v. 2.0

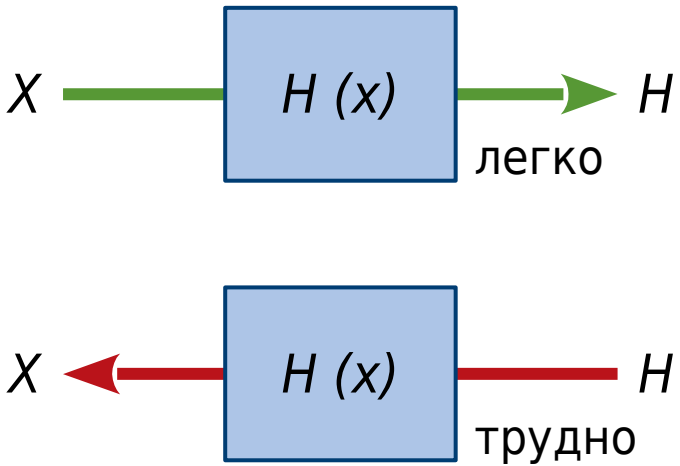
Криптография

Лекция 3. Хеш-функции.

Дмитрий Яхонтов

“Кочерга”, 2019

Хеш-функция (свёртка, отпечаток, дайджест)



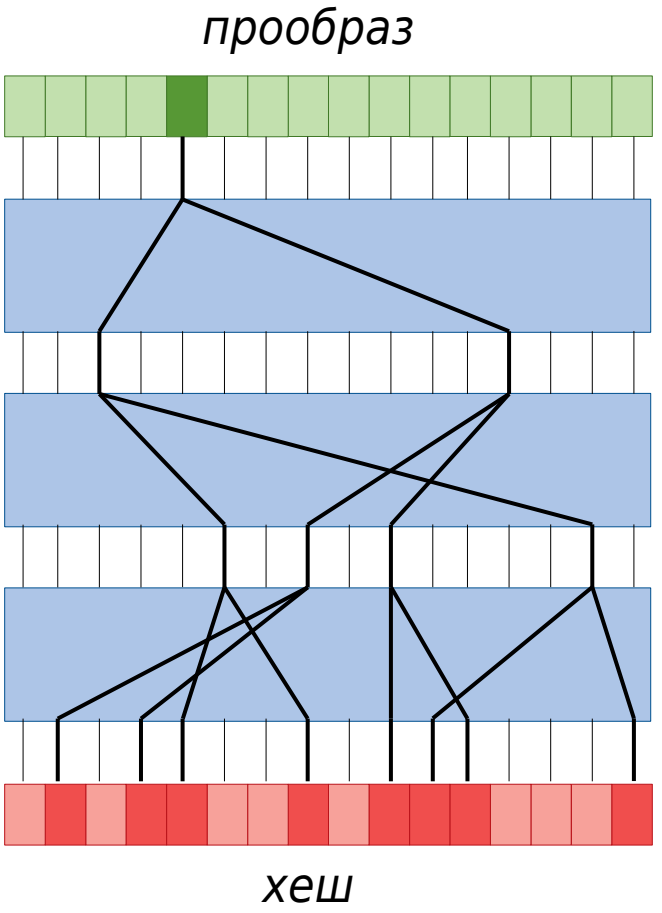
Хеш-функция необратимо преобразует набор данных *X* произвольной длины в хеш *H* фиксированной длины.

Применения:

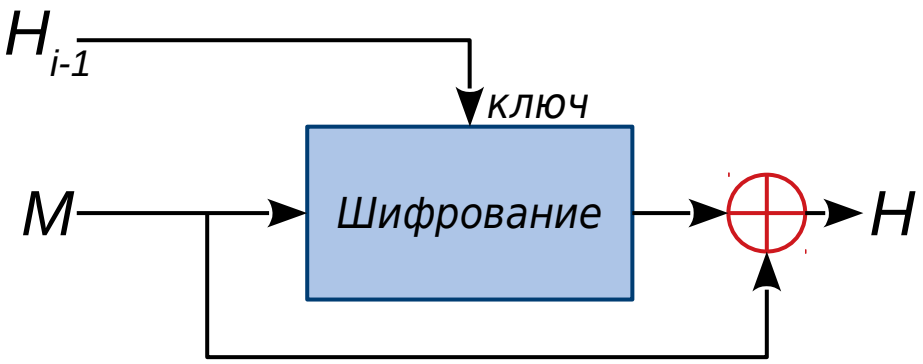
- контроль целостности сообщений
- цифровые подписи:
  вместо всего сообщения
  достаточно подписать его хеш
- проверка данных без хранения образца
- доказательство работы (proof-of-work)

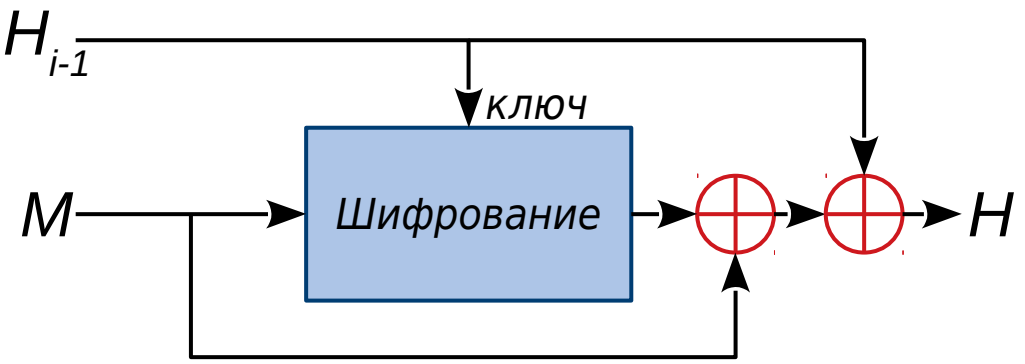
Требования к хеш-функциям

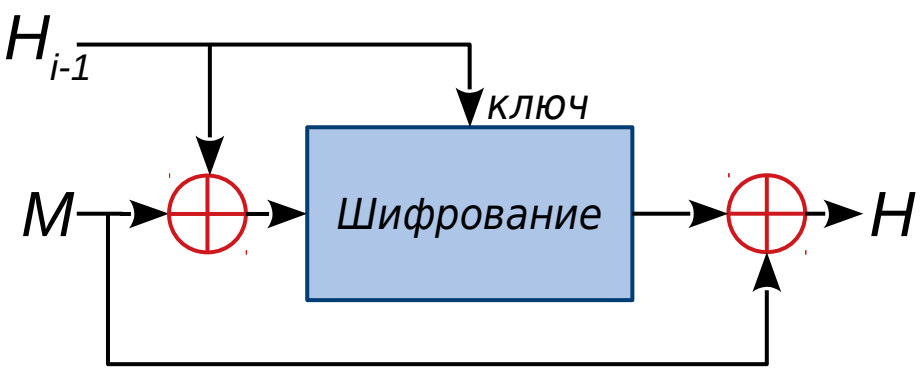
- Необратимость
  нельзя восстановить сообщение
  по значению его хеша.
- Стойкость к поиску второго прообраза
  зная сообщение и его хеш, нельзя подобрать
  другое сообщение с тем же хешем.
- Стойкость к коллизиям
  нельзя подобрать пару сообщений
  с одинаковыми хешами.
- Стойкость к удлинению прообраза
  нельзя дополнить сообщение так,
  чтобы получить нужный хеш.

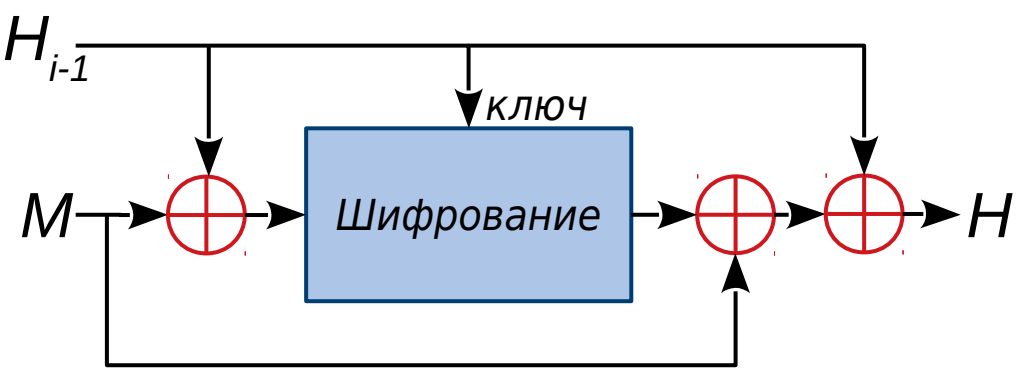


На основе симметричных шифров



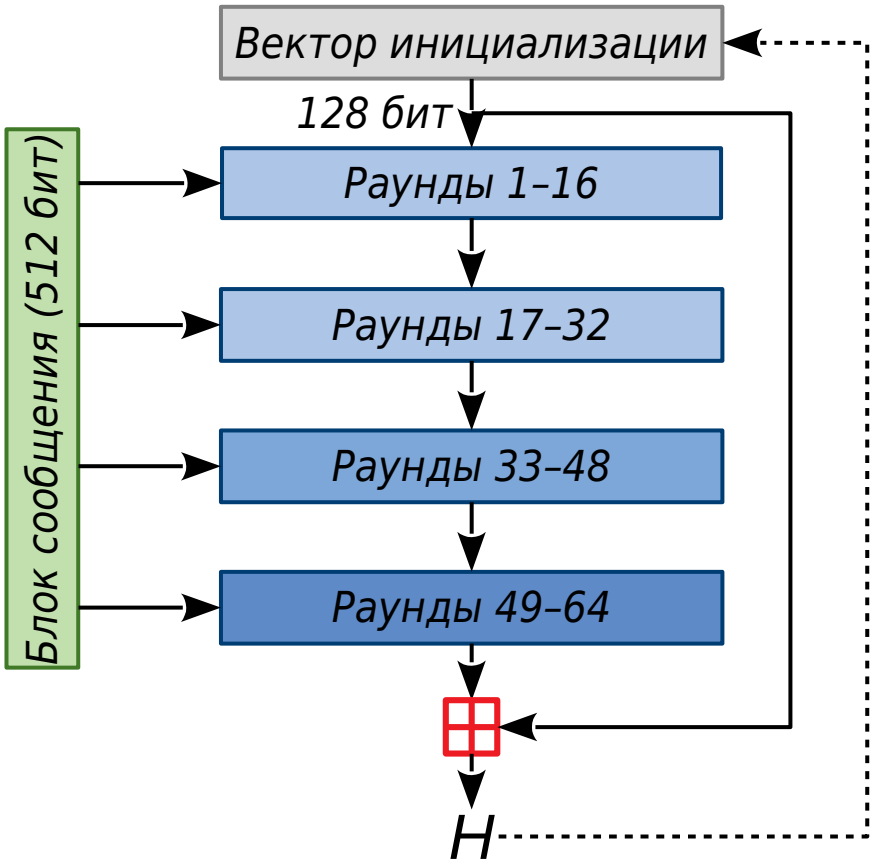






Структура Меркла—Дамгора (Merkle—Damgård)

на примере MD5



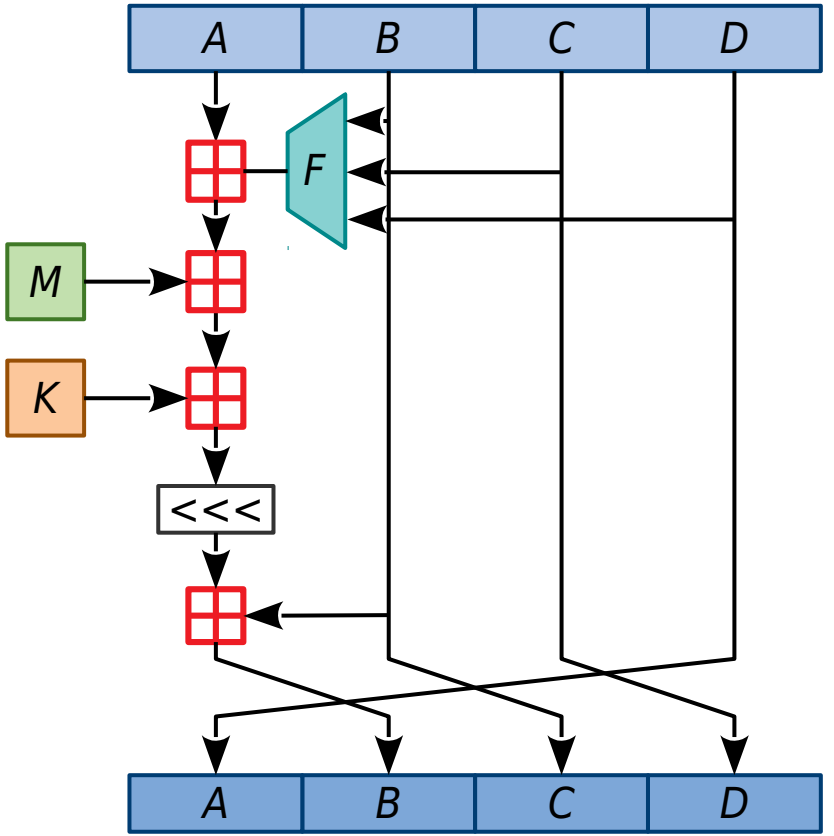
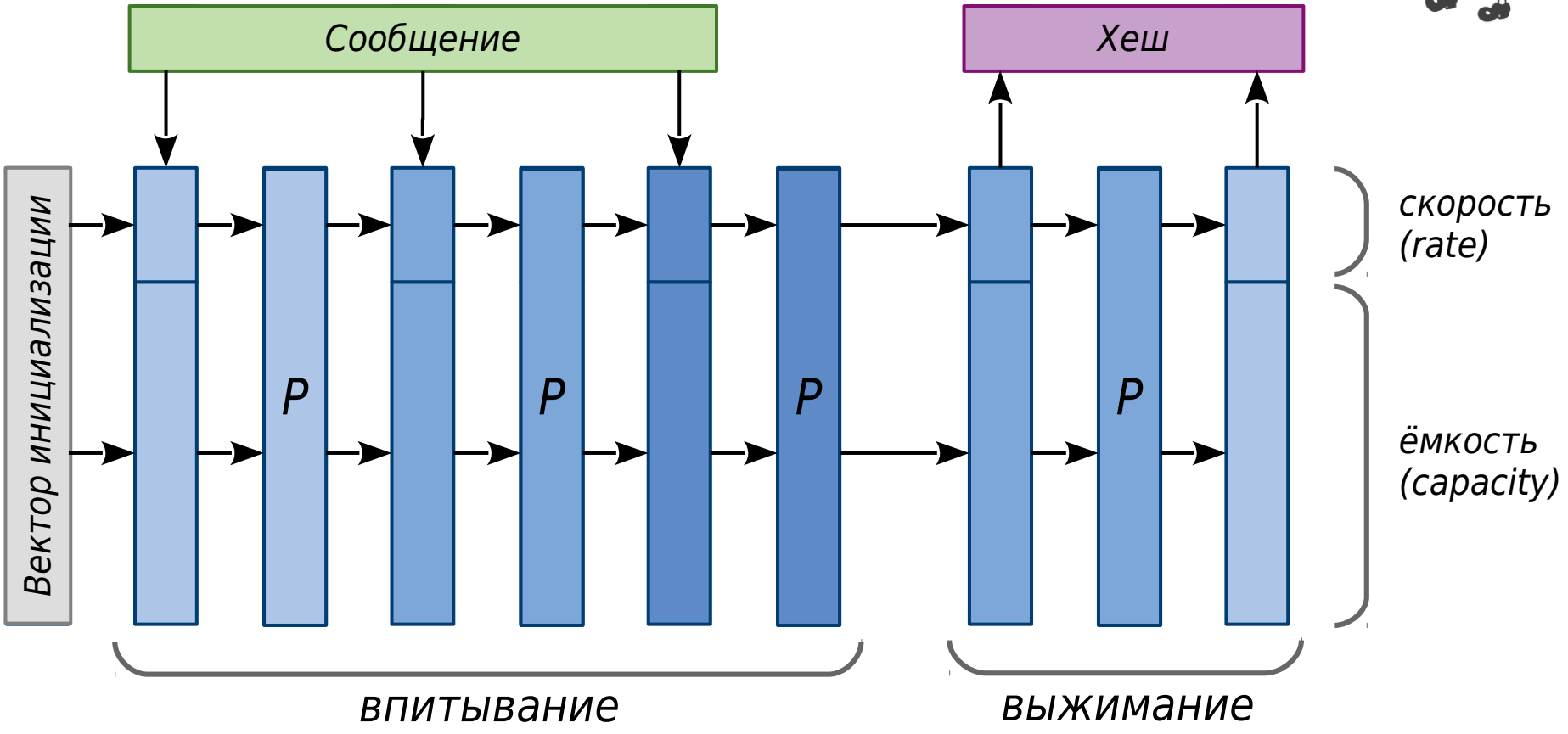


схема одного раунда

Криптографическая губка



название хеш-функции	год	число раундов	размер блока	размер хеша
MD2	1989	18	128	128
MD4	1990	48	512	128
MD5	1991	64	512	128
MD6	2008	40 – 80	4096	128 – 512
RIPEMD	1996	64 / 80	512	128 / 160 / 256 / 320
SHA-1	1995	80	512	160
SHA-2	2002	64 / 80	512 / 1024	224 / 256 / 384 / 512
SHA-3 (Keccak)	2008	24	1600	224 / 256 / 384 / 512
Стрибог (ГОСТ Р 34.11-2012)	2012	12	512	256 / 512

Коды проверки подлинности
(MAC — Message Authentication Code)

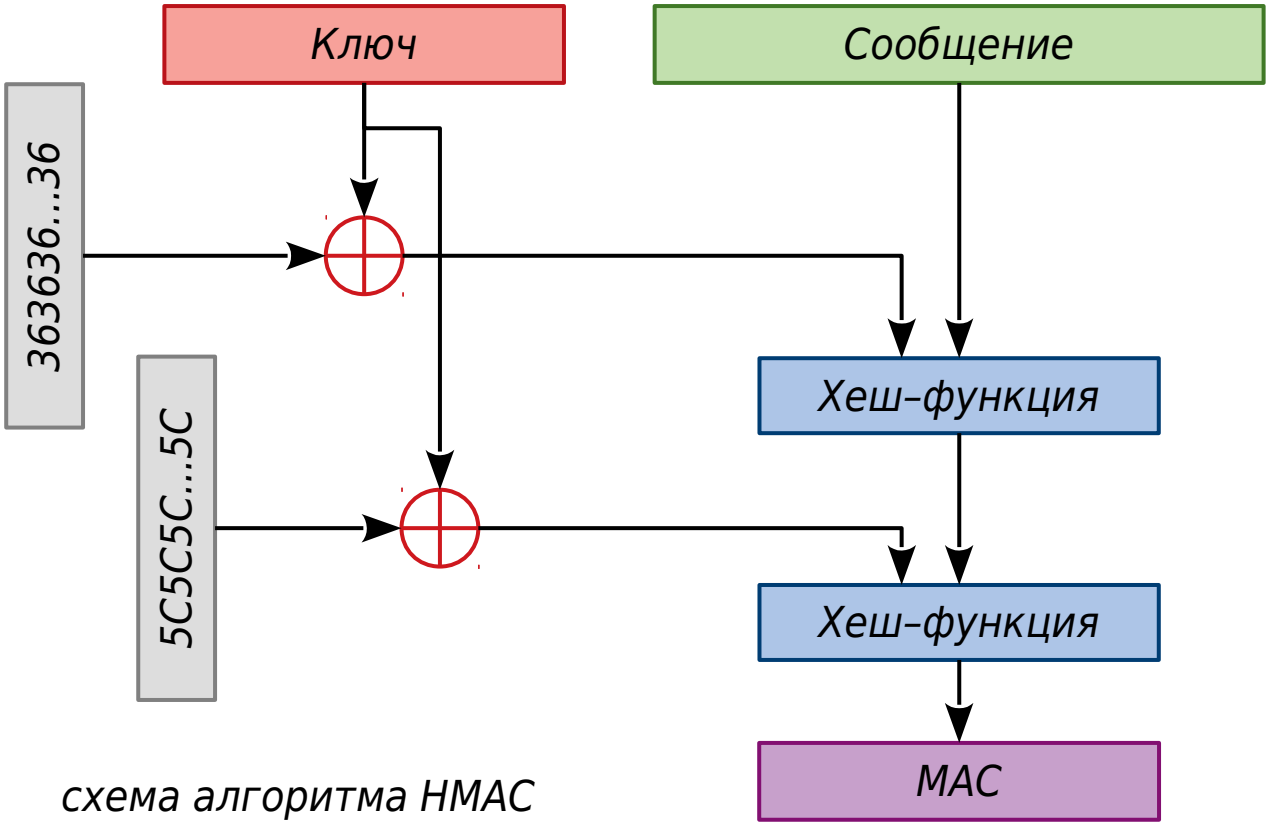
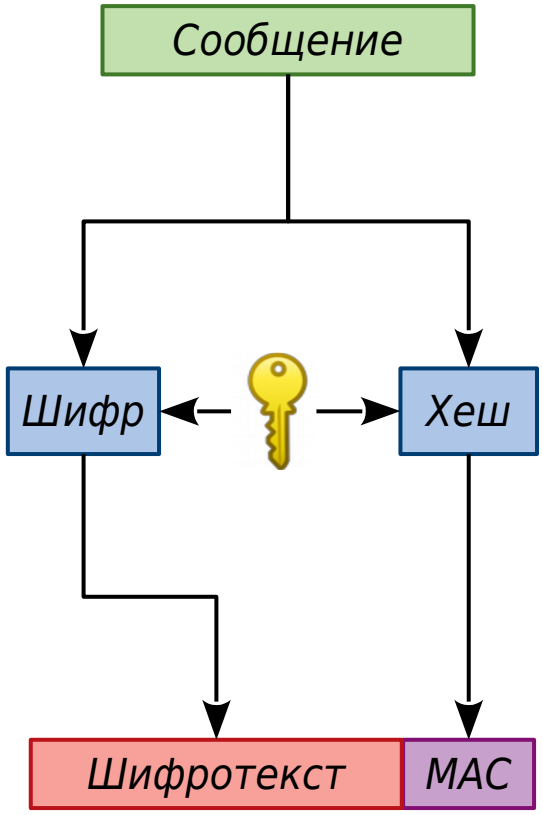


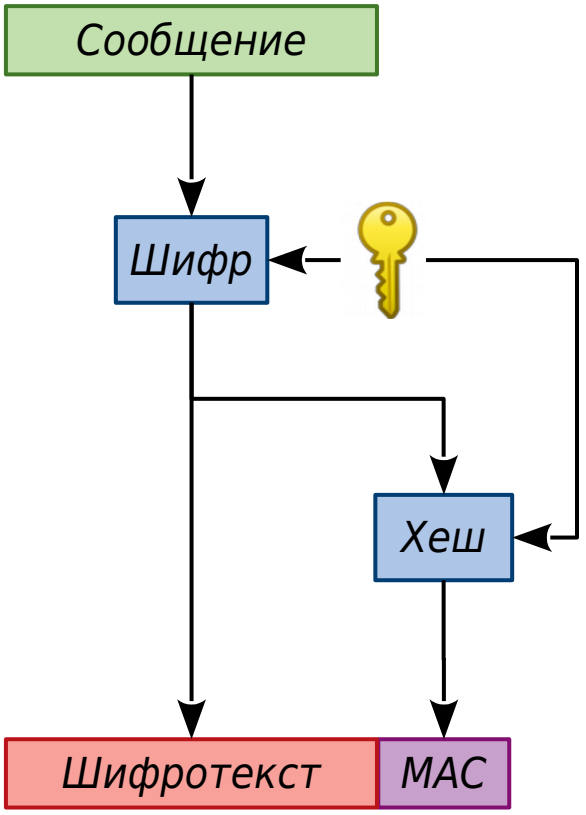
схема алгоритма HMAC

MAC служит для защиты сообщения от изменеий и контроля целостности.

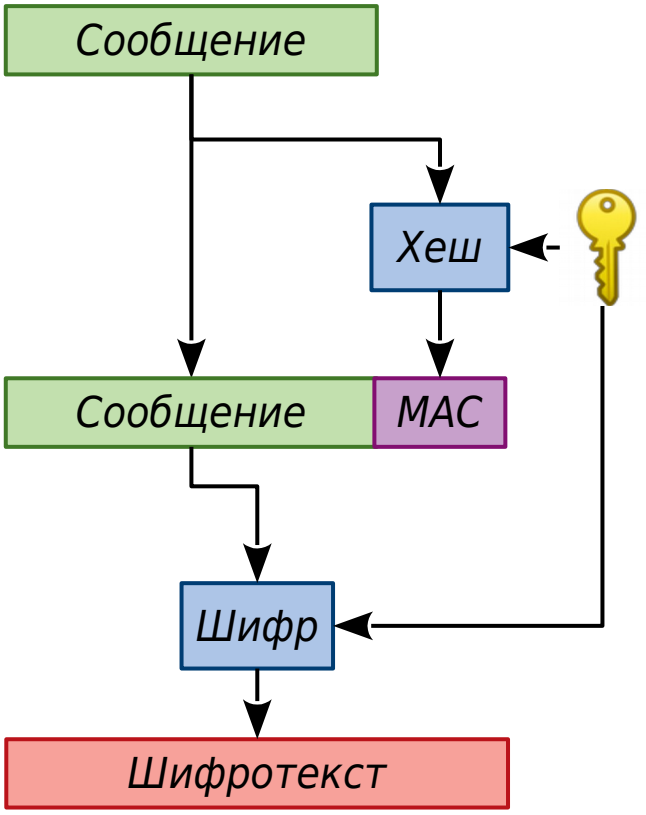
Для генерации и проверки используется секретный ключ.



Encrypt–and–MAC



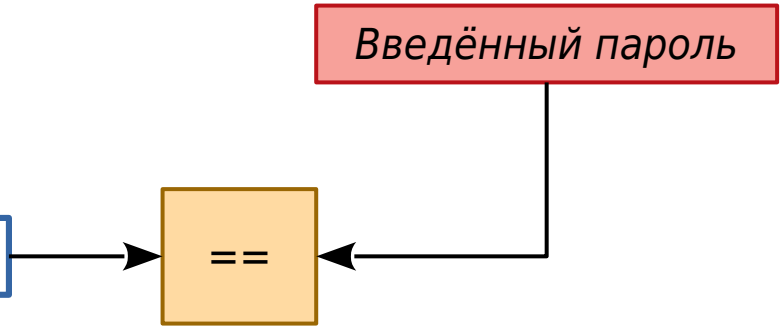
Encrypt–then–MAC



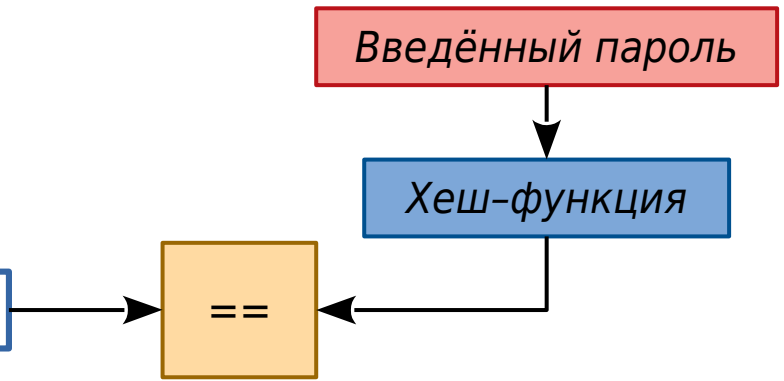
MAC–then–Encrypt

Хранение и проверка паролей

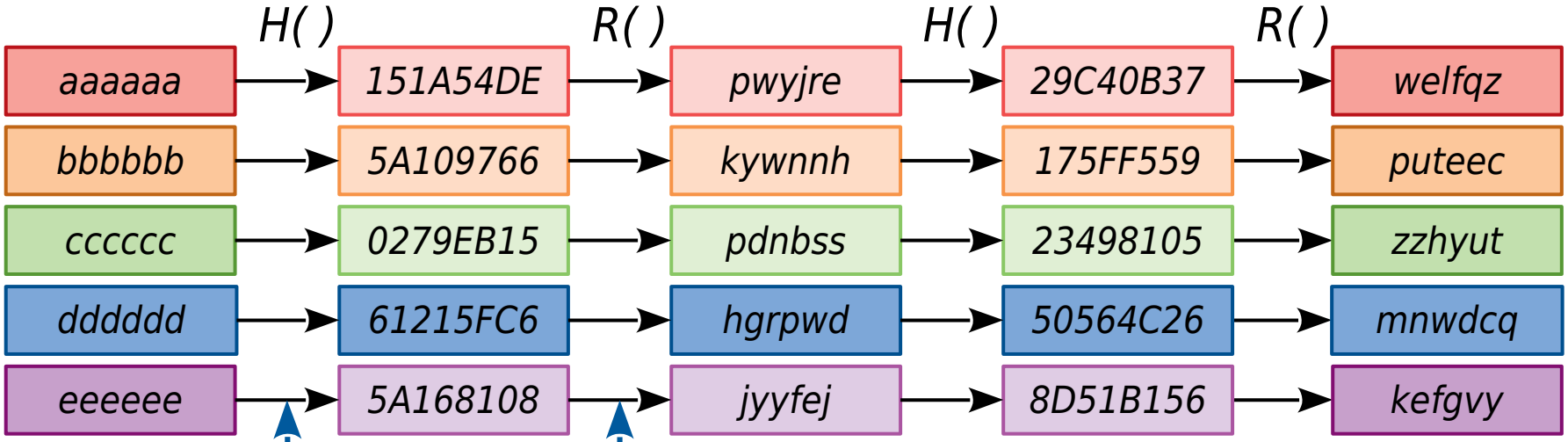
логин	пароль
chingiz	40000MoNkEyS
padla	pere$%*za&$#na
pat	40000MoNkEyS
strelok	74606247WHOds



логин	хеш пароля
chingiz	1A1B612107856C81C8DF311174E766C1
padla	ECDAC781EFBC47BB0571DC73CDA50AE3
pat	1A1B612107856C81C8DF311174E766C1
strelok	66DB5F9AB5EBEF2D0417741FD75D021D

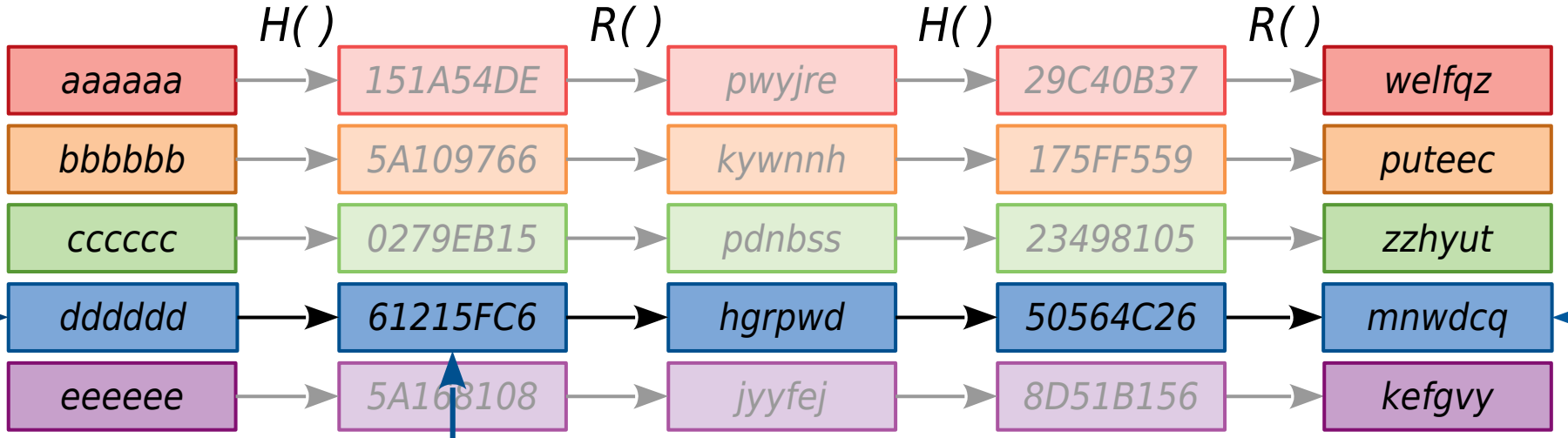


Радужные таблицы
(Rainbow Tables)



- Хеш–функция
- Функция редукции
  (каждому хешу однозначно ставит в соответствие какую–либо строчку)
- Можно хранить только начальные и конечные элементы цепочек

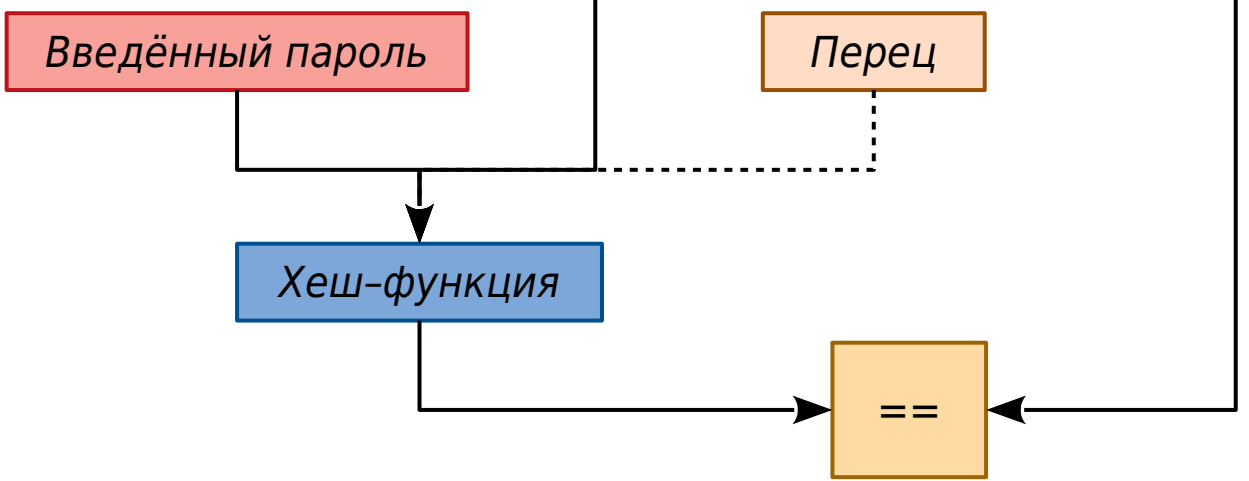
Как пользоваться радужной таблицей



- Хеш *61215FC6*
- Строим цепочку *61215FC6 → hgrpwd → 50564C26 →* *mnwdcq*
- По последнему элементу находим строку таблицы, восстанавливаем её
- Находим в строке хеш, восстанавливаем прообраз

Соль и перец

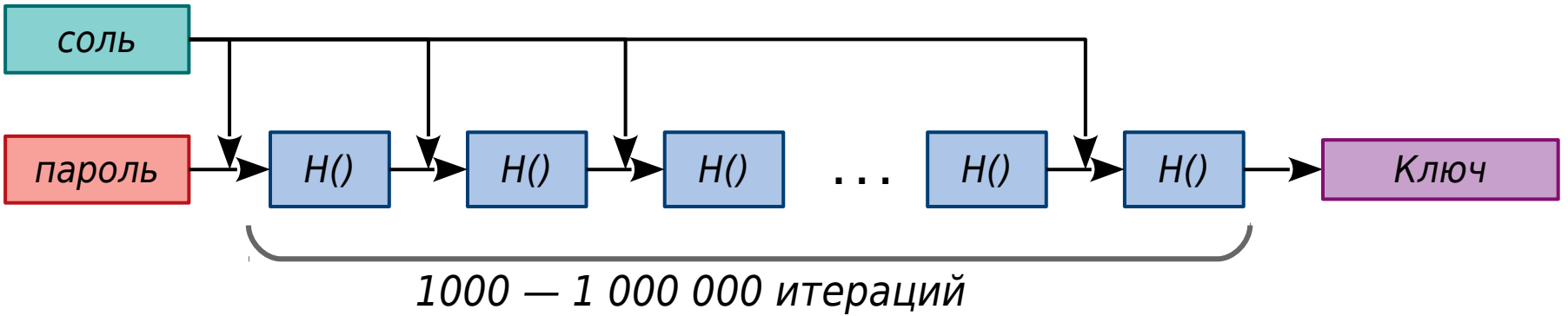
логин	соль	хеш (пароль + соль)
chingiz	6FA1C5E911	C1B408F36309AA830099A3B491BF8E5B
padla	11EDE09DB6	0754B72042E7876D6640EC0726A71653
pat	EA8DA196B6	560BF30473FFD653FA6FC0A2FF06B4E9
strelok	A87AA3794B	94F9E99CD2C5D4923CDB9F5F3F5517B1



Соль случайная, не секретная, уникальная для каждого пользователя.

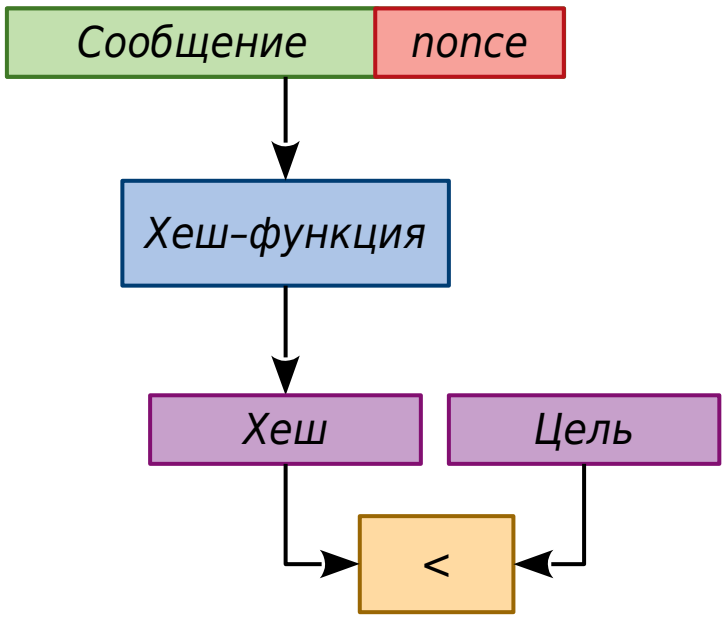
Перец случайный, секретный, хранится отдельно от базы хешей

Функция формирования ключа
(KDF — Key Derivation Function)



- Получение ключа заданной длины и формата
- Получение нескольких независимых ключей из одного пароля
- Защита от атаки перебором:
  многократное хеширование увеличивает время вычисления ключа, снижая скорость перебора до неприемлемо низкой

Доказательство выполнения работы
(Proof–of–Work)



Вычислительная задача: подобрать ***nonce*** так, чтобы хеш ***H(сообщение + nonce)*** получился меньше целевого значения (медленно)

Проверка: однократное вычисление ***H(сообщение + nonce)*** и сравнение с целевым значением (быстро)

- Защита от спама (Hashcash, Bitmessage)
- Генерация блоков (криптовалюты)

Задачи

1. Перед вычислением хеш-функции сообщение необходимо дополнить до длины, кратной размеру блока. Самый простой способ — дополнить нулевыми битами.

   Чем этот способ плох и как его можно улучшить?

2. Четверо хакеров спорят, как назвать их группу.
   Каждый предлагает свой вариант и считает его единственно правильным. Решено выбрать случайным образом.
   Нет независимой стороны, которая помогла бы провести жеребьёвку, а каждый хакер доверяет только себе.

   Предложите протокол, который позволит сделать честный случайный выбор в пользу одного из участников.

3. Алиса и Боб проводят взаимную аутентификацию, которая состоит в проверке знания общего секретного ключа. Протокол аутентификации следующий:

- Стороны обмениваются именами, каждый проверяет, что имена не совпадают.
- Алиса склеивает строки: своё имя + имя Боба + метка времени.
- Алиса вычисляет MAC полученной строки, используя секретный ключ, и отправляет результат Бобу.
- Боб проверяет MAC Алисы, используя секретный ключ.
- Боб склеивает строки: своё имя + имя Алисы + метка времени.
- Боб вычисляет MAC полученной строки, используя секретный ключ, и отправляет результат Алисе.
- Алиса проверяет MAC Боба, используя секретный ключ.

Как Мэллори, не зная ключа, может выдать себя за Боба?
Какие изменения нужно внести в протокол, чтобы закрыть уязвимость?

Ссылки

- Обратная связь:
  - android.ruberoid@gmail.com
  - @androidruberoid
- Анонсы:
  - facebook.com/kocherga.club
  - vk.com/kocherga_club
  - vk.com/kocherga_prog
- Материалы лекций:
  - github.com/notOcelot/Kocherga_crypto
- Видео:
  - youtube.com/channel/UCeLSDFOndl4eKFutg3oowHg