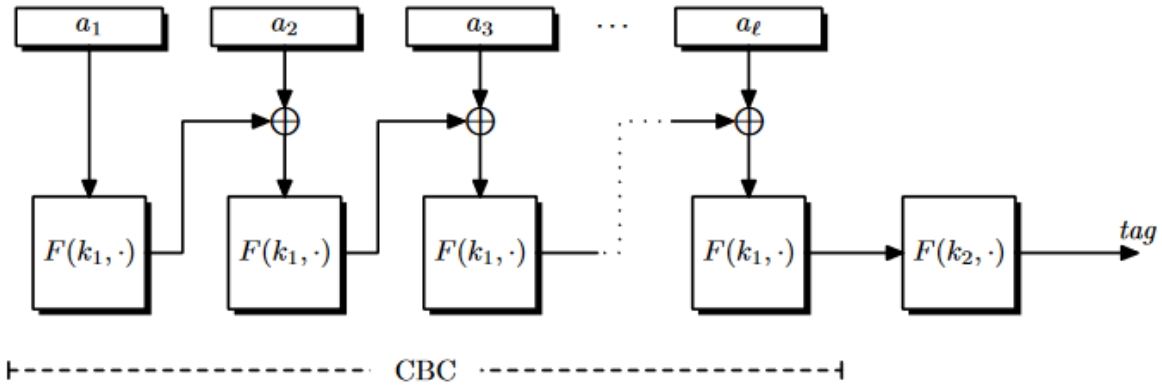
Зашифрование выхода беспрификсной PRF

Пусть PF – PRF:$K_1 \times X^{\leq l} \to Y$, F – PRF: $K_2 \times Y \to T$.

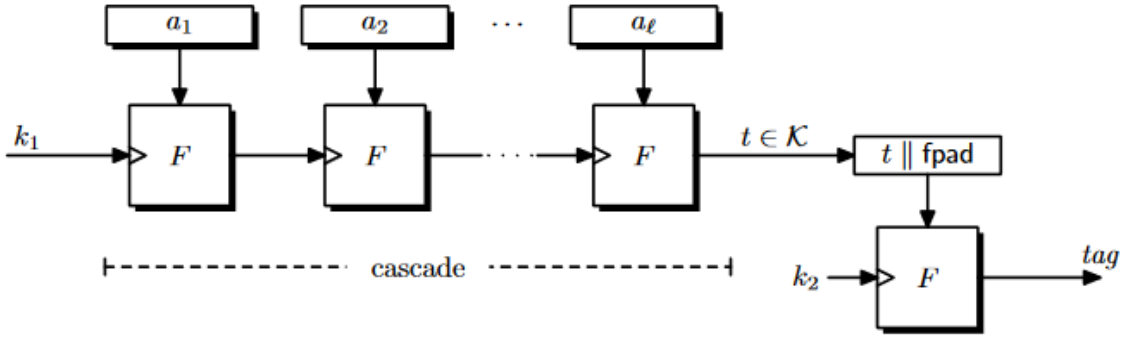
Определим $EF\big((k_1, k_2), m\big) = F\big(k_2, PF(k_1, m)\big), k_1 \in K_1, k_2 \in K_2, m \in X^{\leq l}$



Зашифрование выхода беспрификсной PRF

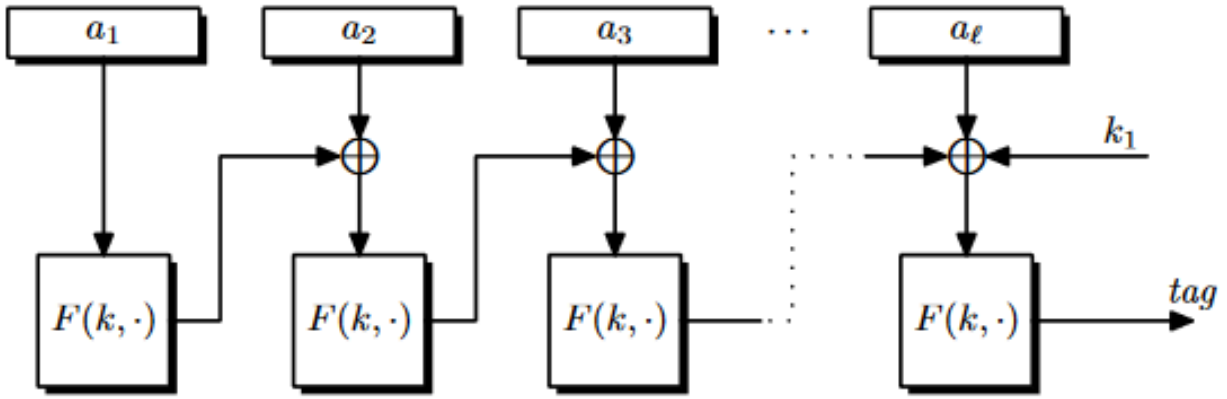


(a) The ECBC construction $\mathrm{ECBC}(k,m)$ (encrypted CBC)

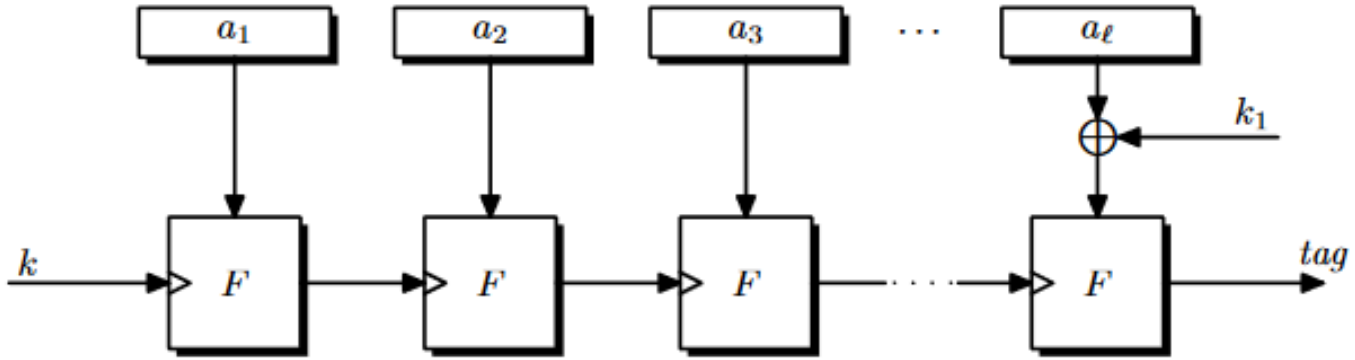


(b) The NMAC construction $\mathrm{NMAC}(k,m)$ (encrypted cascade)

Беспрификсное кодирование с рандомизацией



(a) *rpf* applied to CBC



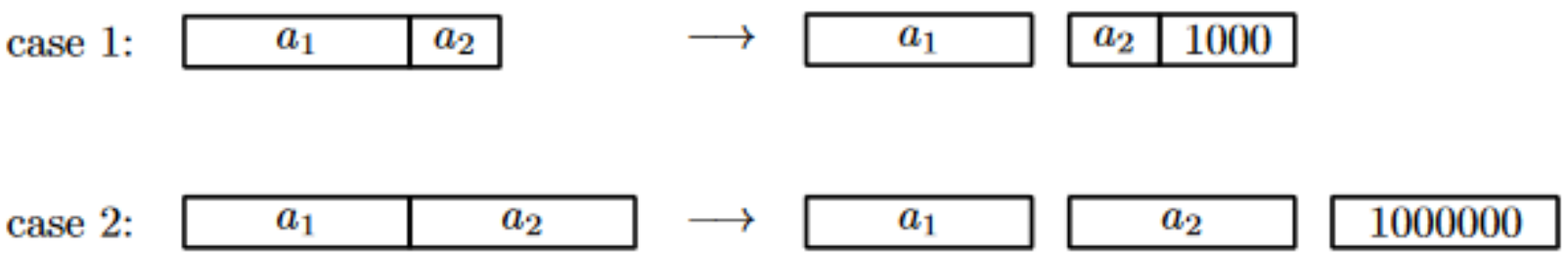
(b) *rpf* applied to cascade

Построение инъективных функций

Пусть $X = \{0,1\}^n, inj: \{0,1\}^{\leq nl} \to X^{\leq l+1}$

inj:

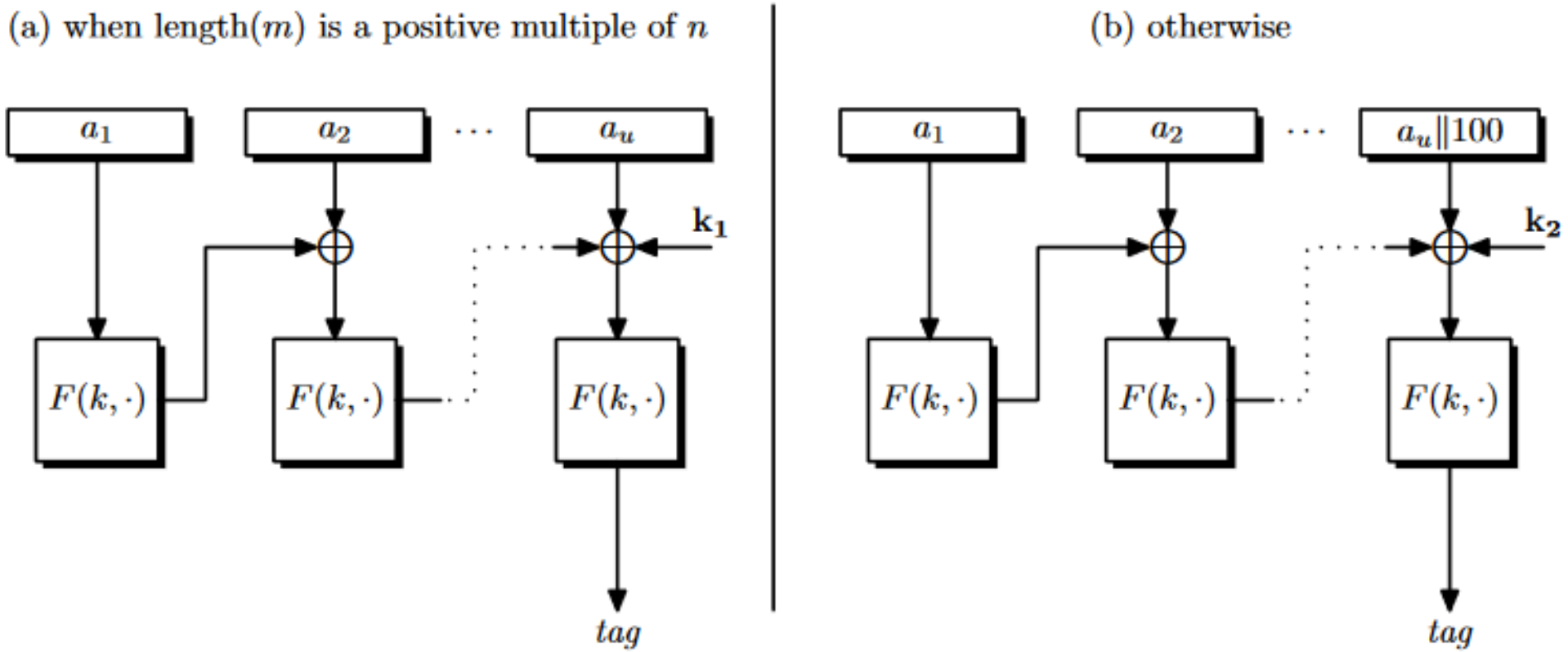
- Если входное сообщение имеет длину не кратную n – добавить 10…00 до длинны кратной n
- Иначе – добавить n-блок $(1||0^{n-1})$
- Инъективна и обратима



CMAC

- Стандарт NIST
- Один из наиболее популярных алгоритмов вычисления MAC (самый популярных после HMAC)
- Использует три различных ключа (могут быть выработаны на основе одного ключа)

CMAC


(a) when length(m) is a positive multiple of n
a_1
a_2
a_u
$\mathbf{k_1}$
$F(k,\cdot)$
$F(k,\cdot)$
$F(k,\cdot)$
tag
(b) otherwise
a_1
a_2
$a_u \| 100$
$\mathbf{k_2}$
$F(k,\cdot)$
$F(k,\cdot)$
$F(k,\cdot)$
tag

OMAC

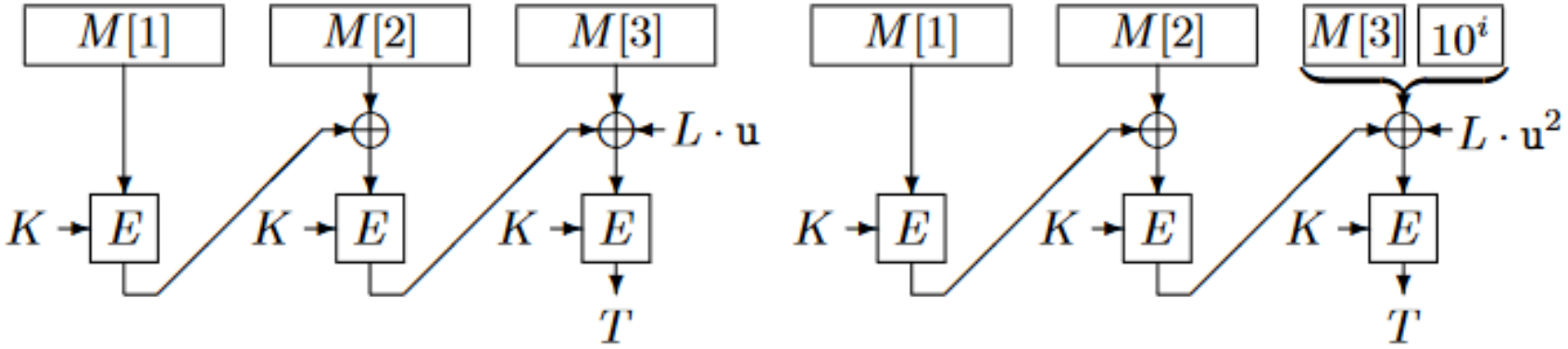
- В текущей вариации (OMAC) использует единственный ключ для генерации этих трех ключей для некоторой константы R_n:

input: key $k \in \mathcal{K}$
output: keys $k_0, k_1, k_2 \in \mathcal{X}$

$k_0 \leftarrow k$
$L \leftarrow F(k, 0^n)$
(1) if $\mathrm{msb}(L) = 0$ then $k_1 \leftarrow (L \ll 1)$ else $k_1 \leftarrow (L \ll 1) \oplus R_n$
(2) if $\mathrm{msb}(k_1) = 0$ then $k_2 \leftarrow (k_1 \ll 1)$ else $k_2 \leftarrow (k_1 \ll 1) \oplus R_n$
output k_0, k_1, k_2.

OMAC

- Фактически для получения трех ключей реализуется умножение в кольце многочленов на некоторую константу u



Trunkated CBC MAC

Основная идея – не дать противнику возможность воспользоваться MAC для осуществления префиксной атаки.

Использование части кода аутентичности. Используется в ГОСТ 28147-98

Оптимально использовать половину исходного MAC

Основной недостаток – фактически понижаем параметр стойкости в 2 раза

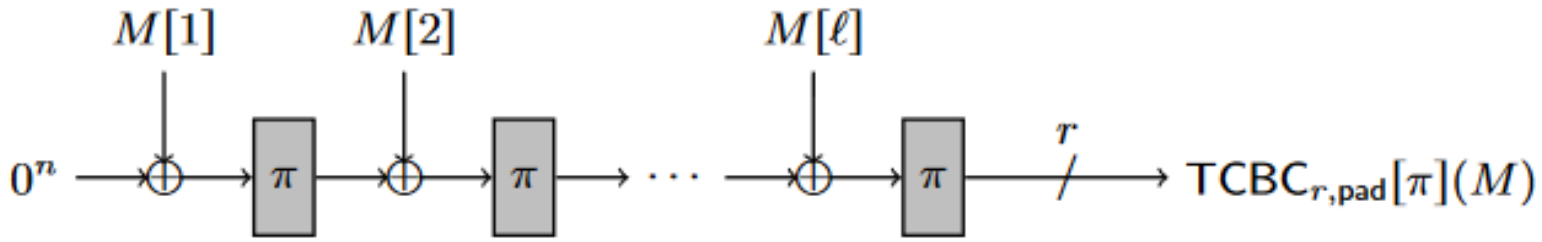
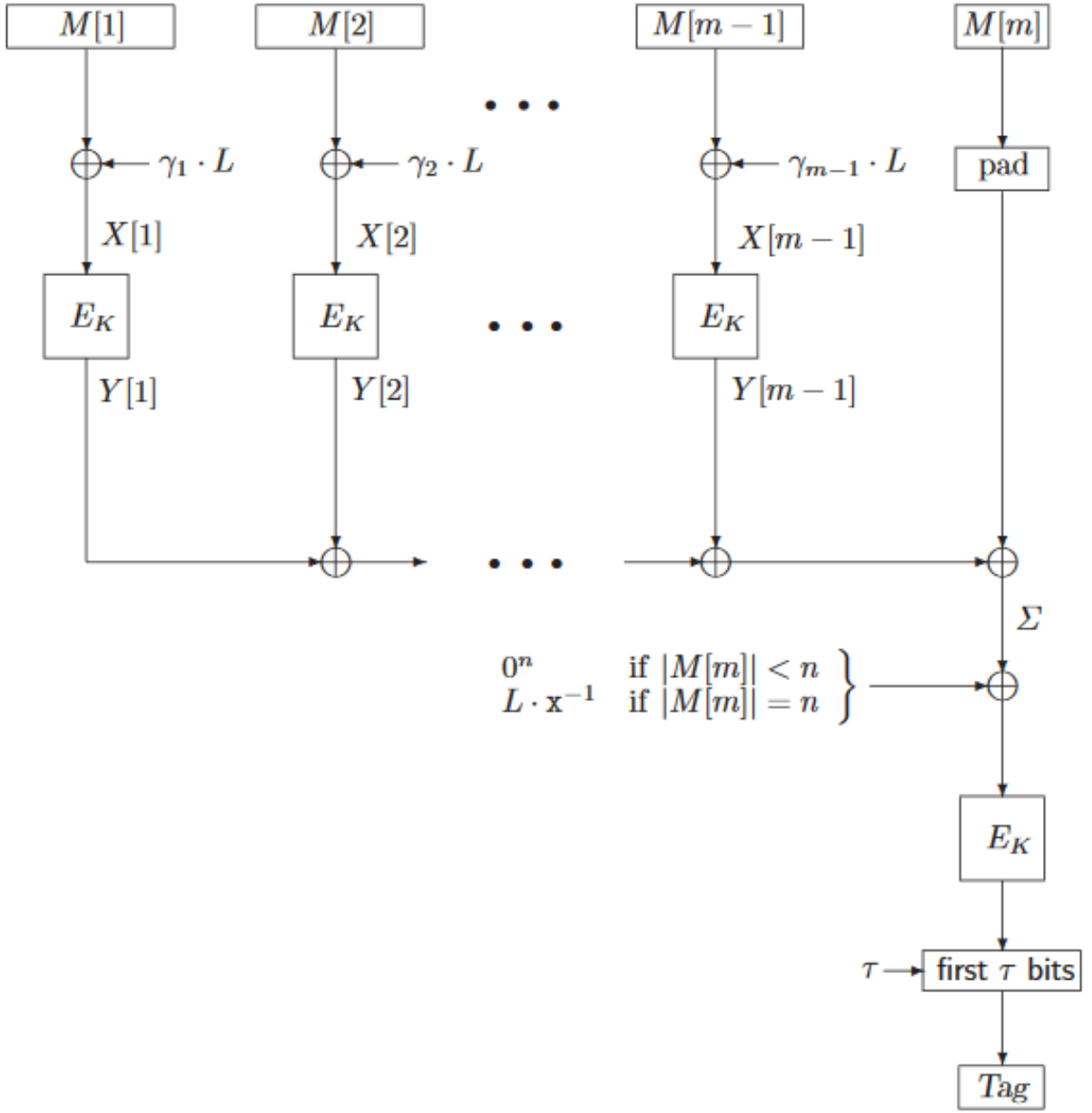


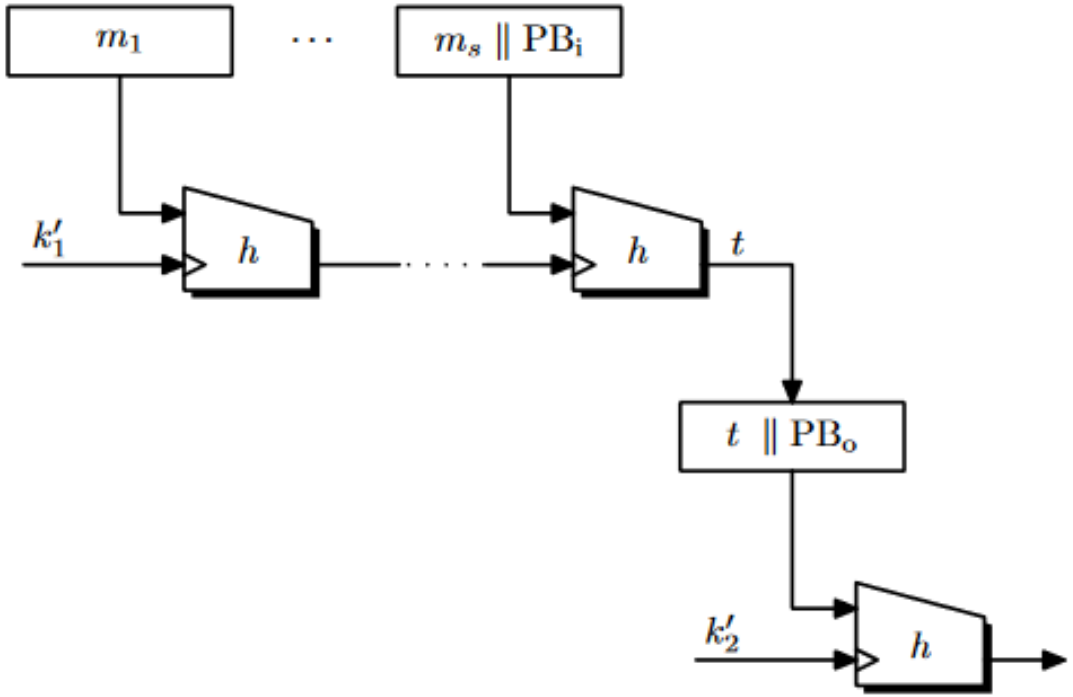
Fig. 1. Truncated CBC. Representation of $\mathsf{TCBC}_{r,\mathsf{pad}}[\pi]$. Here, $M[1], \ldots, M[\ell]$ are n-bit blocks resulting from applying the padding scheme pad to the input message $M \in \{0,1\}^*$.

PMAC


$M[1]$
$M[2]$
$M[m-1]$
$M[m]$
$\gamma_1 \cdot L$
$\gamma_2 \cdot L$
$\gamma_{m-1} \cdot L$
pad
$X[1]$
$X[2]$
$X[m-1]$
E_K
E_K
E_K
$Y[1]$
$Y[2]$
$Y[m-1]$
Σ
0^n if $|M[m]| < n$
$L \cdot \mathrm{x}^{-1}$ if $|M[m]| = n$
E_K
τ
first τ bits
Tag

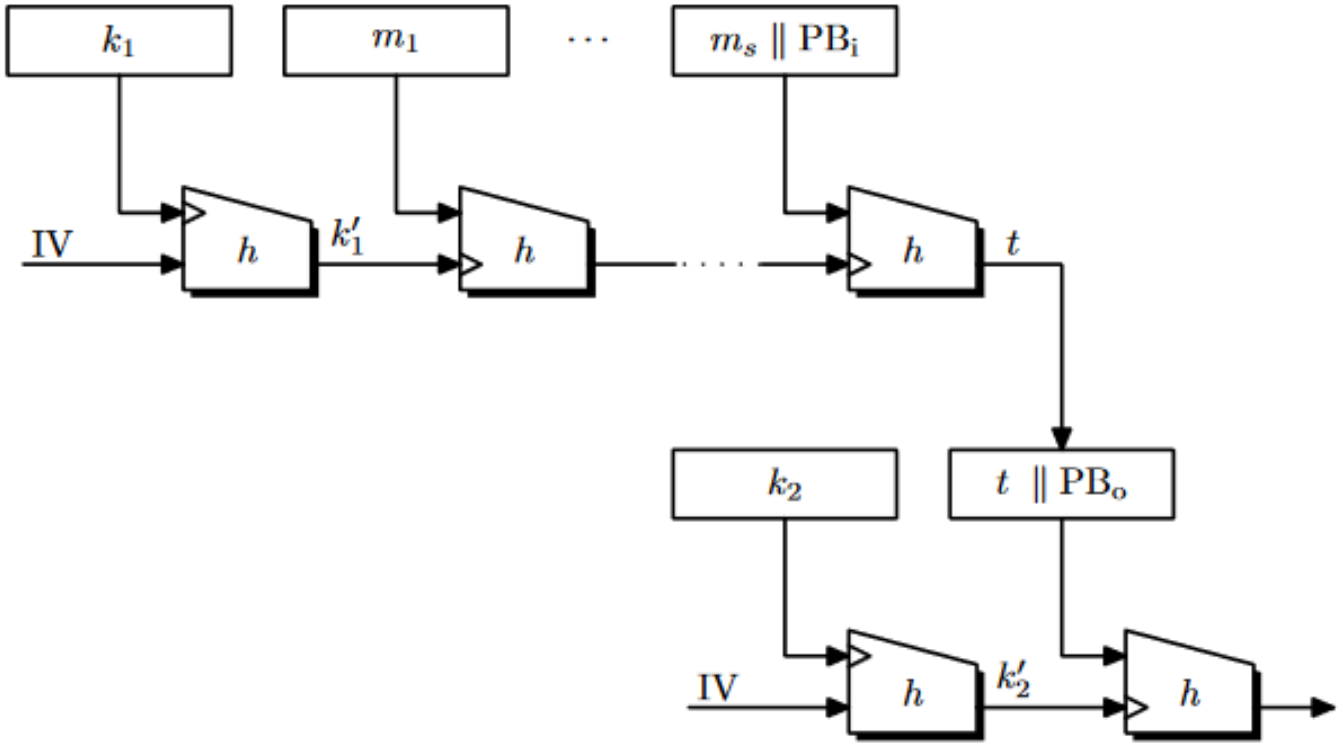
NMAC

- Заменим F на итеративную хэш-функцию хэш-функцию. Получим:



NMAC

Реализуем алгоритм получения ключей k_1' и k_2' с помощью IV и k_1 и k_2



HMAC

Уберём независимость ключей k_1 и k_2:

- $k_1 \leftarrow k \oplus ipad$
- $k_2 \leftarrow k \oplus opad$

Где

- $ipad = (0\text{x}36, 0\text{x}36, \ldots, 0\text{x}36)$
- $opad = (0\text{x}5\text{c}, 0\text{x}5\text{c}, \ldots, 0\text{x}5\text{c})$

Итого:

$$HMAC(k, m) = H(k \oplus opad \| H(k \oplus ipad, m))$$

HMAC

- Де-факто интернет стандарт
- Не требует блочного шифра для реализации, основан на хэш-функции
- Используется во множестве протоколов
- Самый распространённых MAC
- Может быть построен с использованием произвольной хэш-функции (включая ГОСТ)
- В настоящий момент используется HMAC-SHA-256
- Лучше избегать использование HMAC-SHA-1, хотя в настоящий момент не известны практические атаки, существенно лучше перебора