

The Sum can be Weaker Than Each Part

Generic Attacks against the Sum of Two Hash Functions

Gaëtan Leurent¹ Lei Wang²

¹Inria, France

²NTU, Singapore

Eurocrypt 2015

The Sum can be Weaker Than Each Part

Generic Attacks against the Sum of Two Hash Functions

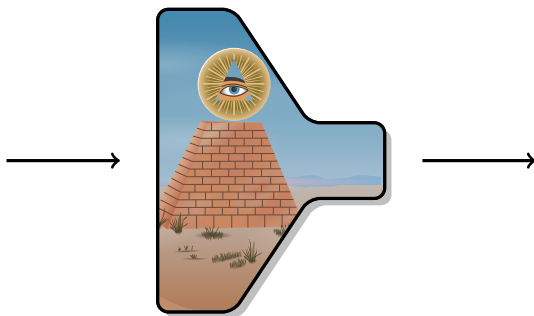
Gaëtan Leurent¹ Lei Wang²

¹Inria, France

²NTU, Singapore

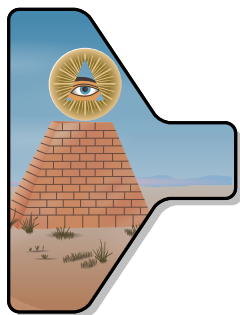
Eurocrypt 2015

An Ideal Hash Function: the Random Oracle



- ▶ Public Random Oracle
- ▶ The output can be used as a fingerprint of the document

An Ideal Hash Function: the Random Oracle



0x1d66ca77ab361c6f

- ▶ Public Random Oracle
- ▶ The output can be used as a fingerprint of the document

Concrete security goals

Preimage attack

Given F and \overline{H} , find M s.t. $F(M) = \overline{H}$.

Ideal security: 2^n .

Second-preimage attack

Given F and M_1 , find $M_2 \neq M_1$ s.t. $F(M_1) = F(M_2)$.

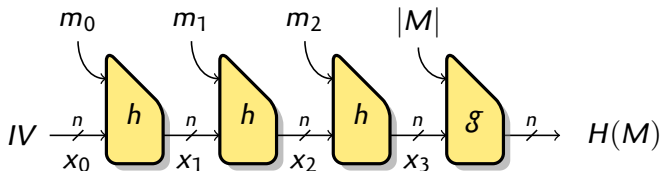
Ideal security: 2^n .

Collision attack

Given F , find $M_1 \neq M_2$ s.t. $F(M_1) = F(M_2)$.

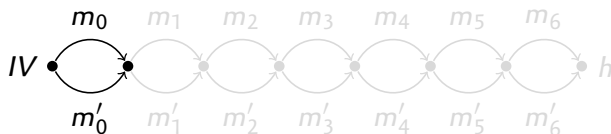
Ideal security: $2^{n/2}$.

Iterated hash function (Merkle-Damgård)



- ▶ n -bit state, compression function
- ▶ Security with ideal compression function:
 - ▶ Collisions: $2^{n/2}$ (optimal)
 - ▶ Preimages: 2^n (optimal)
 - ▶ Second-preimage: 2^{n-t}
 - ▶ Non-ideal after $2^{n/2}$: multi-collisions, herding, long 2nd-preimage

Joux's multicollision attack

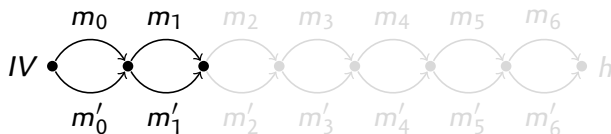


- 1 Find a collision pair m_0/m'_0 starting from IV
- 2 Find a collision pair m_1/m'_1 starting from $x_1 = h(IV, m_0)$
- 3 Repeat k times
- 4 This yields 2^k messages with the same hash:

$$\begin{array}{cccc}
 m_0 m_1 m_2 \dots & m'_0 m_1 m_2 \dots & m_0 m'_1 m_2 \dots & m'_0 m'_1 m_2 \dots \\
 m_0 m_1 m'_2 \dots & m'_0 m_1 m'_2 \dots & m_0 m'_1 m'_2 \dots & m'_0 m'_1 m'_2 \dots
 \end{array}$$

► Complexity $k \cdot 2^{n/2}$ vs. $\approx 2^{\frac{2^k-1}{2^k}n}$ for a random function

Joux's multicollision attack

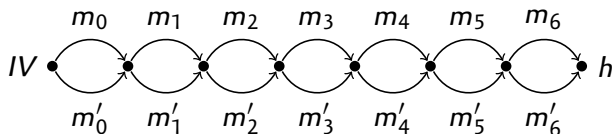


- 1 Find a collision pair m_0/m'_0 starting from IV
- 2 Find a collision pair m_1/m'_1 starting from $x_1 = h(IV, m_0)$
- 3 Repeat k times
- 4 This yields 2^k messages with the same hash:

$m_0 m_1 m_2 \dots$	$m'_0 m_1 m_2 \dots$	$m_0 m'_1 m_2 \dots$	$m'_0 m'_1 m_2 \dots$
$m_0 m_1 m'_2 \dots$	$m'_0 m_1 m'_2 \dots$	$m_0 m'_1 m'_2 \dots$	$m'_0 m'_1 m'_2 \dots$

► Complexity $k \cdot 2^{n/2}$ vs. $\approx 2^{\frac{2^k - 1}{2^k} n}$ for a random function

Joux's multicollision attack

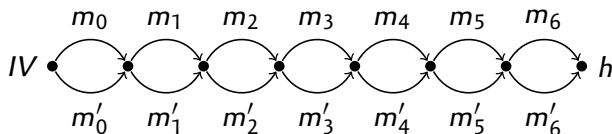


- 1 Find a collision pair m_0/m'_0 starting from IV
- 2 Find a collision pair m_1/m'_1 starting from $x_1 = h(IV, m_0)$
- 3 Repeat k times
- 4 This yields 2^k messages with the same hash:

$m_0 m_1 m_2 \dots$ $m'_0 m_1 m_2 \dots$ $m_0 m'_1 m_2 \dots$ $m'_0 m'_1 m_2 \dots$
 $m_0 m_1 m'_2 \dots$ $m'_0 m_1 m'_2 \dots$ $m_0 m'_1 m'_2 \dots$ $m'_0 m'_1 m'_2 \dots$

► Complexity $k \cdot 2^{n/2}$ vs. $\approx 2^{\frac{2^k - 1}{2^k} n}$ for a random function

Joux's multicollision attack

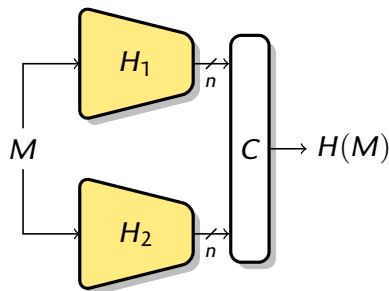


- 1 Find a collision pair m_0/m'_0 starting from IV
- 2 Find a collision pair m_1/m'_1 starting from $x_1 = h(IV, m_0)$
- 3 Repeat k times
- 4 This yields 2^k messages with the same hash:

$$\begin{array}{cccc}
 m_0 m_1 m_2 \dots & m'_0 m_1 m_2 \dots & m_0 m'_1 m_2 \dots & m'_0 m'_1 m_2 \dots \\
 m_0 m_1 m'_2 \dots & m'_0 m_1 m'_2 \dots & m_0 m'_1 m'_2 \dots & m'_0 m'_1 m'_2 \dots
 \end{array}$$

► Complexity $k \cdot 2^{n/2}$ vs. $\approx 2^{\frac{2^k - 1}{2^k} n}$ for a random function

Combining two hash functions



"In order to make the PRF as secure as possible, it uses two hash algorithms in a way which should guarantee its security if either algorithm remains secure."

– RFC 2246 (TLS 1.0)

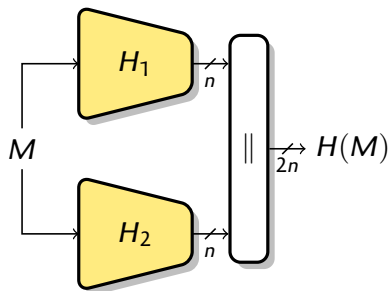
Classical combiners:

- ▶ Concatenation:
 $H_1(M) \parallel H_2(M)$
- ▶ Xor:
 $H_1(M) \oplus H_2(M)$

"The whole is greater than the sum of its parts"

– Aristotle

Combining two hash functions



"In order to make the PRF as secure as possible, it uses two hash algorithms in a way which should guarantee its security if either algorithm remains secure."

– RFC 2246 (TLS 1.0)

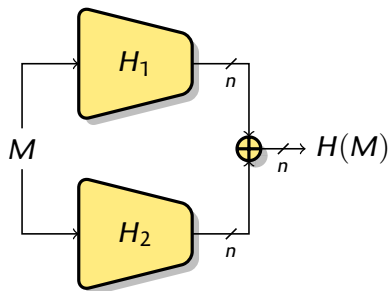
Classical combiners:

- ▶ **Concatenation:**
 $H_1(M) \parallel H_2(M)$
- ▶ **Xor:**
 $H_1(M) \oplus H_2(M)$

"The whole is greater than the sum of its parts"

– Aristotle

Combining two hash functions



"In order to make the PRF as secure as possible, it uses two hash algorithms in a way which should guarantee its security if either algorithm remains secure."

– RFC 2246 (TLS 1.0)

Classical combiners:

► Concatenation:
 $H_1(M) \parallel H_2(M)$

► Xor:
 $H_1(M) \oplus H_2(M)$

"The whole is greater than the sum of its parts"

– Aristotle

Known results: Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2 \times n$ -bit internal state, $2n$ -bit output

- ▶ *Robust combiner* for collisions

- ▶ A collision in H implies a collision in H_1 and H_2

- ▶ $2 \times n$ -bit internal state can increase security?

- ▶ **NO:** Multicollision attack

[Joux '04]

- ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Essentially n -bit security

Known results: Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2 \times n$ -bit internal state, $2n$ -bit output
- ▶ *Robust combiner* for collisions
 - ▶ A collision in H implies a collision in H_1 and H_2
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ **NO:** Multicollision attack [Joux '04]
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Essentially n -bit security

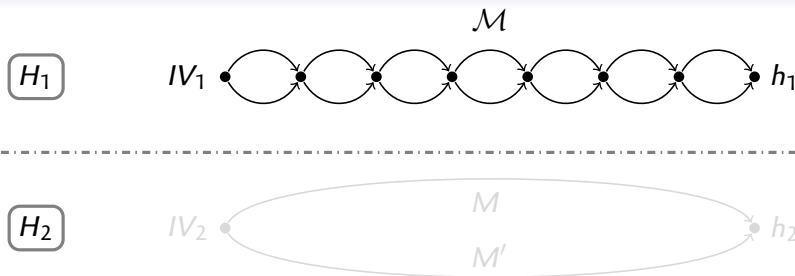
Known results: Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2 \times n$ -bit internal state, $2n$ -bit output
- ▶ *Robust combiner* for collisions
 - ▶ A collision in H implies a collision in H_1 and H_2
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ **NO:** Multicollision attack [Joux '04]
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Essentially n -bit security

Known results: Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2 \times n$ -bit internal state, $2n$ -bit output
- ▶ *Robust combiner* for collisions
 - ▶ A collision in H implies a collision in H_1 and H_2
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ **NO:** Multicollision attack [Joux '04]
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Essentially n -bit security

Collision attack for $H_1(M) \parallel H_2(M)$



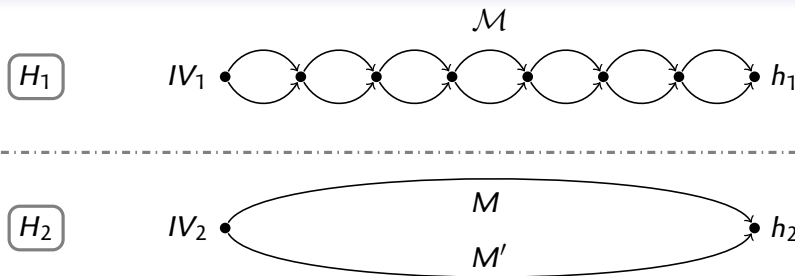
1 Build a $2^{n/2}$ -multicollision for H_1

$$\forall M \in \mathcal{M}, H_1(M) = h_1$$

2 Find $M, M' \in \mathcal{M}$ s.t. $H_2(M) = H_2(M')$

► Complexity $n \cdot 2^{n/2}$ vs. 2^n for a $2n$ -bit hash function.

Collision attack for $H_1(M) \parallel H_2(M)$



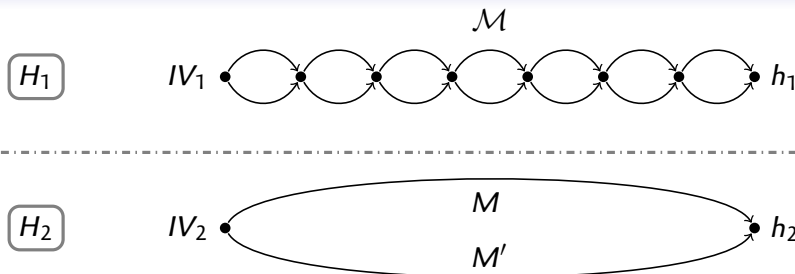
- 1 Build a $2^{n/2}$ -multicollision for H_1

$$\forall M \in \mathcal{M}, H_1(M) = h_1$$

- 2 Find $M, M' \in \mathcal{M}$ s.t. $H_2(M) = H_2(M')$

► Complexity $n \cdot 2^{n/2}$ vs. 2^n for a $2n$ -bit hash function.

Collision attack for $H_1(M) \parallel H_2(M)$



- 1 Build a $2^{n/2}$ -multicollision for H_1

$$\forall M \in \mathcal{M}, H_1(M) = h_1$$

- 2 Find $M, M' \in \mathcal{M}$ s.t. $H_2(M) = H_2(M')$

► Complexity $n \cdot 2^{n/2}$ vs. 2^n for a $2n$ -bit hash function.

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
 - ▶ $2 \times n$ -bit internal state, n -bit output
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ NO: Joux's attacks are applicable
 - ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]
 - ▶ Doesn't imply a generic attack...
 - ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ $2 \times n$ -bit internal state, n -bit output

- ▶ $2 \times n$ -bit internal state can increase security?

- ▶ NO: Joux's attacks are applicable

- ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]

- ▶ Doesn't imply a generic attack...

- ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ $2 \times n$ -bit internal state, n -bit output

- ▶ $2 \times n$ -bit internal state can increase security?

▶ **NO:** Joux's attacks are applicable

- ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]

▶ Doesn't imply a generic attack...

- ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ $2 \times n$ -bit internal state, n -bit output

- ▶ $2 \times n$ -bit internal state can increase security?

▶ **NO:** Joux's attacks are applicable

- ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]

▶ Doesn't imply a generic attack...

- ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ $2 \times n$ -bit internal state, n -bit output
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ **NO:** Joux's attacks are applicable
- ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]
 - ▶ Doesn't imply a generic attack...
- ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Known results: Xor Combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ $2 \times n$ -bit internal state, n -bit output
- ▶ $2 \times n$ -bit internal state can increase security?
 - ▶ **NO:** Joux's attacks are applicable
- ▶ Short output robust combiners don't exist [Boneh & Boyen '06, ...]
 - ▶ Doesn't imply a generic attack...
- ▶ Secure up to $2^{n/2}$ with weak compression fcts [Hoch & Shamir '08]
 - ▶ In particular, no generic collision attack

Generic attacks against combiniers

Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2n$ -bit output
- ▶ Generic attacks:
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Non-ideal after $2^{n/2}$

XOR combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ n -bit output
- ▶ Generic attacks:
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in ???
 - ▶ Non-ideal after $2^{n/2}$

Suprising result

If H_1 and H_2 are good MD hash functions, $H_1 \oplus H_2$ is weak!

Generic attacks against combiniers

Concatenation combiner

- ▶ $H(M) = H_1(M) \parallel H_2(M)$
- ▶ $2n$ -bit output
- ▶ Generic attacks:
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in 2^n
 - ▶ Non-ideal after $2^{n/2}$

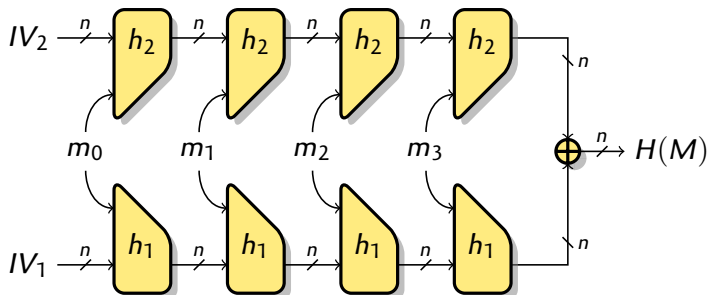
XOR combiner

- ▶ $H(M) = H_1(M) \oplus H_2(M)$
- ▶ n -bit output
- ▶ Generic attacks:
 - ▶ Collisions in $2^{n/2}$
 - ▶ Preimages in $\leq 2^{5n/6}$
 - ▶ Non-ideal after $2^{n/2}$

Suprising result

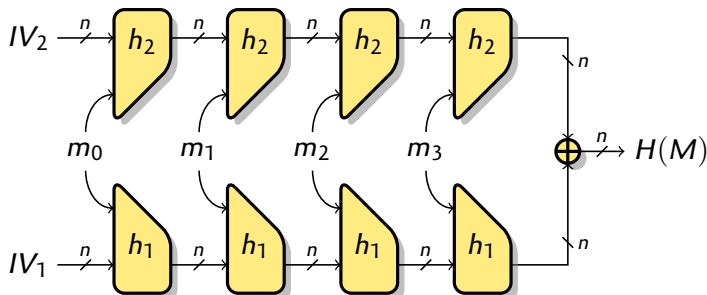
If H_1 and H_2 are good MD hash functions, $H_1 \oplus H_2$ is weak!

Our target: $H(M) = H_1(M) \oplus H_2(M)$



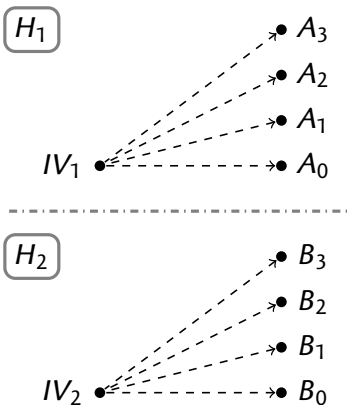
- ▶ $2 \times n$ -bit state, 2 compression functions
- ▶ Can we use a birthday-type attack on the final XOR?

Our target: $H(M) = H_1(M) \oplus H_2(M)$



- ▶ $2 \times n$ -bit state, 2 compression functions
- ▶ Can we use a birthday-type attack on the final XOR?

Overview



- Build a structure $\{\mathbf{M}_{ij}\}$ to control H_1 and H_2 :

$$(IV_1, IV_2) \overset{\mathbf{M}_{jk}}{\rightsquigarrow} (A_j, B_k)$$

- Horizontal lines: common message M

$$(a_j^i, b_k^i) \xrightarrow{M_i} (a_j^{i+1}, b_k^{i+1})$$

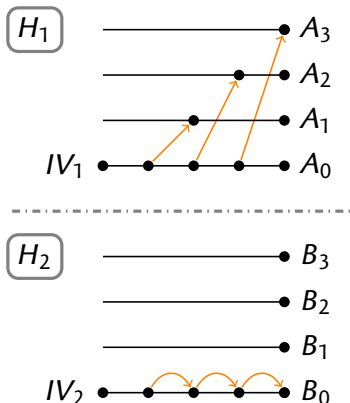
- Orange lines: alternative messages

$$(a_{j_0}^i, b_{k_0}^i) \xrightarrow{M'_i} (a_{j_0}^{i+1}, b_{k_1}^{i+1})$$

- Message in the structure use a few alternative chunks:

$$\mathbf{M}_{ij} = M_0, M_1, \dots, M'_{ij}, \dots$$

Overview



- Build a structure $\{\mathbf{M}_{ij}\}$ to control H_1 and H_2 :

$$(IV_1, IV_2) \overset{\mathbf{M}_{jk}}{\rightsquigarrow} (A_j, B_k)$$

- Horizontal lines: common message M

$$(a_j^i, b_k^i) \xrightarrow{M_i} (a_j^{i+1}, b_k^{i+1})$$

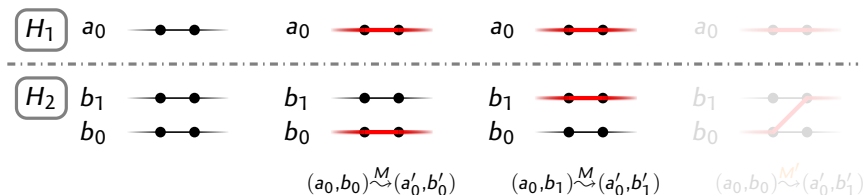
- Orange lines: **alternative messages**

$$(a_{j_0}^i, b_{k_0}^i) \xrightarrow{M'_i} (a_{j_0}^{i+1}, b_{k_1}^{i+1})$$

- Message in the structure use a few alternative chunks:

$$\mathbf{M}_{ij} = M_0, M_1, \dots, \mathbf{M}'_{ij}, \dots$$

Switch structure



► Simple case: one H_1 -chain, and two H_2 -chains

► Input: a_0, b_0, b_1

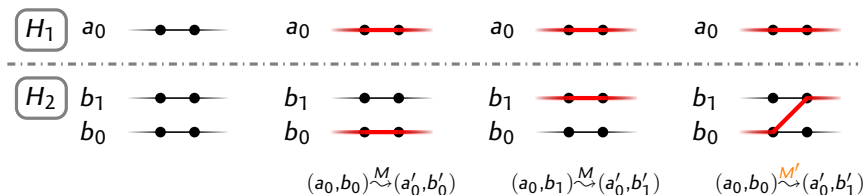
► Output: M, M' , s.t.

$$a'_0 = h_1^*(a_0, M) = h_1^*(a_0, M')$$

$$b'_1 = h_2^*(b_1, M) = h_2^*(b_0, M')$$

$$b'_0 = h_2^*(b_0, M) \neq b'_1$$

Switch structure



► Simple case: one H_1 -chain, and two H_2 -chains

► Input: a_0, b_0, b_1

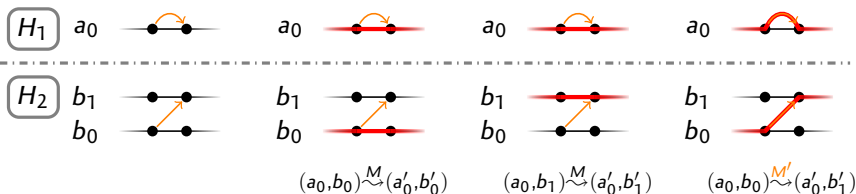
► Output: M, M' , s.t.

$$a'_0 = h_1^*(a_0, M) = h_1^*(a_0, M')$$

$$b'_1 = h_2^*(b_1, M) = h_2^*(b_0, M')$$

$$b'_0 = h_2^*(b_0, M) \neq b'_1$$

Switch structure



► Simple case: one H_1 -chain, and two H_2 -chains

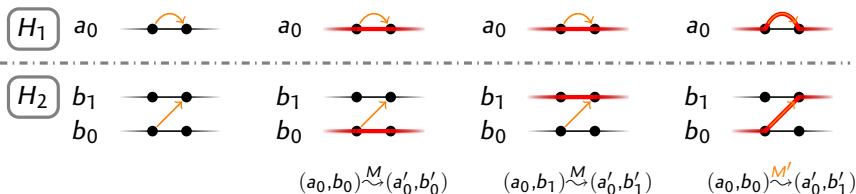
- Input: a_0, b_0, b_1
- Output: M, M' , s.t.

$$a_0' = h_1^*(a_0, M) = h_1^*(a_0, M')$$

$$b_1' = h_2^*(b_1, M) = h_2^*(b_0, M')$$

$$b_0' = h_2^*(b_0, M) \neq b_1'$$

Switch structure

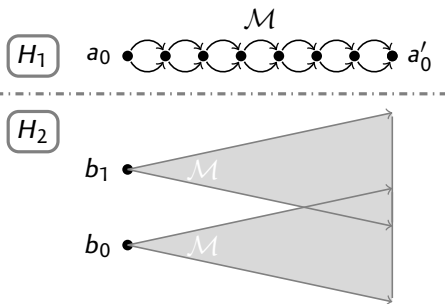
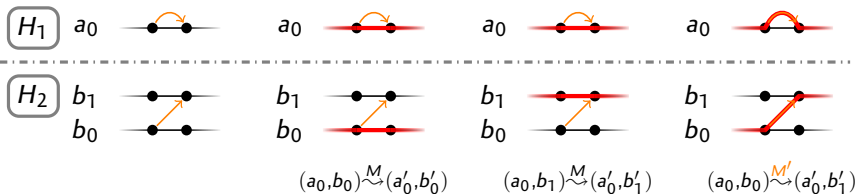


► Simple case: one H_1 -chain, and two H_2 -chains

- Input: a_0, b_0, b_1
- Output: M, M' , s.t.

$$\begin{aligned}
 a'_0 &= h_1^*(a_0, M) = h_1^*(a_0, M') \\
 b'_1 &= h_2^*(b_1, M) = h_2^*(b_0, M') \\
 b'_0 &= h_2^*(b_0, M) \neq b'_1
 \end{aligned}$$

Switch structure



1 Build multicollision \mathcal{M} for H_1

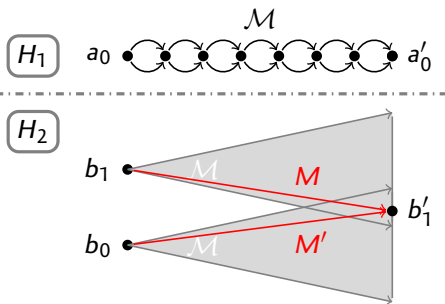
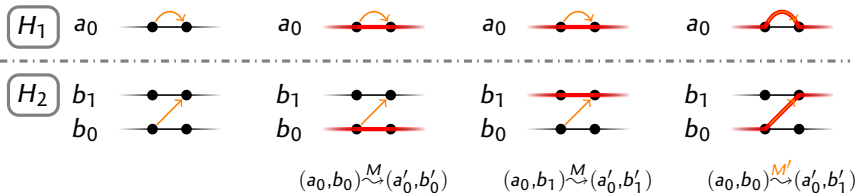
2 Select $M, M' \in \mathcal{M}$ s.t.
 $h_2^*(b_1, M) = h_2^*(b_0, M') \triangleq b'_1$

3 Set $a'_0 \triangleq h_1^*(a_0, M)$,

$$b'_0 \triangleq h_2^*(b'_1, M)$$

► Complexity $\approx n \cdot 2^{n/2}$

Switch structure



1 Build multicollision \mathcal{M} for H_1

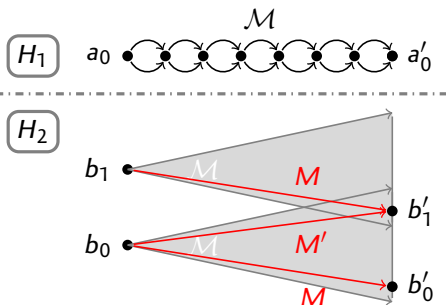
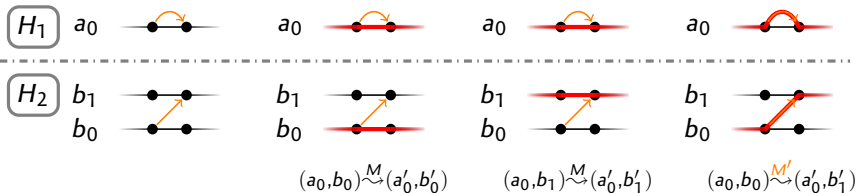
2 Select $M, M' \in \mathcal{M}$ s.t.
 $h_2^*(b_1, M) = h_2^*(b_0, M') \triangleq b'_1$

3 Set $a'_0 \triangleq h_1^*(a_0, M)$,

$$b'_0 \triangleq h_2^*(b_i, M)$$

► Complexity $\approx n \cdot 2^{n/2}$

Switch structure



- 1 Build multicollision \mathcal{M} for H_1
- 2 Select $M, M' \in \mathcal{M}$ s.t.

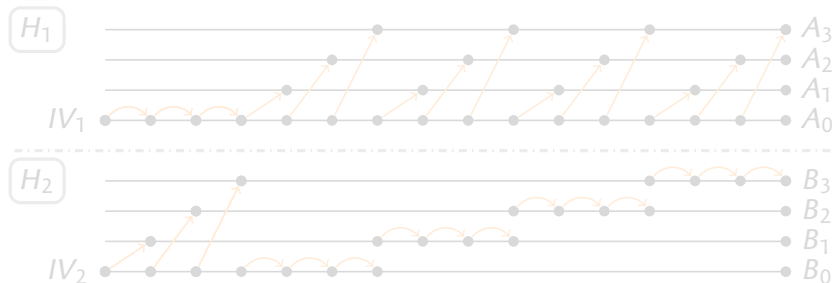
$$h_2^*(b_1, M) = h_2^*(b_0, M') \triangleq b'_1$$
- 3 Set $a'_0 \triangleq h_1^*(a_0, M)$,

$$b'_0 \triangleq h_2^*(b'_1, M)$$

► Complexity $\approx n \cdot 2^{n/2}$

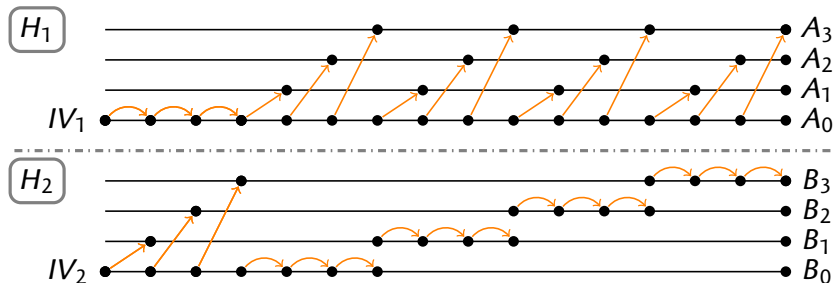
Switch structure

- ▶ We call this structure a **switch**. It can be used with more chains:
 - ▶ Update inactive chains with common message M
 - ▶ Jump from (a_j, b_k) to $(a_{j'}, b_k)$ or to $(a, b_{k'})$
 - ▶ Alternate message to be used only from (a_j, b_k) !
- ▶ Reach all chain combinations by combining several switches:
 - ▶ **Interchange structure:**

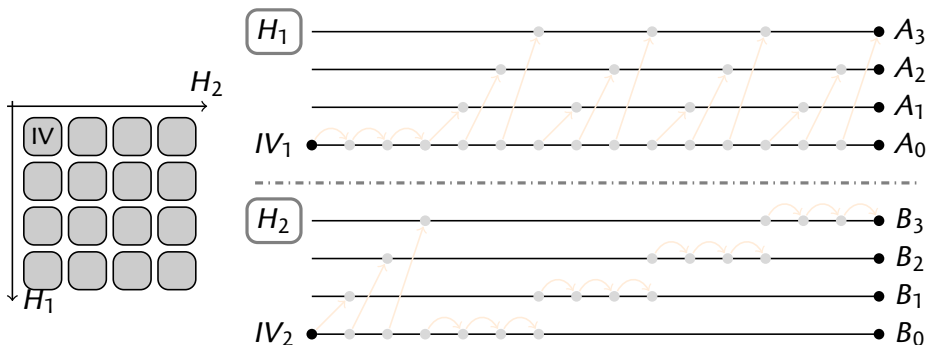


Switch structure

- ▶ We call this structure a **switch**. It can be used with more chains:
 - ▶ Update inactive chains with common message M
 - ▶ Jump from (a_j, b_k) to $(a_{j'}, b_k)$ or to $(a, b_{k'})$
 - ▶ Alternate message to be used only from (a_j, b_k) !
- ▶ Reach all chain combinations by combining several switches:
 - ▶ **Interchange structure:**

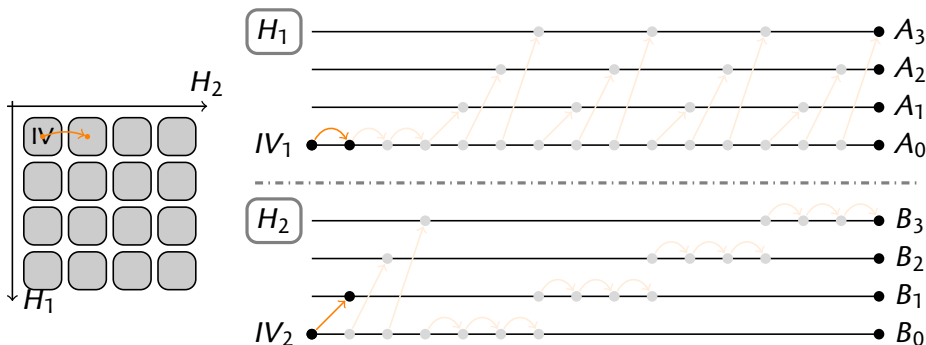


Interchange structure



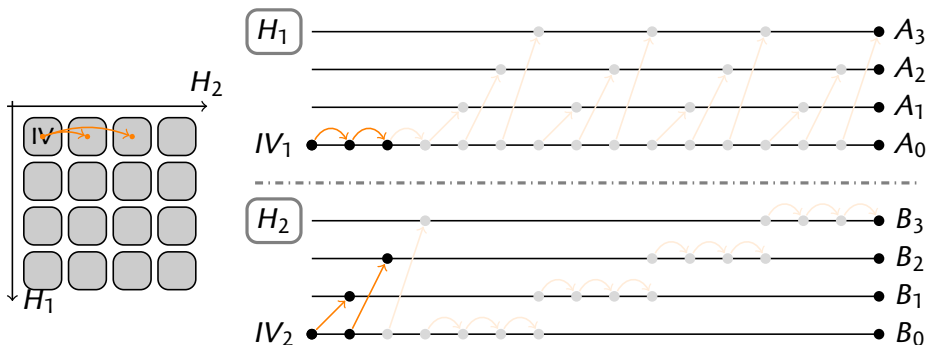
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



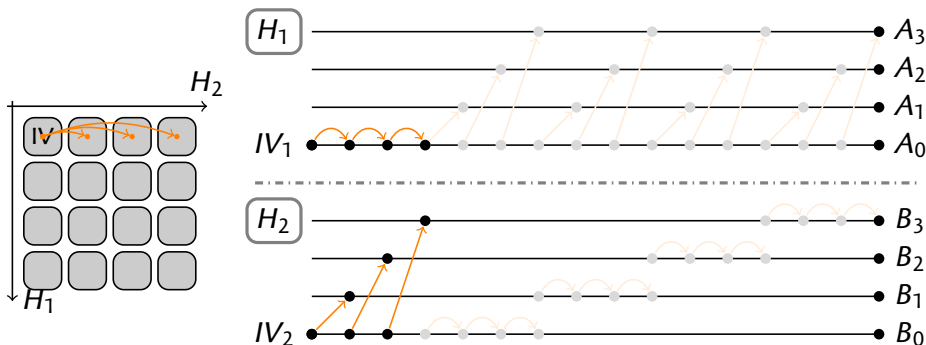
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



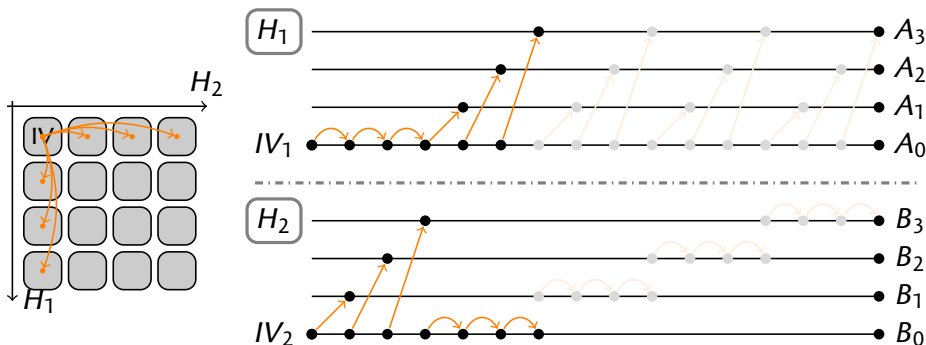
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



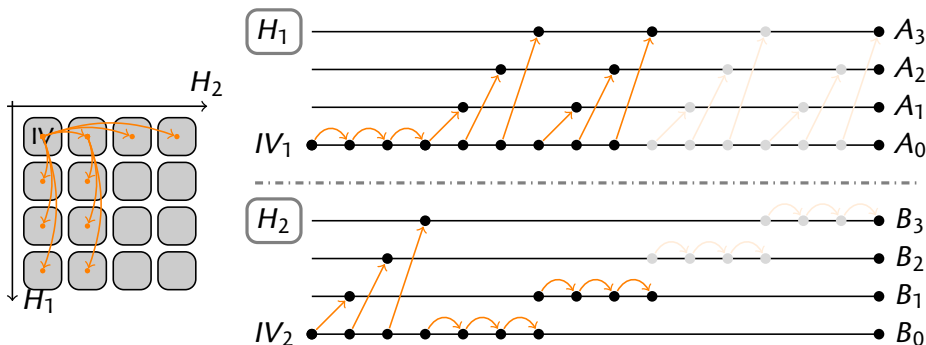
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



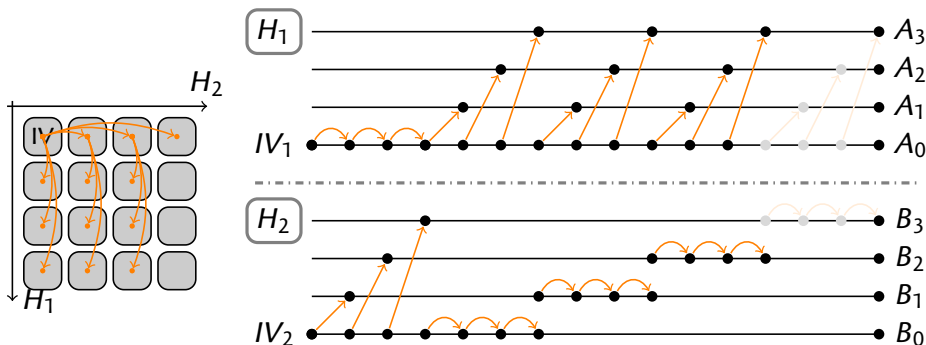
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



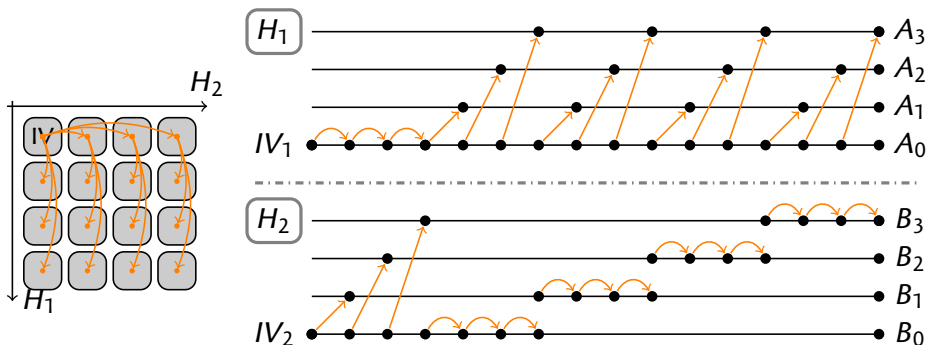
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



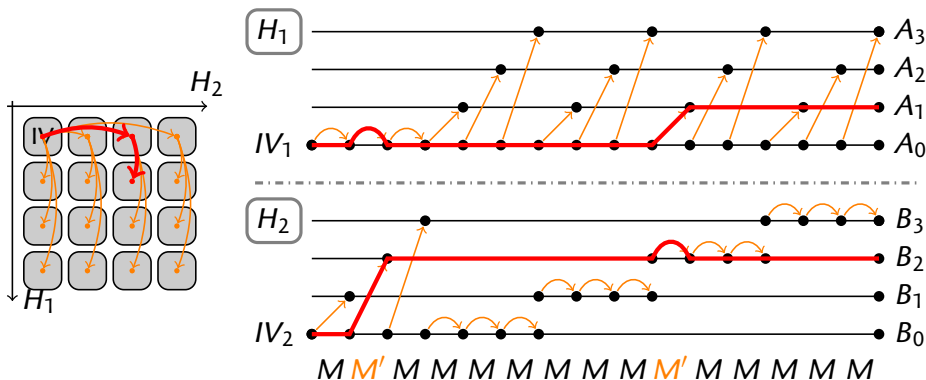
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



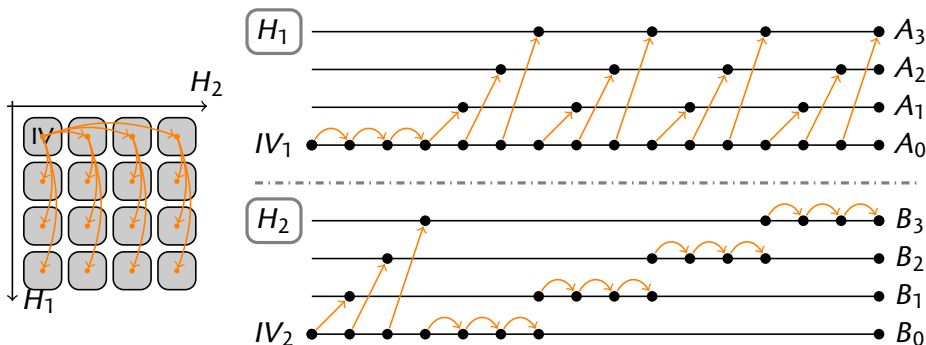
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



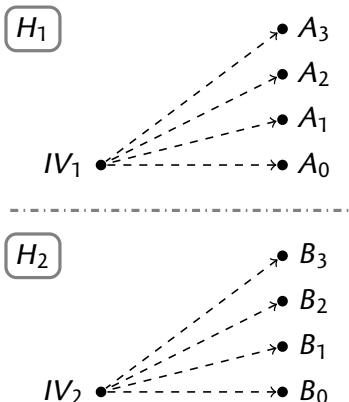
- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Interchange structure



- ▶ Control 2^t H_1 -chains and 2^t H_2 -chains using 2^{2t} switches
- ▶ Message length: $n/2 \cdot 2^{2t}$, memory: $n \cdot 2^{2t}$
- ▶ Time complexity: $n/2 \cdot 2^{n/2} \cdot 2^{2t}$

Preimage Attack



- 1 Build a 2^t -interchange structure $\{\mathbf{M}_{ij}\}$:

$$(IV_1, IV_2) \xrightarrow{\mathbf{M}_{jk}} (A_j, B_k)$$

- Complexity: $\tilde{O}(2^{2t} \cdot 2^{n/2})$

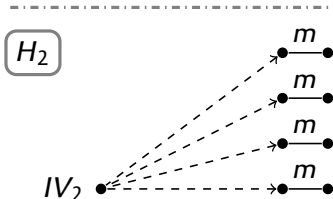
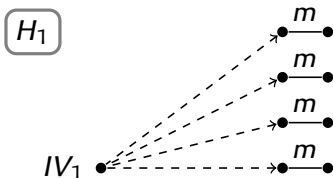
- 2 Preimage search for \bar{H} :

- For random blocks m , match $\{h_1(A_j, m)\}$ and $\{h_2(B_k, m) \oplus \bar{H}\}$
- If there is a match (i, j) :
Get $\mathbf{M}_{jk'}$, preimage is $\mathbf{M}_{jk} \parallel m$
- Complexity: $\tilde{O}(2^{n-t})$

- 3 Optimal complexity: $\mathcal{O}(n \cdot 2^{5n/6})$

- $t = n/6$

Preimage Attack



- 1 Build a 2^t -interchange structure $\{\mathbf{M}_{ij}\}$:

$$(IV_1, IV_2) \xrightarrow{\mathbf{M}_{jk}} (A_j, B_k)$$

- Complexity: $\tilde{O}(2^{2t} \cdot 2^{n/2})$

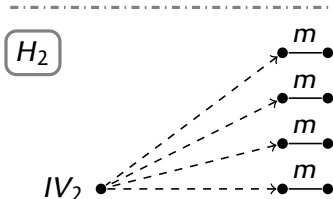
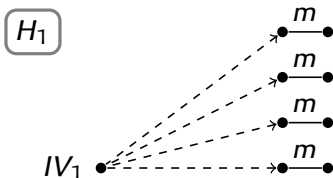
- 2 Preimage search for \bar{H} :

- For random blocks m , match $\{h_1(A_j, m)\}$ and $\{h_2(B_k, m) \oplus \bar{H}\}$
- If there is a match (i, j) :
Get $\mathbf{M}_{jk'}$, preimage is $\mathbf{M}_{jk} \parallel m$
- Complexity: $\tilde{O}(2^{n-t})$

- 3 Optimal complexity: $\mathcal{O}(n \cdot 2^{5n/6})$

- $t = n/6$

Preimage Attack



- 1 Build a 2^t -interchange structure $\{\mathbf{M}_{ij}\}$:

$$(IV_1, IV_2) \xrightarrow{\mathbf{M}_{jk}} (A_j, B_k)$$

- Complexity: $\tilde{O}(2^{2t} \cdot 2^{n/2})$

- 2 Preimage search for \bar{H} :

- For random blocks m , match $\{h_1(A_j, m)\}$ and $\{h_2(B_k, m) \oplus \bar{H}\}$
- If there is a match (i, j) :
Get $\mathbf{M}_{jk'}$, preimage is $\mathbf{M}_{jk} \parallel m$
- Complexity: $\tilde{O}(2^{n-t})$

- 3 Optimal complexity: $\mathcal{O}(n \cdot 2^{5n/6})$

- $t = n/6$

Extensions

- ▶ Works for the **HAIFA** mode
 - ▶ Finalization function, block counter at each round
- ▶ Works with **internal checksum** (GOST)
 - ▶ Using pairs of blocks with constant sum
- ▶ Works with $H_1(M) \boxplus H_2(M)$
 - ▶ Or any easy to invert operation
- ▶ For **wide-pipe** ($\ell > n$ bits of internal state), complexity $\ell/2 \cdot 2^{2n/3 + \ell/6}$
 - ▶ E.g. 2^{199} for $\text{SHA-224} \oplus \text{BLAKE-224}$
- ▶ Variants with **shorter messages**:
 - ▶ Time complexity 2^{n-m} with length 2^{2m} ; memory 2^{2m} ($m < n/6$)
- ▶ Can be extended to the **sum of three** or more (k) hash functions
 - ▶ Complexity $\mathcal{O}(n^{k-1} \cdot 2^{5n/6})$

Conclusion

- ▶ New technique to **control two hash functions** independently
 - ▶ Stronger than previous techniques (multi-collisions, diamonds)
 - ▶ **Preimage attack** for $H_1(M) \oplus H_2(M)$ with complexity: $n/2 \cdot 2^{5n/6}$
-
- ▶ **The sum of two good narrow-pipe hash functions is a bad hash function**
 - ▶ Whirlpool \oplus Streebog: complexity 2^{435} ($n = 512$)
 - ▶ SHA-512 \oplus Whirlpool: complexity 2^{461} ($n = 512$)
 - ▶ SHA-224 \oplus BLAKE-224: complexity 2^{199} ($n = 224$)