Collision resistance

This slide is made based the online course of Cryptography by Dan Boneh

Collision Resistance

Let $H: M \rightarrow T$ be a hash function ($|M| >> |T|$)

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) "eff" algs. A:

$$\mathbf{Adv_{CR}[A,H] = Pr[\ A \text{ outputs collision for } H]}$$

is "neg".

Example: SHA-256 (outputs 256 bits)

MACs from Collision Resistance

Let I = (S,V) be a MAC for short messages over (K,M,T) (e.g. AES)

Let H: $M^{big} \rightarrow M$

Def: $I^{big} = (S^{big}, V^{big})$ over (K, M^{big}, T) as:

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

Thm: If I is a secure MAC and H is collision resistant

then I^{big} is a secure MAC.

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

MACs from Collision Resistance

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

Collision resistance is necessary for security:

Suppose adversary can find $m_0 \neq m_1$ s.t. $H(m_0) = H(m_1)$.

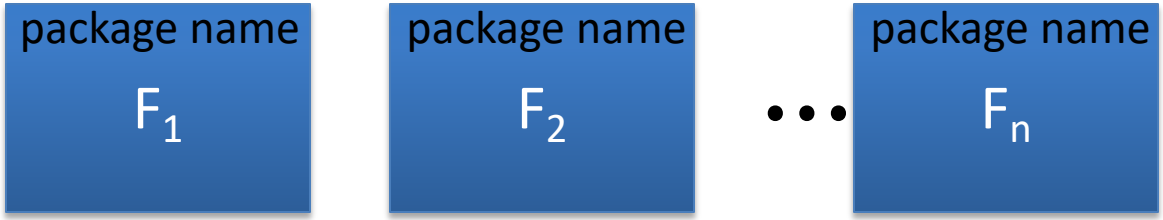
Then: S^{big} is insecure under a 1-chosen msg attack

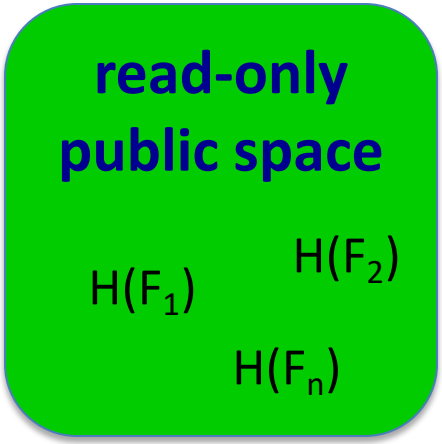
step 1: adversary asks for $t \leftarrow S(k, m_0)$

step 2: output (m_1, t) as forgery

Protecting file integrity using C.R. hash

Software packages:





When user downloads package, can verify that contents are valid

H collision resistant ⇒
attacker cannot modify package without detection

no key needed (public verifiability), but requires read-only space

End of Segment

Collision resistance

Generic birthday attack

Generic attack on C.R. functions

Let $H: M \rightarrow \{0,1\}^n$ be a hash function ($|M| >> 2^n$)

Generic alg. to find a collision **in time $O(2^{n/2})$** hashes

Algorithm:

1. Choose $\mathbf{2^{n/2}}$ random messages in M: $m_1, \ldots, m_{2^{n/2}}$ (distinct w.h.p)
2. For i = 1, …, $2^{n/2}$ 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.

How well will this work?

The birthday paradox

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

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

Proof: (for uniform indep. $r_1, \ldots, r_n$)

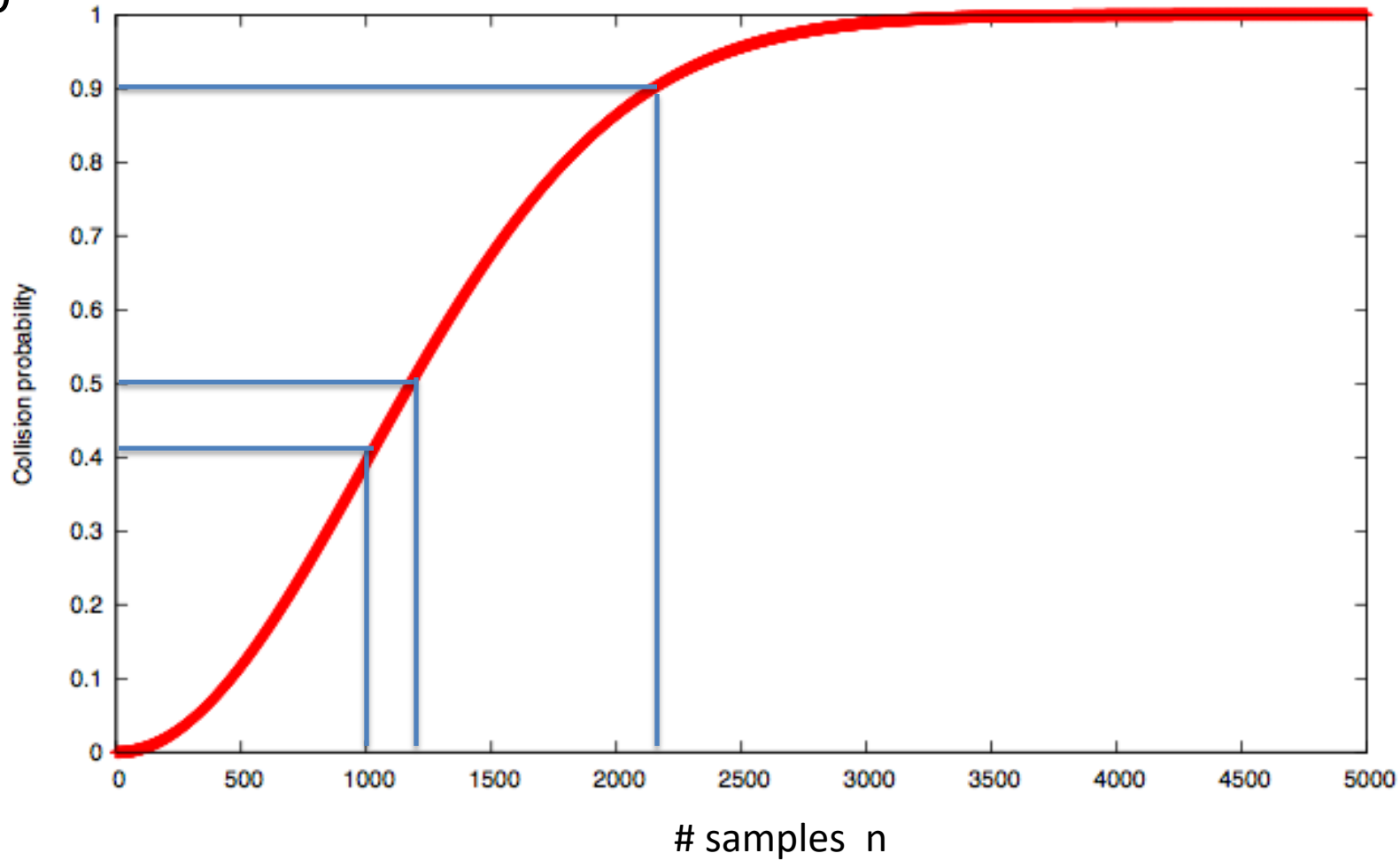
$$\Pr[\exists\, i \neq j: r_i = r_j] = 1 - \Pr[\forall i \neq j: r_i \neq r_j]$$

$$= 1 - \left(\frac{B-1}{B}\right)\left(\frac{B-2}{B}\right)\ldots\left(\frac{B-n+1}{B}\right) = 1 - \prod_{i=1}^{n-1}\left(1 - \frac{i}{B}\right) \geq 1 - \prod_{i=1}^{n-1} e^{-\frac{i}{B}}$$

$$= 1 - e^{-\frac{1}{B}\sum_{i=1}^{n-1} i} \geq 1 - e^{-\frac{n^2}{2B}} \geq 1 - e^{-0.72} = 0.53 > 1/2$$

$$1 - x \leq e^{-x},\ \frac{n^2}{2B} = 0.72$$

$B=10^6$


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
samples n

Generic attack

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

1. Choose $\mathbf{2^{n/2}}$ random elements in M: $m_1, \ldots, m_{2^{n/2}}$
2. For i = 1, …, $2^{n/2}$ 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: $\mathbf{O(2^{n/2})}$ (space $O(2^{n/2})$)

Sample C.R. hash functions:

Crypto++ 5.6.0 [Wei Dai]

AMD Opteron, 2.2 GHz (Linux)

	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}
	Whirlpool	512	57	2^{256}

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

Quantum Collision Finder

	Classical algorithms	**Quantum algorithms**
Block cipher **E: K × X ⟶ X** exhaustive search	$O(\|K\|)$	$O(\|K\|^{1/2})$
Hash function **H: M ⟶ T** collision finder	$O(\|T\|^{1/2})$	$O(\|T\|^{1/3})$

End of Segment

Collision resistance

The Merkle-Damgard Paradigm

Collision resistance: review

Let $H: M \rightarrow T$ be a hash function ($|M| >> |T|$)

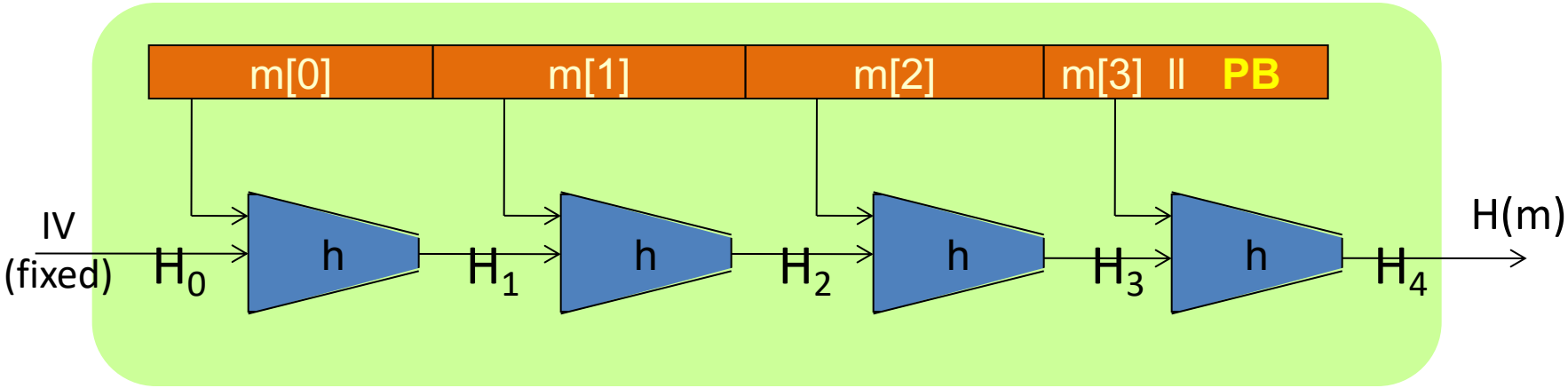
A **collision** for H is a pair $m_0, m_1 \in M$ such that:

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

Goal: collision resistant (C.R.) hash functions

Step 1: given C.R. function for **short** messages,
construct C.R. function for **long** messages

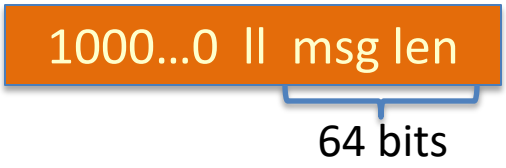
The Merkle-Damgard iterated construction



Given **h: T × X ⟶ T** (compression function)

we obtain **H: $X^{\leq L}$ ⟶ T** . H_i - chaining variables

PB: padding block



If no space for PB
add another block

MD collision resistance

Thm: if h is collision resistant then so is H.

Proof: collision on H ⇒ collision on h

Suppose H(M) = H(M'). We build collision for h.

IV = H_0 , H_1 , … , H_t , H_{t+1} = H(M)

IV = H_0' , H_1' , … , H'_r, H'_{r+1} = H(M')

If $[H_t \neq H'_r \; or M_t \neq M'_r \; or \; PB \neq PB']$ ⇒ We have a collission on h. Stop.

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

Otherwise, Suppose $H_t = H'_r$ and $M_t = M'_r$ and PB = PB'

t=r

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

If $[H_t \neq H'_{t-1} \; or M_t \neq M'_{t-1}]$ then we have a collision on h. Stop.

Otherwise, $H_t \neq H'_{t-1} \; and \; M_t \neq M'_t$ and $M_{t-1} = M'_{t-1}$

Iterate all the way to beginning and either :

(1)find collision on h,or

(2) $\forall \, i: M_i = M'_i \Longrightarrow M = M'$ (Cannot happen because M, M' are collision on H.)

$\Rightarrow$ To construct C.R. function,

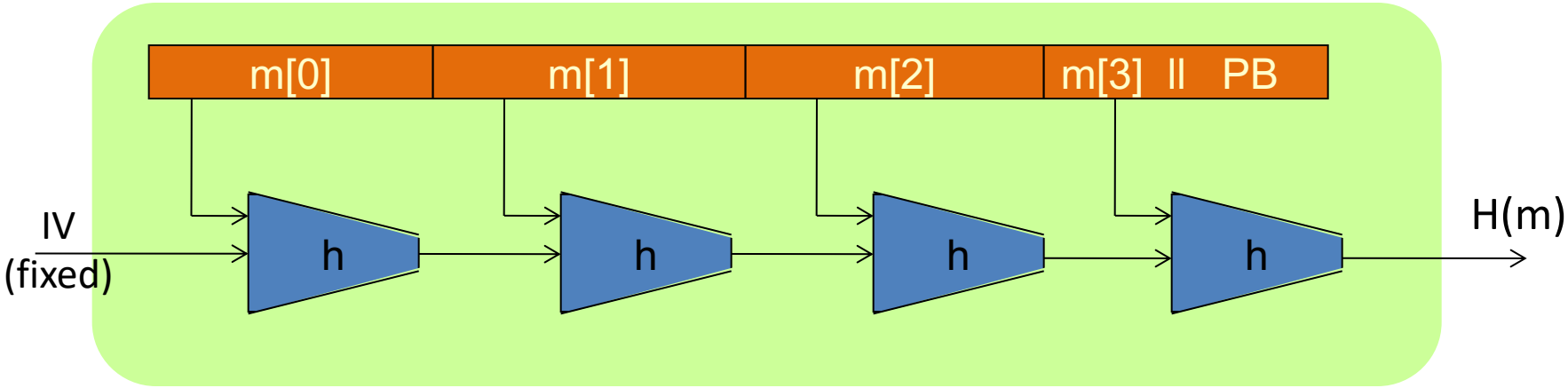
suffices to construct compression function

End of Segment

Collision resistance

HMAC:
a MAC from SHA-256

The Merkle-Damgard iterated construction



Thm: h collision resistant ⇒ H collision resistant

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

MAC from a Merkle-Damgard Hash Function

H: $X^{\leq L} \longrightarrow T$ a C.R. Merkle-Damgard Hash Function

Attempt #1: **S(k, m) = H(k || m)**

This MAC is insecure because:

- ○ Given H(k || m) can compute H(w || k || m || PB) for any w.
- ○ Given H(k || m) can compute H(k || m || w) for any w.
- √ ○ Given H(k || m) can compute H(k || m || PB || w) for any w.
- ○ Anyone can compute H(k || m) for any m.

Standardized method: HMAC (Hash-MAC)

Most widely used MAC on the Internet.

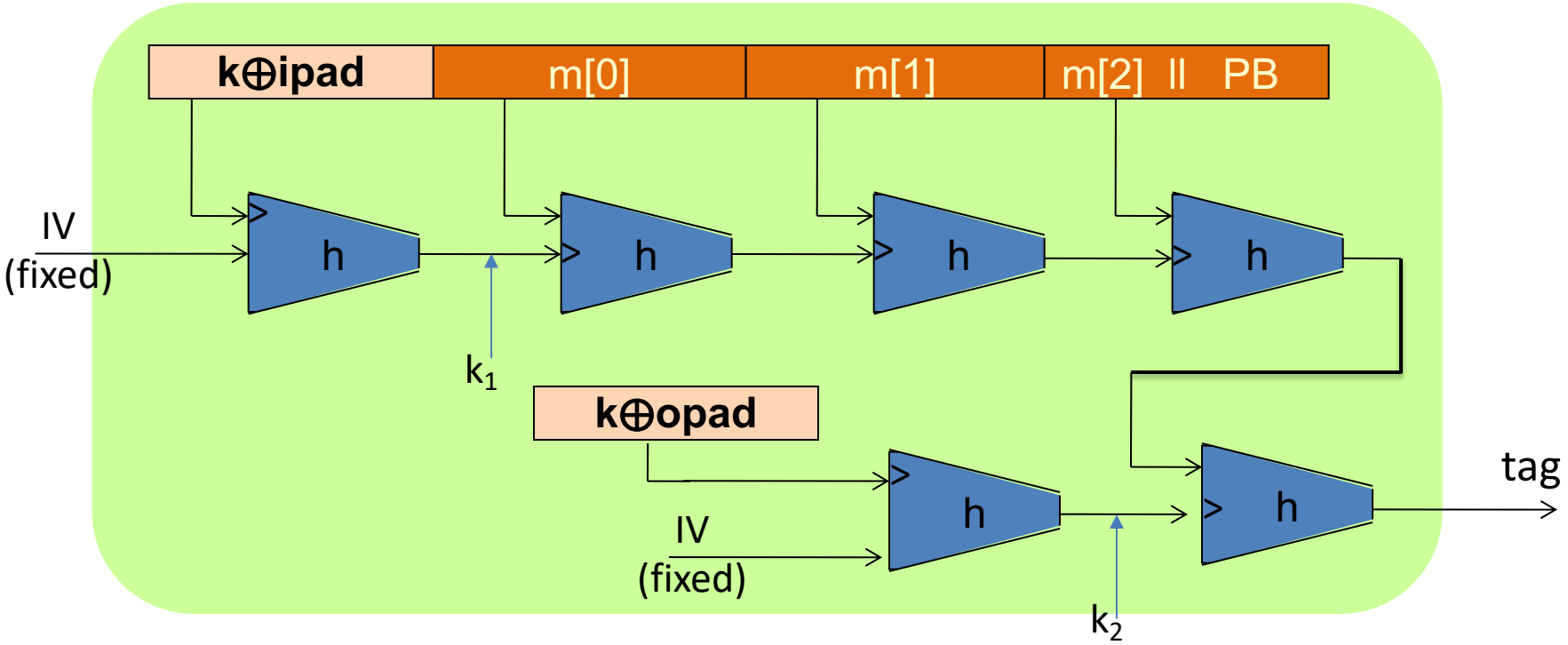
H: hash function.

example: SHA-256 ; output is 256 bits

Building a MAC out of a hash function:

HMAC: $S(k, m) = H(\ k \oplus opad \ \| \ \mathbf{H(\ k \oplus ipad \ \| \ m\ )}\ )$

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 ($q << |T|^{½}$)

In TLS: must support HMAC-SHA1-96

End of Segment