Защита информации

Павел Юдаев

МГТУ им. Баумана, Кафедра ИУ-9

Москва, 2014

Раздел 8 - Криптографические хэш функции

Хэш функции

Хэш на основе функции сжатия

HMAC

Опр.

детерминированная ф-я $H : M \to T$ - *хэш функция*, если $|M| \gg |T|$

Опр.

Коллизия для H - это пара $m_0, m_1 \in M : H(m_0) = H(m_1)$

Опр.

Функция H наз. *устойчивой к коллизиям*, если для любого явно описанного алгоритма $A \in \mathrm{PPT}$,
$Adv_{CR}[A, H] = P(A$ дает коллизию для H$) < \varepsilon(log(|T|))$ пренебр. малая.

На практике нужно $\forall A : time(A) < N$
$Adv_{CR}[A, H] = P(A$ дает коллизию для H$) < \varepsilon = const$

Следствие 1: (*Стойкость к поиску второго прообраза.*) Если H уст. к коллизиям, по паре (m, t) трудно найти $m_1 : t = H(m_1)$.

Следствие 2: Если H уст. к коллизиям, трудно найти два сообщения $m, m_1 : H(m) = H(m_1)$

Опр.

Функция H наз. *стойкой к поиску прообраза*, если $\forall t \in T$ трудно найти прообраз m, т.е. $m : t = H(m)$.
$\forall A \in \mathrm{PPT}\ Adv_{PI}[A, H] = P(A(t) = x : H(x) = t) < \varepsilon(log(|T|))$ (по выборам A).

Следствие: H - односторонняя функция.

Опр.

Функция H наз. *криптографической хэш функцией*, если она стойкая к коллизиям и к поиску прообраза.

Утверждение

$\forall$ хэш функции $H : M \to \{0, 1\}^n$ за время $O(2^{n/2})$ можно найти коллизию с вер-тью более 0.5.

Док-во

Применить парадокс дней рождения.

Скорость хэш функций (*)

AMD Opteron, 2.2 GHz, Linux, Crypto++ v5.6

	function	digest size (bits)	Speed (MB/sec)	generic attack time
NIST standards	SHA-1	160	153	2^{80}
	SHA-256	256	111	2^{128}
	SHA-512	512	99	2^{256}

(Лучший алгоритм поиска коллизий для SHA-1 требует вычисл. хэша для 2^{52} сообщ.)

TODO: в 2015 убрать все упоминания о квантовом компьютере. И так много материала.

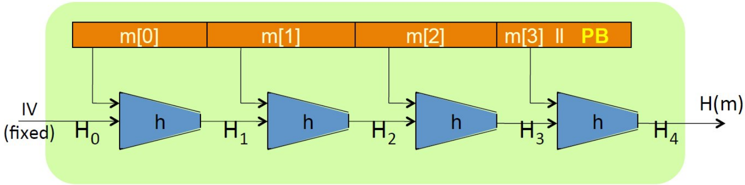
Поиск коллизий на квантовом компьютере (для справки) (*)

	обычн. комп.	квант. комп.
Блочн. шифр $C : K \times X \to X$, перебор ключей	$O(\lvert K \rvert)$	$O(\lvert K \rvert^{1/2})$
Хэш $H : M \to T$, поиск коллизий	$O(\lvert T \rvert^{1/2})$	$O(\lvert T \rvert^{1/3})$

Раздел 8 - Криптографические хэш функции

Конструкция Меркля-Дамгарда (Merkle-Damgard) - построение хэш функций на основе функции сжатия

По функции сжатия $h : T \times X \to T$ постр. хэш функцию $H : X^{\leq L} \to T$



PB - дополнение до длины блока, $(10..0||msg\_len)$, длина поля msg_len - 64 бита

Теорема 1

если h устойчива к коллизиям, то H - тоже.

Т.е., чтобы построить устойч. к коллизиям хэш функцию, достаточно
построить устойч. к коллизиям функцию сжатия.

Док-во

покажем, что коллизия h необх. для коллизии H,
т.е. коллизия $H \Rightarrow$ коллизия h.
Пусть $H(M) = H(M')$, $M \neq M'$. Построим коллизию для h.

Док-во (Продолжение)

В нашей конструкции для M и M' имеем две последовательности

$IV = H_0$	H_1	...	H_t	$h(H_t, M_t \| PB) = H_{t+1} = H(M)$
$IV' = H'_0$	H'_1	...	H'_r	$h(H'_r, M'_r \| PB') = H'_{r+1} = H(M')$

Справа $h(H_t, M_t || PB) = H_{t+1} = H'_{r+1} = h(H'_r, M'_r || PB')$

Если $H_t \neq H'_r$ или $M_t \neq M'_r$ или $PB \neq PB'$,
то имеем коллизию для h на последнем шаге.

Док-во (Продолжение)

Иначе имеем $H_t = H'_r$ & $M_t = M'_r$ & $PB = PB'$, след-но $t = r$.

След-но на предыд. шаге величины были равны:

$$h(H_{t-1},\ M_{t-1}) = H_t = H'_t = h(H'_{t-1},\ M'_{t-1})$$

Снова, если $H_{t-1} \neq H'_{t-1}$ или $M_{t-1} \neq M'_{t-1}$, то имеем коллизию для h.

Док-во (Продолжение)

След-но на предыд. шаге величины были равны:

$H_{t-1} = H'_{t-1}$ & $M_{t-1} = M'_{t-1}$

Пройдем по всем шагам, тогда: либо мы встретим коллизию для h, либо (т.к. по опр., $IV = IV'$)
$\forall i \; M_i = M'_i$, т.е. $M = M'$ - против. с тем, что $M \neq M'$.

Ч.т.д.

Функция сжатия на основе блочного шифра

(E, D) - блочный шифр, $E,\ D : K \times \{0,1\}^n \to \{0,1\}^n$

Если ф-я сжатия $h(H, m) := E(m, H)$,

то она имеет коллизии $(H, m), (H', m')$:
случ. одноблочные m, m',
пусть $H' = D(m', E(m, H)) \Rightarrow\ h(H', m') = E(m', H') = h(H, m)$.

Не годится для конструкции М.-Д.

Опр.

Функция сжатия Дэвиса-Мейера (Davies, Meyer) - это $h(H, m) = E(m, H) \oplus H$.

Теорема 2

Пусть (E,D) - идеальный шифр (т.е. это |K| различных перестановок множества M). Тогда для того, чтобы найти коллизию функции Дэвиса-Мейера $h(H, m) = h(H', m')$, необх. не менее $O(2^{n/2})$ вычислений шифра.

Т.е. достигается теоретическая граница. Без док-ва.

Другие варианты с одним $\oplus$ не уст. к коллизиям. Без док-ва.

(*) С двумя $\oplus$ многие варианты уст. к коллизиям,
но это медленнее на целый $\oplus$, напр.

(*) $h(H, m) = E(m, H) \oplus H \oplus m$
$h(H, m) = E(H \oplus m, m) \oplus m$

Пример

Х/ф SHA-256 (2001):

- Конструкция Меркля-Дамгарда
- Функция сжатия Дэвиса-Мейера
- Блочный шифр SHACAL-2

Раздел 8 - Криптографические хэш функции

Хэш функции

Хэш на основе функции сжатия

HMAC

MAC на основе хэш функции, устойчивой к коллизиям

Общий вид.

Пусть $I = (S, V) : K \times M \to T$ - MAC короткого сообщения, напр., AES для неск. блоков.
Пусть $H : M^{big} \to M$.

Определим $I^{big} = (S^{big}, V^{big}) : K \times M^{big} \to T$:
$S^{big}(k, m) = S(k, H(m))$,
$V^{big}(k, m, t) = V(k, H(m), t)$

(рисунок)

Пример

$S(k, m) = AES_{2-block-cbc}(k,\ SHA256(m))$ - криптостойкий MAC.

Теорема 3 (Достаточность)

Если I - криптостойкий MAC и H^{big} устойчивая к коллизиям х/ф, то I^{big} - криптостойкий MAC.

Док-во (Набросок. От противного.)

Пусть $y := H(m),\ t := S(k, y) = S(k, H(m))$.
1) A может найти $m' \neq m :\ H(m') = H(m)$. Против.: H устойч. к коллизиям.
2) A не может найти коллизию H, но может найти новую верную пару (m, t) для I^{big}
$\Leftrightarrow$ A может найти новую верную пару (y, t) для I. Против.: MAC I - криптостойкий.
Ч.т.д.

(рисунок)

Теорема 4 (Необходимость)

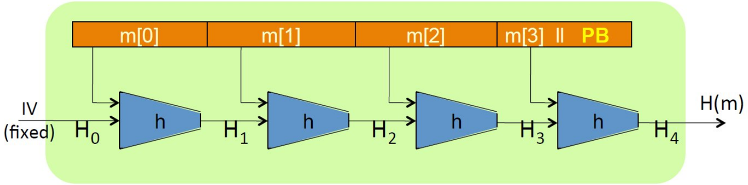
Если I^{big} криптостойкий, то H устойчивая к коллизиям х/ф.

Док-во

(От противн.) Пусть злоум-к может найти коллизию $m_0 \neq m_1 : H(m_0) = H(m_1)$. Тогда S^{big} не криптостойкий:
1) злоум-к получает $t = S(k, m_0)$
2) злоум-к предъявляет пару (m_1, t).
Ч.т.д.

HMAC - Hash-MAC

MAC на основе х/ф SHA-256



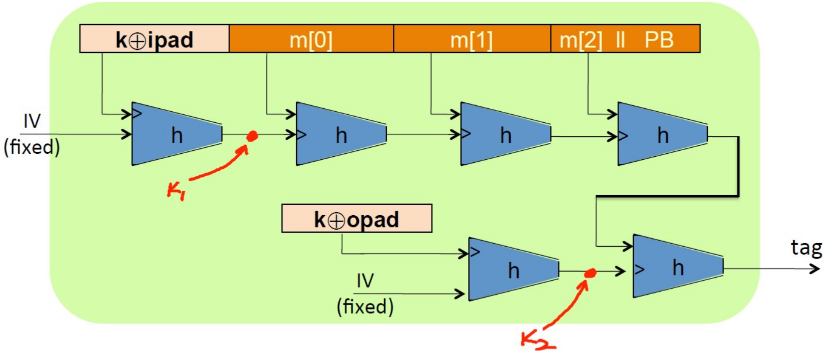
Х/ф Меркля-Дамгарда, устойч. к коллизиям. Построим MAC на ее основе.

Примитивно: $S(k, m) = H(k||m)$ или k вместо IV...

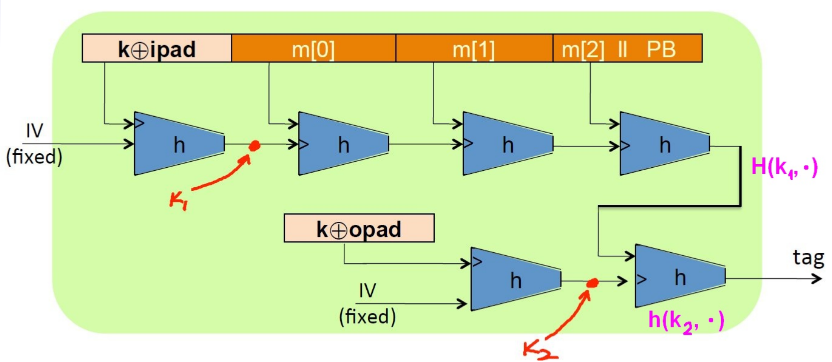
Проблема: $S(k, m) \Rightarrow S(k, m||PB||w) = h(w, S(k, m))$

Опр.

HMAC: $S(k,m) = H(k \oplus opad || H(k \oplus ipad || m))$. H - это SHA256.



opad, ipad - разные, открытые, одноблочные константы.
$\Rightarrow$ k_1, k_2 разные. (Похоже на CBC-MAC.)



$\hat{H} := H(k_1, \cdot)$ - SHA256, параметризованная секр. ключом k_1. Устойчивая к коллизиям по m.

$h(k_2, \cdot)$ - ф. сжатия Д.-М., уст. к коллизиям по m.

MAC $\hat{h}$: $S(k_2, m) = h(k_2, m)$.

Теорема 5

Пусть $\hat{H}$ - устойчивая к коллизиям х/ф.
Пусть $\hat{h}$ - криптостойкий MAC для сообщений фикс. длины.
Тогда HMAC - криптостойкий MAC для сообщений произв. длины.

Док-во (Набросок)

Аналогично тому, что было ранее для MAC.

Срок жизни ключа:
HMAC криптостойкий при $q^2/|T| < \varepsilon$

HMAC используется:
в TLS/SSL и во мн. др. сетевых протоколах.

Атаки на HMAC по сторонним каналам
Side-channel attacks

- по времени выполнения сравнения. Тривиальный программный код:

```
def Verify(key, msg, t):
  return t == HMAC(key, msg)
```

Размер тэга - 256 бит. Сравнение ленивое, побайтно.

Злоум-к за 256 запросов (разные $t[0]$) может узнать правильное значение первого байта тэга. Потом - второго байта...

Напишем код, который принудительно сравнивает все байты... компилятор может соптимизировать!

Поэтому - решение:

```
def Verify(key, msg, t):
  mac = HMAC(key,msg)
  return HMAC(key, mac) == HMAC(key, t)
```

Злоум-к не знает, какие значения сравниваются.

Литература к лекции

1*. Rogaway, Shrimpton. *Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance*
http://web.cs.ucdavis.edu/~rogaway/papers/relates.pdf