HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Introduction to Cryptography and Security

Collision Resistance

Slides are taken from Dan Boneh's Course

Outline

Collision Resistance

Definition

- Let $H : M \rightarrow T$ be a hash function where $|M| \gg |T|$.
  A **collision** for H is a pair $m_0, m_1 \in M$ such that:

$$H(m_0) = H(m_1) \quad \text{and} \quad m_0 \neq m_1$$

- A function H is **collision resistant** if for all (explicit) "efficient" algorithms A:

$$\text{Adv}_{CR}[A, H] = \Pr[A \text{ outputs collision for } H] \quad \text{is "negligible".}$$

Example

SHA-256 (outputs 256 bits)

MACs from Collision Resistance

- Let $I = (S, V)$ be a MAC for short messages over (K, M, T).
- Let $H : M^{big} \to M$ be a collision resistant hash function.
- We define $I^{big} = (S^{big}, V^{big})$ over (K, M^{big}, T) as follows:

$$S^{big}(k, m) = S(k,\ H(m)\ ) \quad ; \quad V^{big}(k, m, t) = V(k,\ H(m)\ , t)$$

Theorem

If I is a secure MAC and H is collision resistant, then I^{big} is a secure MAC.

Example

$S(k, m) = AES_{\text{2-block-cbc}}(k, \text{SHA-256}(m))$ is a secure MAC.

MACs from Collision Resistance

$$S^{big}(k, m) = S(k, H(m)) \quad ; \quad V^{big}(k, m, t) = V(k, H(m), t)$$

Collision resistance is necessary for security:

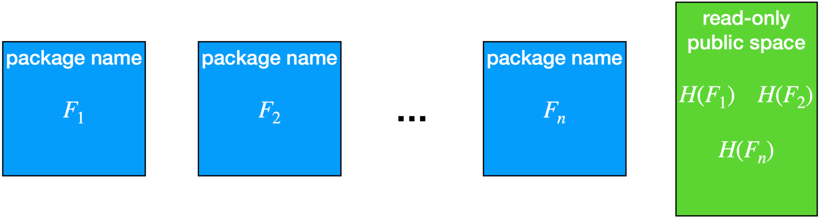
Suppose adversary can find $m_0 \neq m_1$ such that

$$H(m_0) = H(m_1).$$

Then S^{big} is insecure under a 1-chosen message attack:

1. adversary asks for $t \leftarrow S(k, m_0)$
2. output (m_1, t) as forgery

Protecting file integrity using C.R. hash



When user downloads package, can verify that contents are valid

- If H collision resistant then attacker cannot modify package without detection
- no key needed (public verifiability), but requires read-only space

Outline

Generic attack on C.R. functions

- Let $H : M \to \{0,1\}^n$ be a hash function ($|M| \gg 2n$)
- Generic algorithm to find a collision in time $O(2^{n/2})$ hashes:

Algorithm

1. Choose $2^{n/2}$ random messages in M: $m_1, \dots, m_{2^{n/2}}$ (distinct w.h.p)
2. For $i = 1, \dots, 2^{n/2}$ compute $t_i = H(m_i) \in \{0,1\}^n$
3. Look for a collision $(t_i = t_j)$. If not found, got back to step 1.

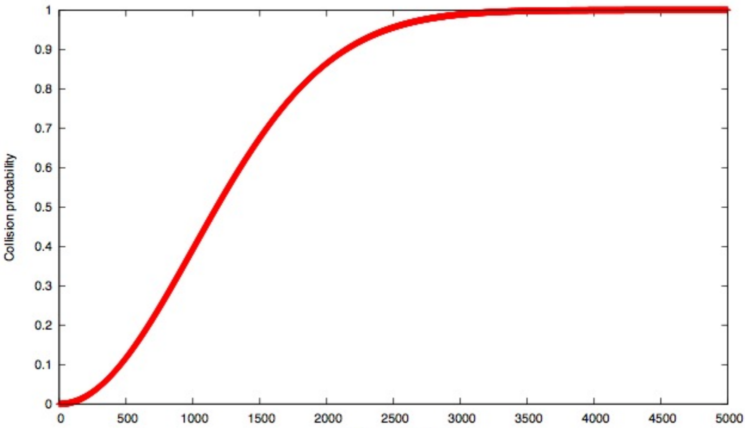
The birthday paradox

Let $r_1, \ldots, r_n \in \{1, \ldots, B\}$ be independent identically distributed integers.

Theorem

When $n = 1.2 \times B^{1/2}$ *then* $\Pr[\ \exists i \neq j: \ \ r_i = r_j\ ] \geq 1/2$.

$B = 10^6$



Collision probability
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000

Generic attack

Let $H : M \to \{0,1\}^n$. Collision finding algorithm:

1. Choose $2^{n/2}$ random elements in M:

$$m_1, \ldots, m_{2^{n/2}}$$

2. For $i = 1, \ldots, 2^{n/2}$ compute $t_i = H(m_i) \in \{0,1\}^n$
3. Look for a collision $(t_i = t_j)$. If not found, got back to step 1.

Expected number of iteration ≈ 2

Running time: $O(2^{n/2})$ (space $O(2^{n/2})$)

Sample C.R. hash functions

Crypto++ 5.6.0 [Wei Dai]

Function	Digest size (bits)	Speed (MB/sec)	Generic attack time
SHA-1	160	153	280
SHA-256	256	111	2128
SHA-512	512	99	2256
Whirlpool	512	57	2256

Best known collision finder for SHA-1 requires 2^{51} hash evaluations

Quantum Collision Finder

	Classical algo	Quantum algo
Block cipher $E : K \times X \to X$ exhaustive search	$O(\lvert K \rvert)$	$O(\lvert K \rvert^{1/2})$
Hash function $H : M \to T$ collision finder	$O(\lvert T \rvert^{1/2})$	$O(\lvert T \rvert^{1/3})$

Outline

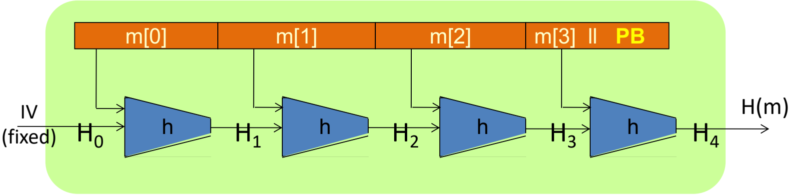
Collision resistance: review

- Let $H : M \to T$ be hash function ($|M| \gg |T|$).
- A **collision** for H is a pair $m_0, m_1 \in M$ such that:

$$H(m_0) = H(m_1) \quad \text{and} \quad m_0 \neq m_1$$

- Goal: collision resistant (C.R.) hash functions
- Step 1: given C.R. function for short messages, construct C.R. function for long messages

The Merkle-Damgard iterated construction

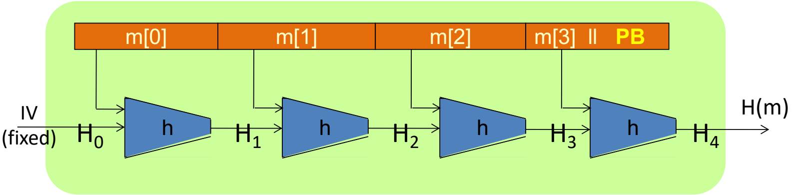


- Given a (compression) function $h : T \times X \to T$,
- we obtain $H : X^{\leq L} \to T$ where H_i are chaining variables and
- PB is padding block

$$1000\cdots \,\|\, \overbrace{\text{msg len}}^{64 \text{ bit}}$$

If no space for PB, then add another block.

The Merkle-Damgard iterated construction



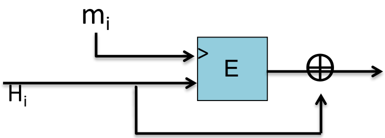
Theorem

If the compression function h is collision resistant, then H is collision resistant.

Compression Functions from Block Cipher

- Let $E : K \times \{0,1\}^n \to \{0,1\}^n$ a block cipher.
- The Davies-Meyer compression function:

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



Theorem

Suppose E is an ideal cipher (collection of $|K|$ random permutations.). Finding a collision $h(H, m) = h(H', m')$ takes $O(2^{n/2})$ evaluations of (E, D).

Question

Suppose we define

$$h(H, m) = E(m, H).$$

Then the resulting $h(.,.)$ is not collision resistant.

To build a collision (H, m) and (H', m') choose random (H, m, m') and construct H' as follows:

1. $H' = D(m', E(m, H))$
2. $H' = E(m', D(m, H))$
3. $H' = E(m', E(m, H))$
4. $H' = D(m', D(m, H))$

Other block cipher constructions

Let $E : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ for simplicity.

Miyaguchi-Preneel:

- $h(H, m) = E(m, H) \oplus H \oplus m$ (Whirlpool)
- $h(H, m) = E(H \oplus m, m) \oplus m$

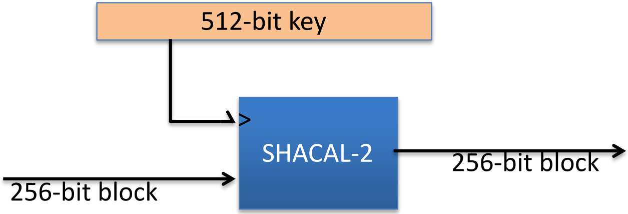
total of 12 variants like this

Other natural variants are insecure:

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

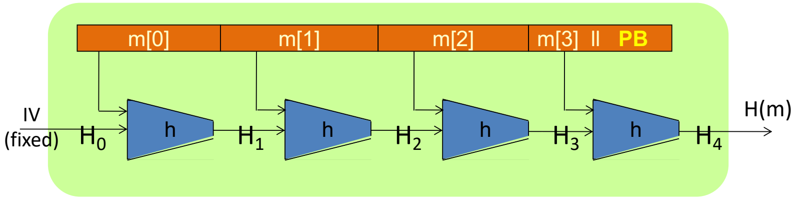
Case study: SHA-256

- Merkle-Damgard function
- Davies-Meyer compression function
- Block cipher: SHACAL-2



Outline

The Merkle-Damgard iterated construction



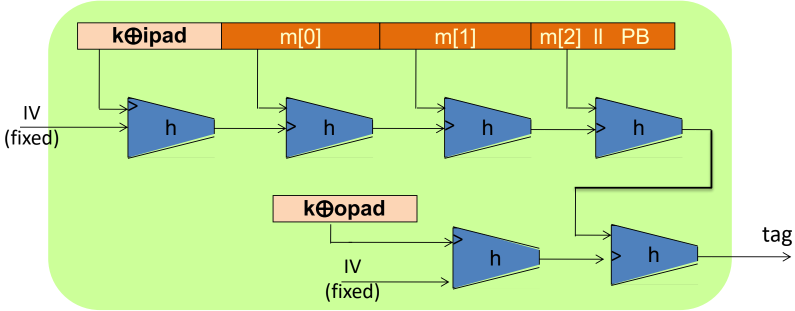
Theorem

If h collision resistant, then H collision resistant.

Question

Can we use $H(.)$ to directly build a MAC?

HMAC in pictures



- Similar to the NMAC PRF.
- Main difference: the two keys k_1, k_2 are dependent.

HMAC properties

Built from a black-box implementation of SHA-256.

HMAC is assumed to be a secure PRF

- Can be proven under certain PRF assumptions about $h(.,.)$
- Security bounds similar to NMAC: Need $q^2/|T|$ to be negligible ($q \ll |T|^{1/2}$)

In TLS: must support HMAC-SHA1-96

Outline

Warning: verification timing attacks

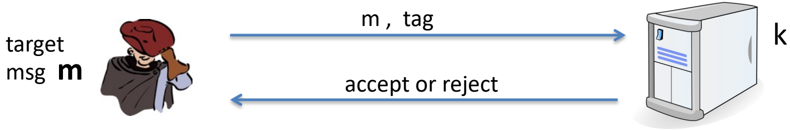
Example: Keyczar crypto library (Python) [simplified]

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

The problem: `==' implemented as a byte-by-byte comparison

- Comparator returns false when first inequality found

Warning: verification timing attacks



Timing attack: to compute tag for target message m do:

1. Query server with random tag.
2. Loop over all possible first bytes and query server.
   Stop when verification takes a little longer than in step 1.
3. Repeat for all tag bytes until valid tag found.

35	53	*	*	*	*

Defense #1

Make string comparator always take same time (Python):

```
return false if  sig_bytes  has wrong length
result = 0
for x, y in zip( HMAC(key,msg) , sig_bytes):
    result |= ord(x) ^ ord(y)
return result == 0
```

Can be difficult to ensure due to optimizing compiler.

Defense #2

Make string comparator always take same time (Python) :

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

Attacker doesn't know values being compared.

Lesson

Don't implement crypto yourself !

VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Thank you!