Cryptanalysis Beyond Primitives

Gaëtan Leurent

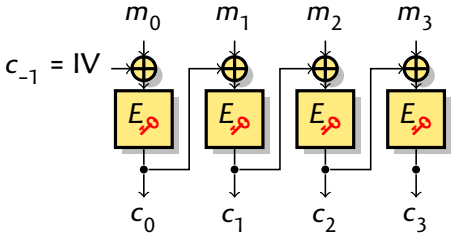
Inria, France

FSE 2024

Modes and primitives

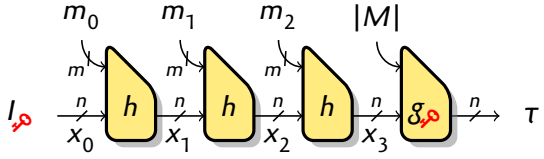
- Primitive with fixed-size inputs, and mode of operation

- Encryption example: CBC-AES



  - Mode: CBC
    - Encryption mode
  - Primitive: AES block cipher
    - $E : \{0,1\}^\kappa \times \{0,1\}^n \to \{0,1\}^n$
    - $E_k : x \mapsto E(k, x)$ permutation

- Authentication example: HMAC-SHA1



  - Mode: HMAC
    - Message Authentication Code (MAC)
  - Primitive: SHA-1 compression function
    - $h : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$
    - Public function without structural property

Security Analysis

Classical approach

- Security of the protocol (TLS, SSH, ...)
  - Security proofs, assuming security of cryptographic operations
- Security of the modes (HMAC, CBC, ...)
  - Security proofs, assuming security of the primitive
- Security of the primitives (AES, SHA-1, RSA, ...)
  - Studied with cryptanalysis

- Cryptanalysis usually applied inside primitives
- If this talk: cryptanalysis techniques outside the primitive

Cryptanalysis beyond primitives

Generic attacks

- Target the mode itself without using properties of the primitive
- Nice algorithmic problems & mathematical properties

1. Generic attacks define the expected security of a mode
   - Generic collision attack for hash functions and MACs
2. Generic attack are complementary to security proofs
   - Security proofs give lower bound on the security
   - Generic attacks give upper bound on the security
3. Generic attacks are important when the proof does not apply
   - What is the impact of going over the proof limit? (distinguisher or key-recovery)
   - What happens in a different model? (eg with quantum queries)
   - Can there be a mistake in the proof?

The birthday bound

- Security of common modes of operations is limited by collisions
- With an n-bit state, collisions after $2^{n/2}$ blocks

The birthday paradox

- Draw r random values from $\{0,1\}^n$
  - Expected number of collisions is about $r^2/2^{n+1}$
  - Constant probability of having a collision with $r = \Theta(2^{n/2})$
- Variant: Let $\mathcal{A}, \mathcal{B}$ be random subsets of $\{0,1\}^n$
  - Expected number of matches $|\mathcal{A} \cap \mathcal{B}| \approx |\mathcal{A}| \times |\mathcal{B}|/2^n$
  - In particular, $\mathcal{A} \cap \mathcal{B} \neq \emptyset$ with high probability if $|\mathcal{A}| = |\mathcal{B}| = 2^{n/2}$

- Many generic attacks are based on finding collisions between well-chosen sets

Outline

Generic Attacks Against Hash Functions

Generic Attacks Against Hash Combiners

Generic Attacks Against Hash-based MACs

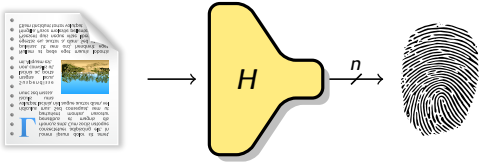
Generic Attacks Against Encryption Modes

Generic Attacks Against MACs in the Quantum Setting

Outline

Hash functions

- Public function $H : \{0,1\}^* \to \{0,1\}^n$
- Should behave like a random function
  - No structural property
  - Cryptographic properties without any key!
- Concrete security goals



Preimage attack

Given H and $\overline{X}$, find M s.t. $H(M) = \overline{X}$. Ideal security: 2^n.

Second-preimage attack

Given H and M_1, find $M_2 \neq M_1$ s.t. $H(M_1) = H(M_2)$. Ideal security: 2^n.

Collision attack

Given H, find $M_1 \neq M_2$ s.t. $H(M_1) = H(M_2)$. Ideal security: $2^{n/2}$.

Finding collisions

Collision expected after $\mathcal{O}(2^{n/2})$ evaluations with random inputs

Collision search in practice

- Sort data to avoid quadratic complexity
- Pollard's rho to avoid memory

Pollard's rho

- Given a function $H : \{0,1\}^n \to \{0,1\}^n$, find x, y with $H(x) = H(y)$

1. Iterate H: $x_i = H(x_{i-1})$
2. After roughly $2^{n/2}$ iterations, sequence cycles
3. Detect cycle, locate collision (Floyd, Brent)

- Complexity $\mathcal{O}(2^{n/2})$

Finding collisions

Collision expected after $\mathcal{O}(2^{n/2})$ evaluations with random inputs

Collision search in practice

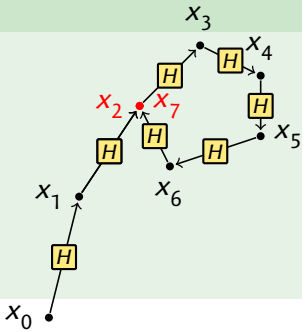
- Sort data to avoid quadratic complexity
- Pollard's rho to avoid memory

Pollard's rho

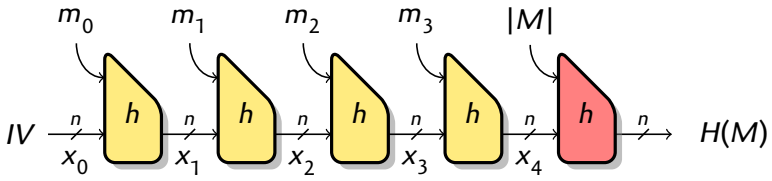
- Given a function $H : \{0,1\}^n \to \{0,1\}^n$, find x, y with $H(x) = H(y)$

1. Iterate H: $x_i = H(x_{i-1})$
2. After roughly $2^{n/2}$ iterations, sequence cycles
3. Detect cycle, locate collision (Floyd, Brent)

- Complexity $\mathcal{O}(2^{n/2})$



The Merkle-Damgård construction (SHA-1, SHA-2)



- n-bit state, compression function $h : \{0,1\}^n \times \{0,1\}^r \to \{0,1\}^n$
- Finalization using message length (MD strengthening)
- Notation: Iterated compression function h^*
  - $h^*(x, m_1 \parallel m_2 \parallel m_3) = h(h(h(x, m_1), m_2), m_3)$
- Security reduction:
  - Hash collisions imply compression function collision (generic security $2^{n/2}$)
  - Hash preimages imply finalization preimages (generic security 2^n)

Generic attacks on Merkle-Damgård

Many properties "between" collision and preimage broken with birthday complexity, by generic attacks exploiting collisions in smart ways

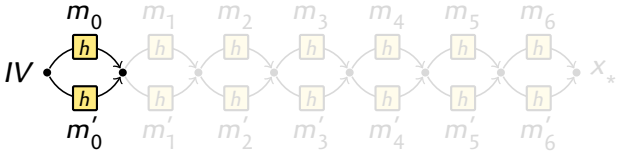
Multicollision	[Joux, Crypto '04]
Find a large set of message $\{M_i\}$ s.t. $\forall i,\ H(M_i) = H(M_0)$	Complexity $\tilde{\mathcal{O}}(2^{n/2})$

Chosen-prefix collision	[Stevens, Lenstra & de Weger, EC'07]
Given challenges C, C', find M, M' s.t. $H(C \parallel M) = H(C' \parallel M')$	Complexity $\mathcal{O}(2^{n/2})$

Diamond structure	[Kelsey & Kohno, EC'06]
Given challenges $\{C_i\}$, find $\{M_i\}$ s.t. $\forall i,\ H(C_i \parallel M_i) = H(C_0 \parallel M_0)$	Complexity $\tilde{\mathcal{O}}(\sqrt{\lvert\{C_i\}\rvert}\,2^{n/2})$

Multicollisions

[Joux, Crypto '04]



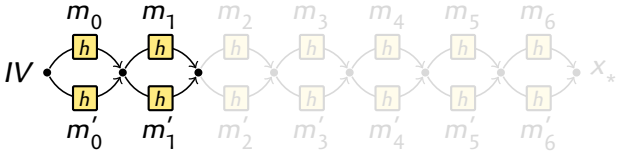
1. Find a collision pair m_0/m'_0 starting from IV
2. Find a collision pair m_1/m'_1 starting from $x_1 = h^*(m_0)$
3. Repeat t times
4. This yields 2^t messages with the same hash:

$$m_0 m_1 m_2 \ldots \quad m'_0 m_1 m_2 \ldots \quad m_0 m'_1 m_2 \ldots \quad m'_0 m'_1 m_2 \ldots$$
$$m_0 m_1 m'_2 \ldots \quad m'_0 m_1 m'_2 \ldots \quad m_0 m'_1 m'_2 \ldots \quad m'_0 m'_1 m'_2 \ldots$$

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

Multicollisions

[Joux, Crypto '04]



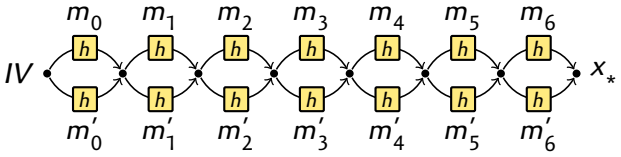
1. Find a collision pair m_0 / m'_0 starting from IV
2. Find a collision pair m_1 / m'_1 starting from $x_1 = h^*(m_0)$
3. Repeat t times
4. This yields 2^t messages with the same hash:

$$m_0 m_1 m_2 \ldots \quad m'_0 m_1 m_2 \ldots \quad m_0 m'_1 m_2 \ldots \quad m'_0 m'_1 m_2 \ldots$$
$$m_0 m_1 m'_2 \ldots \quad m'_0 m_1 m'_2 \ldots \quad m_0 m'_1 m'_2 \ldots \quad m'_0 m'_1 m'_2 \ldots$$

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

Multicollisions

[Joux, Crypto '04]

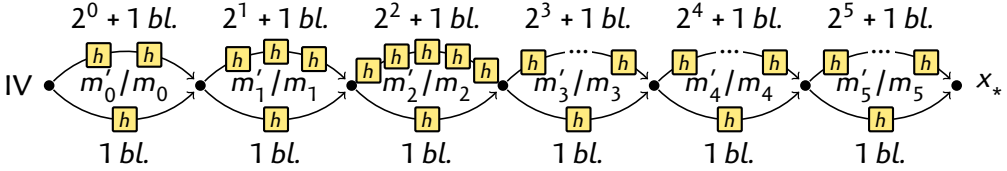


1. Find a collision pair m_0 / m'_0 starting from IV
2. Find a collision pair m_1 / m'_1 starting from $x_1 = h^*(m_0)$
3. Repeat t times
4. This yields 2^t messages with the same hash:

$$m_0 m_1 m_2 \ldots \quad m'_0 m_1 m_2 \ldots \quad m_0 m'_1 m_2 \ldots \quad m'_0 m'_1 m_2 \ldots$$
$$m_0 m_1 m'_2 \ldots \quad m'_0 m_1 m'_2 \ldots \quad m_0 m'_1 m'_2 \ldots \quad m'_0 m'_1 m'_2 \ldots$$

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

Expandable message [Kelsey & Schneier, Eurocrypt '05]

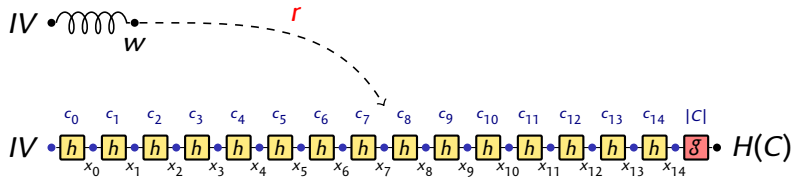


- Multicollision with messages of difference length
  2^t messages of length $t, t+1, \ldots t+2^t-1$ blocks
  - Length 0+6: $m_0 m_1 m_2 m_3 m_4 m_5$
  - Length 1+6: $m'_0 m_1 m_2 m_3 m_4 m_5$
  - Length 2+6: $m_0 m'_1 m_2 m_3 m_4 m_5$
  - Length 3+6: $m'_0 m'_1 m_2 m_3 m_4 m_5$
  - ...
  - Length 63+6: $m'_0 m'_1 m'_2 m'_3 m'_4 m'_5$
- Complexity $t \cdot 2^{n/2}$

Second-preimage for long challenges [Kelsey & Schneier, Eurocrypt '05]

Given a challenge C, find $M \neq C$ with $H(M) = H(C)$ $\quad \text{len}(C) = 2^s$

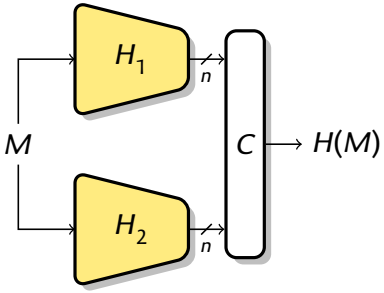
0. Build expandable message $\mathcal{M}$ of length 2^s (final state w)
1. Compute internal states $\{x_i\}$ for $H(C)$ (No key: public values)
2. Find r, i with $h(w, r) = x_i$ (Complexity 2^{n-s})
3. Preimage is $\mathcal{M}_{i-1} \parallel r \parallel C[i:]$



- Complexity $2^s + 2^{n-s}$ ($2^{n/2}$ for $s = n/2$)

Outline

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:

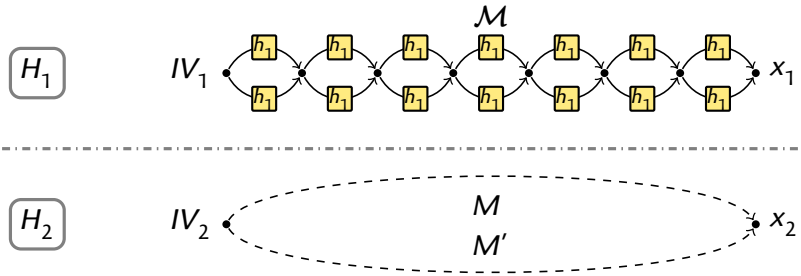
- Concatenation: $H_1(M) \| H_2(M)$
- Xor: $H_1(M) \oplus H_2(M)$

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

– Aristotle

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

[Joux, C'04]



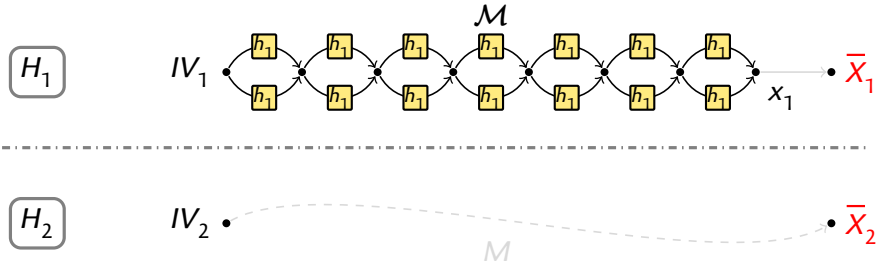
1. Build a $2^{n/2}$-multicollision for H_1

$$\forall M \in \mathcal{M}, H_1(M) = x_1$$

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

- Complexity $\tilde{\mathcal{O}}(2^{n/2})$ vs. 2^n for a $2n$-bit hash function.

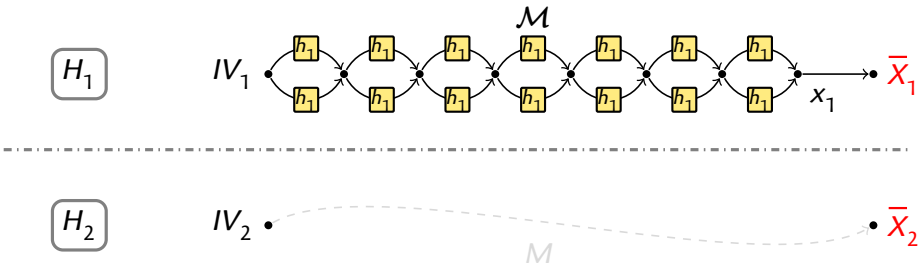
Preimage attack for $H_1(M) \| H_2(M)$ [Joux, C'04]



1. Build a 2^n-multicollision for H_1 — $\forall M \in \mathcal{M}, h_1^*(M) = x_1$
2. Find a preimage for H_1: $g(h(x_1, r)) = \overline{X}_1$ — $\forall M \in \mathcal{M}, H_1(M) = \overline{X}_1$
3. Find $M \in \mathcal{M}$ s.t. $H_2(M \| r) = \overline{X}_2$

- Complexity $\tilde{\mathcal{O}}(2^n)$ vs. 2^{2n} for a $2n$-bit hash function.

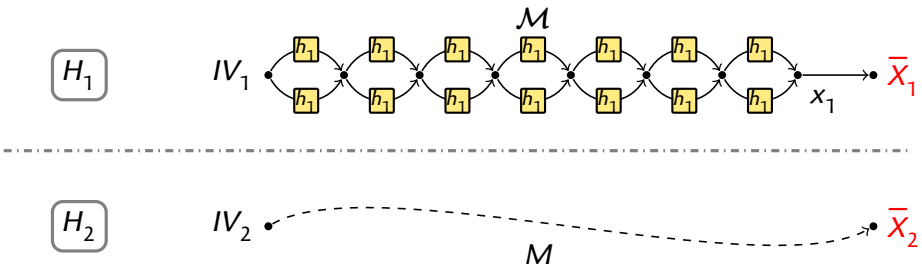
Preimage attack for $H_1(M) \| H_2(M)$ [Joux, C'04]



1. Build a 2^n-multicollision for H_1 $\quad \forall M \in \mathcal{M}, h_1^*(M) = x_1$
2. Find a preimage for H_1: $g(h(x_1, r)) = \overline{X}_1$ $\quad \forall M \in \mathcal{M}, H_1(M) = \overline{X}_1$
3. Find $M \in \mathcal{M}$ s.t. $H_2(M \| r) = \overline{X}_2$

- Complexity $\tilde{\mathcal{O}}(2^n)$ vs. 2^{2n} for a $2n$-bit hash function.

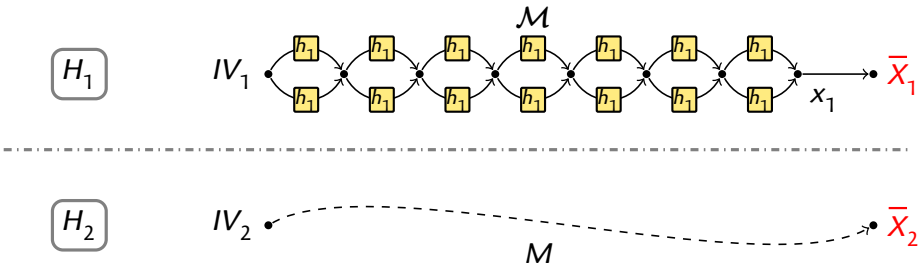
Preimage attack for $H_1(M) \| H_2(M)$ [Joux, C'04]



1. Build a 2^n-multicollision for H_1 $\quad \forall M \in \mathcal{M}, h_1^*(M) = x_1$
2. Find a preimage for H_1: $g(h(x_1, r)) = \overline{X}_1$ $\quad \forall M \in \mathcal{M}, H_1(M) = \overline{X}_1$
3. Find $M \in \mathcal{M}$ s.t. $H_2(M \| r) = \overline{X}_2$

- Complexity $\tilde{\mathcal{O}}(2^n)$ vs. 2^{2n} for a $2n$-bit hash function.

Preimage attack for $H_1(M) \| H_2(M)$ [Joux, C'04]



1. Build a 2^n-multicollision for H_1 $\quad \forall M \in \mathcal{M}, h_1^*(M) = x_1$
2. Find a preimage for H_1: $g(h(x_1, r)) = \overline{X}_1$ $\quad \forall M \in \mathcal{M}, H_1(M) = \overline{X}_1$
3. Find $M \in \mathcal{M}$ s.t. $H_2(M \| r) = \overline{X}_2$

- Complexity $\tilde{\mathcal{O}}(2^n)$ vs. 2^{2n} for a $2n$-bit hash function.

Combining generic and dedicated cryptanalysis

Collision for MD5 ‖ SHA-1

- Using a SHA-1 multicollision (dedicated SHA-1 collision attack)
- Complexity 2^{64} for generic MD5 collision
- ⇒ Complexity $2^{67.6}$ for collision and chosen-prefix collision
- Chosen-prefix collision attack on MD5 ‖ SHA-1 breaks TLS 1.0/1.1 [SLOTH]

Partial preimage for MD5 ‖ SHA-1

- Using a MD5 multicollision (dedicated MD5 collision attack)
- Complexity 2^t for generic partial preimage (fixing t bits)
- ⇒ Complexity $\approx 2^t$ for fixing t bits of MD5 and t bits of SHA-1
- `f5e2024` challenge (rump session)

Generic attacks against combiners

Concatenation combiner

- $H(M) = H_1(M) \| H_2(M)$
- $2n$-bit output
- Generic security: attacks / proofs
  - Collisions: $2^{n/2}$ / $2^{n/2}$
  - Preimages: 2^n / 2^n
  - Non-ideal: $2^{n/2}$ / $2^{n/2}$

XOR combiner

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

Surprising result [Joux, C'04]

If H_1 and H_2 are good MD hash functions, $H_1 \| H_2$ is not stronger!

Generic attacks against combiners

Concatenation combiner

- $H(M) = H_1(M) \| H_2(M)$
- $2n$-bit output
- Generic security: attacks / proofs
  - Collisions: $2^{n/2}$ / $2^{n/2}$
  - Preimages: 2^n / 2^n
  - Non-ideal: $2^{n/2}$ / $2^{n/2}$

XOR combiner

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

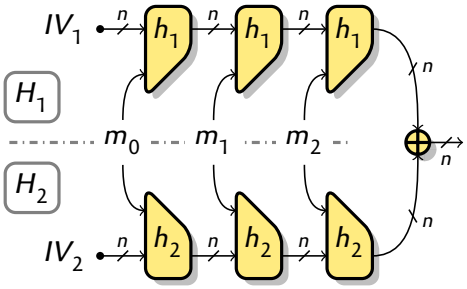
Surprising result [Joux, C'04]

If H_1 and H_2 are good MD hash functions, $H_1 \| H_2$ is not stronger!

Preimage attack against Xor combiner

[L & Wang, EC'15]

$H(M) = H_1(M) \oplus H_2(M)$

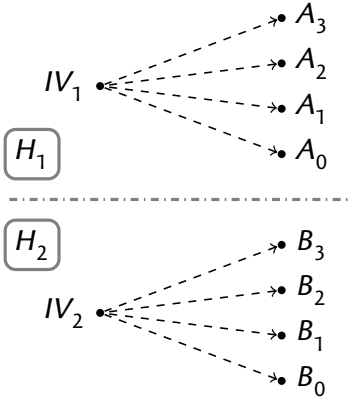


Strategy:

1. Structure to control H_1 and H_2 independently:
   - Sets of states $\mathcal{A} = \{A_j\}$, $\mathcal{B} = \{B_k\}$
   - Set of messages $\{\mathbf{M}_{jk}\}$ with
     $$h_1^*(\mathbf{M}_{jk}) = A_j$$
     $$h_2^*(\mathbf{M}_{jk}) = B_k$$
2. Preimage search for $\overline{X}$:
   - For random blocks r, match $\{g_1(h_1(A_j, r))\}$ and $\{g_2(h_2(B_k, r)) \oplus \overline{X}\}$
   - If there is a match (j, k):
     Get $\mathbf{M}_{jk'}$ preimage is $M = \mathbf{M}_{jk} \| r$
   - Complexity $\mathcal{O}(2^n / \min\{|\mathcal{A}|, |\mathcal{B}|\})$

Preimage attack against Xor combiner [L & Wang, EC'15]

$H(M) = H_1(M) \oplus H_2(M)$

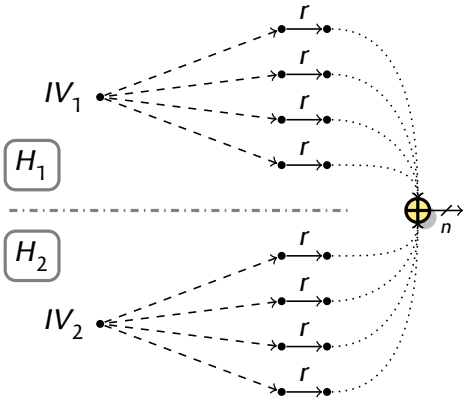


Strategy:

1. Structure to control H_1 and H_2 independently:
   - Sets of states $\mathcal{A} = \{A_j\}$, $\mathcal{B} = \{B_k\}$
   - Set of messages $\{\mathbf{M}_{jk}\}$ with
     $$h_1^*(\mathbf{M}_{jk}) = A_j$$
     $$h_2^*(\mathbf{M}_{jk}) = B_k$$
2. Preimage search for $\overline{X}$:
   - For random blocks r, match $\{g_1(h_1(A_j, r))\}$ and $\{g_2(h_2(B_k, r)) \oplus \overline{X}\}$
   - If there is a match (j, k): Get $\mathbf{M}_{jk'}$ preimage is $M = \mathbf{M}_{jk} \| r$
   - Complexity $\mathcal{O}(2^n / \min\{|\mathcal{A}|, |\mathcal{B}|\})$

Preimage attack against Xor combiner [L & Wang, EC'15]

$H(M) = H_1(M) \oplus H_2(M)$

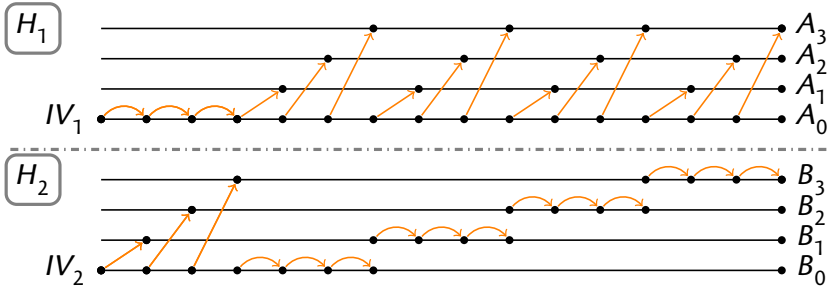


Strategy:

1. Structure to control H_1 and H_2 independently:
   - Sets of states $\mathcal{A} = \{A_j\}$, $\mathcal{B} = \{B_k\}$
   - Set of messages $\{\mathbf{M}_{jk}\}$ with
     $$h_1^*(\mathbf{M}_{jk}) = A_j$$
     $$h_2^*(\mathbf{M}_{jk}) = B_k$$
2. Preimage search for $\overline{X}$:
   - For random blocks r, match $\{g_1(h_1(A_j, r))\}$ and $\{g_2(h_2(B_k, r)) \oplus \overline{X}\}$
   - If there is a match (j, k): Get $\mathbf{M}_{jk}$, preimage is $M = \mathbf{M}_{jk} \,\|\, r$
   - Complexity $\mathcal{O}(2^n / \min\{|\mathcal{A}|, |\mathcal{B}|\})$

Interchange structure [L & Wang, EC'15]

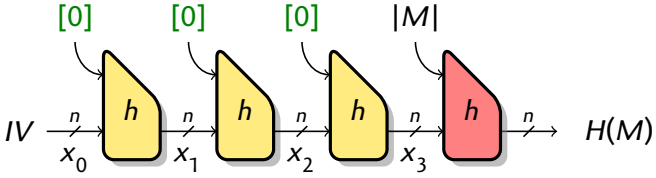
- Interchange structure for a large set of output states



- Complexity $\tilde{\mathcal{O}}(2^{n/2+2t})$ to build a structure with $|\mathcal{A}| = |\mathcal{B}| = 2^t$
- Complexity $\tilde{\mathcal{O}}(2^{5n/6})$ for preimages (tradeoff)

Alternative structure using cycles

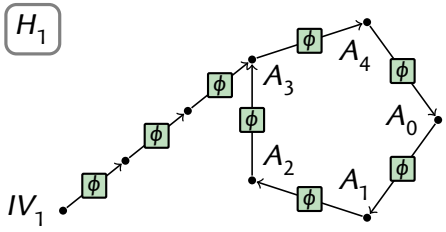
- Alternative presentation of "multicycles" [Bao, Wang, Guo, Gu, C'17]

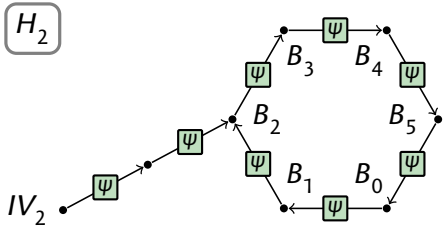


- Using a long message repeating a fixed block $M = [0]^\lambda$, we iterate fixed functions:

$$\phi : x \mapsto h_1(x, [0])$$
$$\psi : x \mapsto h_2(x, [0])$$

Alternative structure using cycles





- Use cyclic nodes as end-point:
  - $\mathcal{A} = H_1$ cycle, length ℓ_1
  - $\mathcal{B} = H_2$ cycle, length ℓ_2
- With suitable naming, for λ large enough:

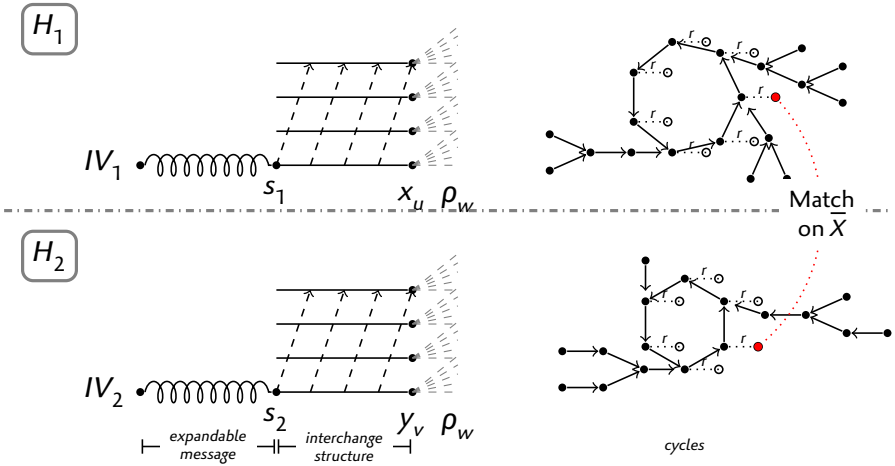
$$h_1^*([0]^\lambda) = A_{\lambda \bmod \ell_1} \qquad h_2^*([0]^\lambda) = B_{\lambda \bmod \ell_2}$$

- To reach (A_j, B_k), use Chinese Remainder

$$\begin{cases} h_1^*([0]^\lambda) = A_j \\ h_2^*([0]^\lambda) = B_k \end{cases} \iff \begin{cases} \lambda \bmod \ell_1 = i \\ \lambda \bmod \ell_2 = j \end{cases}$$

  - λ uniformly distributed in range of size $\ell_1 \ell_2$
  - $\Pr[\lambda < 2^t] \approx 2^{n-t}$
- Complexity $\tilde{\mathcal{O}}(2^{3n/4})$ for preimages (tradeoff)

Advanced preimage attack [ePrint 2024/488]



- Using interchange, small cycles, expandable message
- Complexity $\tilde{\mathcal{O}}(2^{3n/5})$

Summary: Preimage attack for $H_1(M) \oplus H_2(M)$

Interchange structure

- Complexity $\tilde{\mathcal{O}}(2^{5n/6})$ [LW15]

- Works for Merkle-Damgård and HAIFA
  - Finalization function, block counter at each round
- Short messages: length $\tilde{\mathcal{O}}(2^{n/3})$

Using cycles

- Complexity $\tilde{\mathcal{O}}(2^{3n/4})$ (simple)
- Complexity $\tilde{\mathcal{O}}(2^{5n/8})$ [BWGG17]
- Complexity $\tilde{\mathcal{O}}(2^{11n/18})$ [BDGLW20]
- Complexity $\tilde{\mathcal{O}}(2^{3n/5})$ [ePrint 2024/488]

- Works only for Merkle-Damgård mode
  - Finalization function, same function at each step
- Long messages: length $\tilde{\mathcal{O}}(2^{3n/5})$

Generic attacks against combiners

Concatenation combiner

- $H(M) = H_1(M) \| H_2(M)$
- $2n$-bit output
- Generic security: attacks / proofs
  - Collisions: $2^{n/2}$ / $2^{n/2}$
  - Preimages: 2^{n} / 2^{n}
  - Non-ideal: $2^{n/2}$ / $2^{n/2}$

XOR combiner

- $H(M) = H_1(M) \oplus H_2(M)$
- n-bit output
- Generic security: attacks / proofs
  - Collisions: $2^{n/2}$ / $2^{n/2}$
  - Preimages: $2^{3n/5}$ / $2^{n/2}$
  - Non-ideal: $2^{n/2}$ / $2^{n/2}$

Surprising result [Joux, C'04]

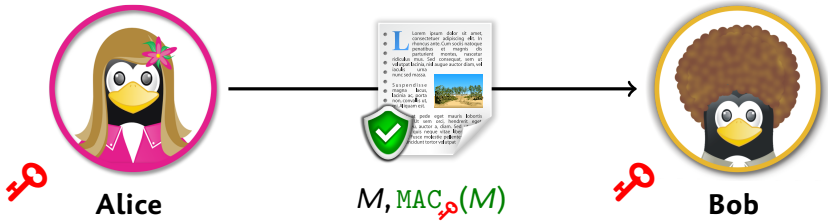
If H_1 and H_2 are good MD hash functions, $H_1 \| H_2$ is not stronger!

Surprising result [L & Wang, EC'15]

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

Outline

Message Authentication Codes (MAC)



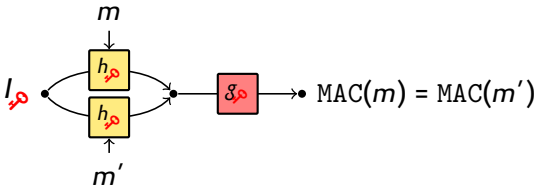
Security Notions

Forgery	Given access to a MAC oracle, forge a valid pair
State recovery	Recover the internal state, for a chosen message
Key recovery	Given access to a MAC oracle, extract the key

Generic forgery attack

[Preneel & van Oorschot '95]

- Consider an iterated deterministic MACs with n-bit state



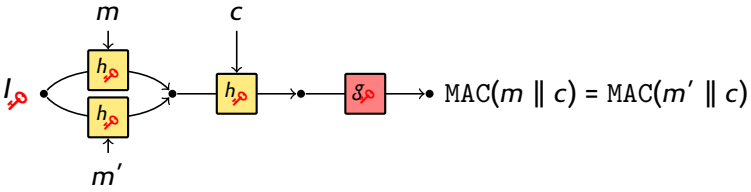
1. Find internal collisions
   - Query $2^{n/2}$ random short messages
   - 1 internal collision expected, detected in the output
2. Query $t = \mathtt{MAC}(m \parallel c)$
3. $(m' \parallel c, t)$ is a forgery

- Attack motivates MACs with security beyond the birthday bound
  - Non-deterministic: RMAC, Wegman-Carter
  - Larger state: SUM-ECBC, HMAC

Generic forgery attack

[Preneel & van Oorschot '95]

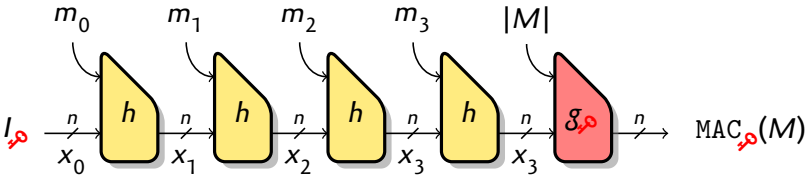
- Consider an iterated deterministic MACs with n-bit state



1. Find internal collisions
   - Query $2^{n/2}$ random short messages
   - 1 internal collision expected, detected in the output
2. Query $t = \texttt{MAC}(m \,\|\, c)$
3. $(m' \,\|\, c, t)$ is a forgery

- Attack motivates MACs with security beyond the birthday bound
  - Non-deterministic: RMAC, Wegman-Carter
  - Larger state: SUM-ECBC, HMAC

Hash-based MACs



- n-bit chaining value, n-bit MAC
- κ-bit key

we focus on $n = \kappa$

- Key-dependent initial value I_k
- Unkeyed compression function h
- Key-dependent finalization, with message length g_k

- Examples: HMAC, envelope MAC, sandwich MAC
- Security proofs up to the birthday bound

Secret-suffix MAC

Definition (Secret-suffix MAC)

$\texttt{MAC}_k(M) = H(M \,\|\, k)$

- Birthday security (tight)
- Birthday key-recovery attack [Preneel & van Oorschot '96]
  1. Guess the first key byte as k^*
  2. Find a one-block hash collision (C_0, C_1) with $C_i = M_i \,\|\, k^*$ (offline)

$$C_1 = \boxed{??? \cdots ??? \; k^*} \qquad M_1 = ???\cdots???$$

$$C_0 = \boxed{\text{¿¿¿} \cdots \text{¿¿¿} \; k^*} \qquad M_0 = \text{¿¿¿}\cdots\text{¿¿¿}$$

  3. Query $\texttt{MAC}(M_1)$ and $\texttt{MAC}(M_2)$ (online)

$$\texttt{MAC}(M_1) = H\Big(\boxed{??? \cdots ??? \; k_0} \; \boxed{k_1 k_2 k_3 \ldots} \Big)$$

$$\texttt{MAC}(M_0) = H\Big(\boxed{\text{¿¿¿} \cdots \text{¿¿¿} \; k_0} \; \boxed{k_1 k_2 k_3 \ldots} \Big)$$

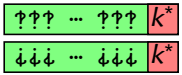
  4. If the MACs are equal, the guess was correct
- Practical attack when using MD5 (*e.g.* APOP) [L'07, Sasaki & al '08]
  - Using cryptanalytic shortcuts

Secret-suffix MAC

Definition (Secret-suffix MAC)

$\texttt{MAC}_k(M) = H(M \| k)$

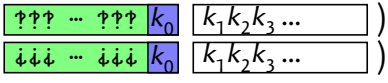
- Birthday security (tight)
- Birthday key-recovery attack [Preneel & van Oorschot '96]
  1. Guess the first key byte as k^*
  2. Find a one-block hash collision (C_0, C_1) with $C_i = M_i \| k^*$ (offline)



M_1 = ???···???

M_0 = ¿¿¿···¿¿¿

  3. Query $\texttt{MAC}(M_1)$ and $\texttt{MAC}(M_2)$ (online)



  4. If the MACs are equal, the guess was correct

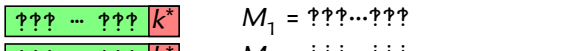
- Practical attack when using MD5 (*e.g.* APOP) [L'07, Sasaki & al '08]
  - Using cryptanalytic shortcuts

Secret-suffix MAC

Definition (Secret-suffix MAC)

$\texttt{MAC}_k(M) = H(M \parallel k)$

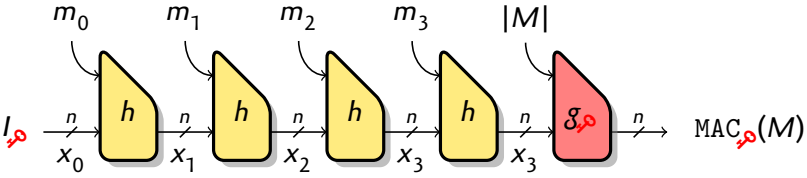
- Birthday security (tight)
- Birthday key-recovery attack [Preneel & van Oorschot '96]
  1. Guess the first key byte as k^*
  2. Find a one-block hash collision (C_0, C_1) with $C_i = M_i \parallel k^*$ (offline)



  3. Query $\texttt{MAC}(M_1)$ and $\texttt{MAC}(M_2)$ (online)

  4. If the MACs are equal, the guess was correct
- Practical attack when using MD5 (*e.g.* APOP) [L'07, Sasaki & al '08]
  - Using cryptanalytic shortcuts

Cycle-based forgery [L, Peyrin, Wang, AC'13]



- Using a long message repeating a fixed block $M = [0]^\lambda$, we iterate a fixed function

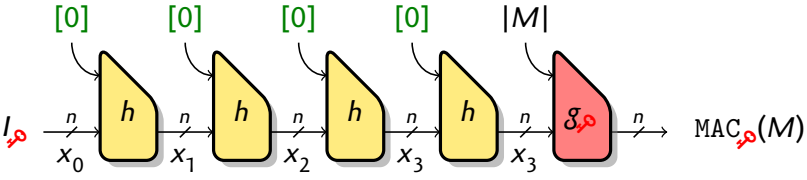
$$\phi : x \mapsto h(x, [0])$$

- Starting point and ending point unknown because of the key

Can we detect properties of ϕ? [Flajolet & Odlyzko, EC'89]

- Largest component size: $\approx 0.76 \times 2^n$
- Largest tree size: $\approx 0.48 \times 2^n$

Cycle-based forgery [L, Peyrin, Wang, AC'13]



- Using a long message repeating a fixed block $M = [0]^\lambda$, we iterate a fixed function

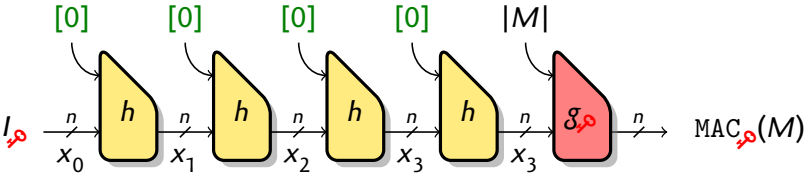
$$\phi : x \mapsto h(x, [0])$$

- Starting point and ending point unknown because of the key

Can we detect properties of ϕ? [Flajolet & Odlyzko, EC'89]

- Largest component size: $\approx 0.76 \times 2^n$
- Largest tree size: $\approx 0.48 \times 2^n$

Cycle-based forgery [L, Peyrin, Wang, AC'13]



- Using a long message repeating a fixed block $M = [0]^\lambda$, we iterate a fixed function

$$\phi : x \mapsto h(x, [0])$$

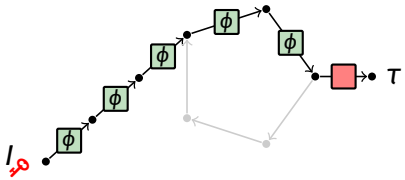
- Starting point and ending point unknown because of the key

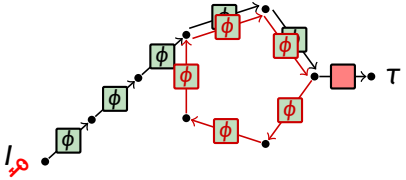
Can we detect properties of ϕ? [Flajolet & Odlyzko, EC'89]

- Largest component size: $\approx 0.76 \times 2^n$
- Largest tree size: $\approx 0.48 \times 2^n$

Using the cycle length

1. Offline: find the cycle length ℓ of the main component of ϕ
2. Online: query $t = \texttt{MAC}([0]^{2^{n/2}})$ and $t' = \texttt{MAC}([0]^{2^{n/2}+\ell})$



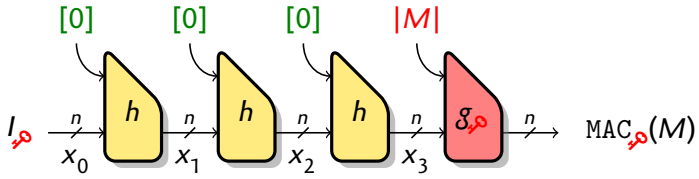


Collision if

- The starting point is in the main component — $p = 0.76$
- The cycle is reached with less than $2^{n/2}$ iterations — $p \geq 0.5$

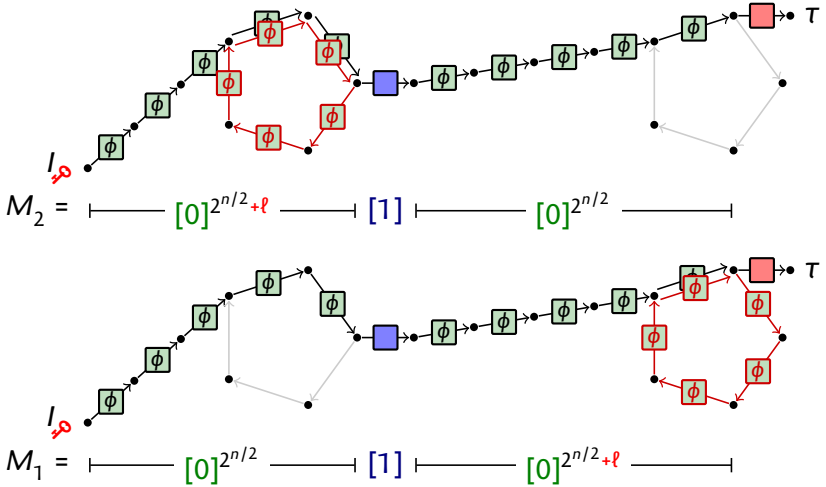
Dealing with the message length

- Problem: MACs use msg length.
- Solution: reach the cycle twice
  - Block [1] moves outside graph
  - With high proba. $\mathtt{MAC}_{k,\rho}(M_1) = \mathtt{MAC}_{k,\rho}(M_2)$
  - Surprisingly powerful: breaks universal hash



Dealing with the message length

- Problem: MACs use msg length.
- Solution: reach the cycle twice
  - Block [1] moves outside graph

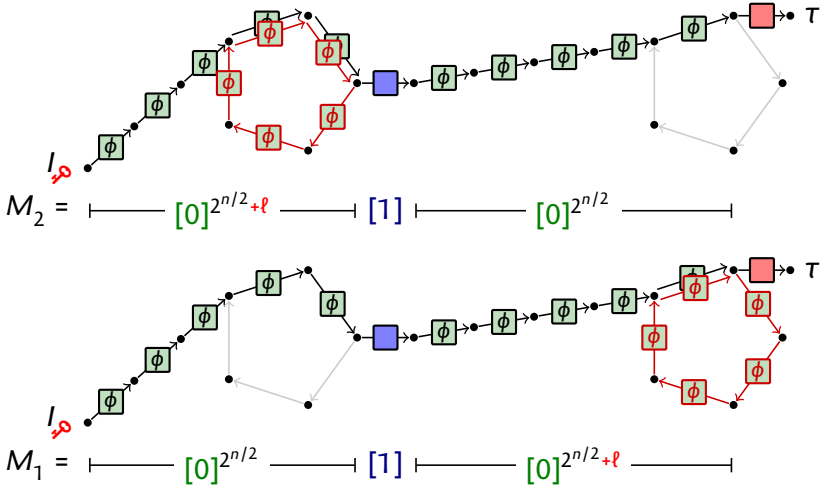


- With high proba. $\mathtt{MAC}_{\mathcal{P}}(M_1) = \mathtt{MAC}_{\mathcal{P}}(M_2)$
- Surprisingly powerful: breaks universal hash

Dealing with the message length

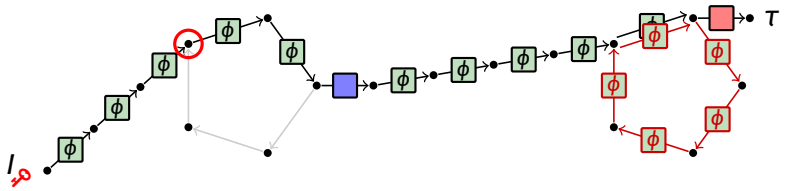
- Problem: MACs use msg length.
- Solution: reach the cycle twice
  - Block [1] moves outside graph

- With high proba. $\mathtt{MAC}_{\$}(M_1) = \mathtt{MAC}_{\$}(M_2)$
- Surprisingly powerful: breaks universal hash



State-recovery attack

- Consider the first cyclic point
- With high probability, root of the giant tree



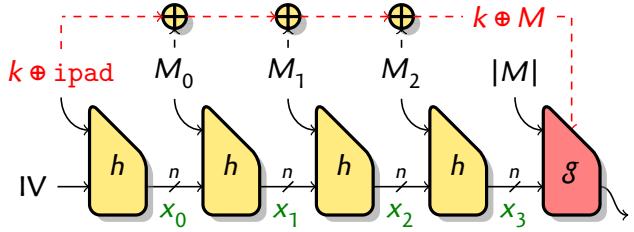
1. Offline: find the root α of the giant tree of ϕ
2. Online: binary search for first cyclic point: smallest z such that

$$\mathtt{MAC}([0]^z \parallel [1] \parallel [0]^{2^{n/2}+\ell}) = \mathtt{MAC}([0]^{z+\ell} \parallel [1] \parallel [0]^{2^{n/2}})$$

- Recover $h^*([0]^z) = \alpha$
- Complexity $\tilde{\mathcal{O}}(2^{n/2})$

Key-recovery attack with a checksum

- Some hash functions have an internal checksum (*e.g.* GOST standards)
- In HMAC, checksum is key-dependent and attacker-controlled



- Related-key attacks on the last block

Surprising result

- The checksum actually makes the hash function weaker!
- HMAC key-recovery attack with complexity $\tilde{\mathcal{O}}(2^{3n/4})$

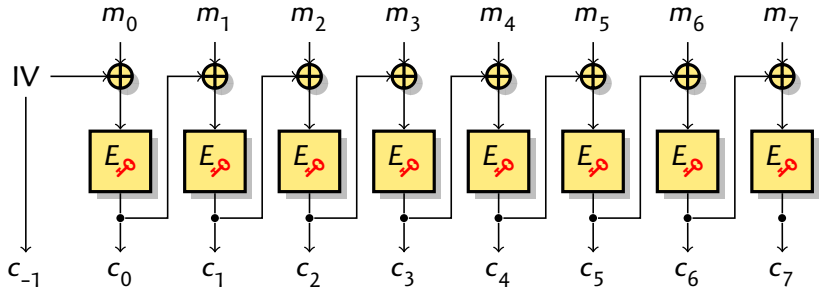
Summary: Cryptanalysis of hash-based MACs

- Security known to be tight (birthday)
- Advanced attacks more powerful than forgeries
  - Using properties of functional graphs, and entropy loss of iteration
- Generic state-recovery attacks
  - Complexity $\tilde{\mathcal{O}}(2^{n/2})$ for Merkle-Damgård (tight)
  - Complexity $\tilde{\mathcal{O}}(2^{4n/5})$ for HAIFA (not tight)
- Generic key-recovery attack against HMAC with a checksum (HMAC-GOST)
  - Complexity $\tilde{\mathcal{O}}(2^{3n/4})$ for Merkle-Damgård (not tight)
  - Complexity $\tilde{\mathcal{O}}(2^{4n/5})$ for HAIFA (not tight)
  - The checksum actually makes the hash function weaker!

Outline

CBC mode

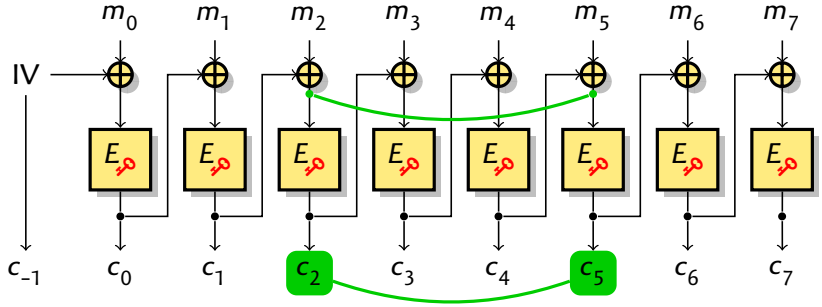
- CBC is a widely used mode, with birthday security [NIST'80]
- Well known collision attack against CBC



- If $c_i = c_j$, then $c_{i-1} \oplus m_i = c_{j-1} \oplus m_j$
  - $m_i \oplus m_j = c_{i-1} \oplus c_{j-1}$
- Ciphertext collision reveals the xor of two plaintext blocks

CBC mode

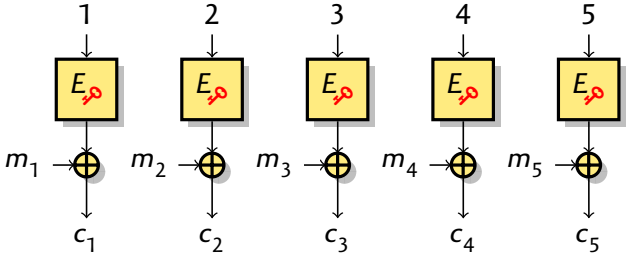
- CBC is a widely used mode, with birthday security [NIST'80]
- Well known collision attack against CBC



- If $c_i = c_j$, then $c_{i-1} \oplus m_i = c_{j-1} \oplus m_j$
  - $m_i \oplus m_j = c_{i-1} \oplus c_{j-1}$
- Ciphertext collision reveals the xor of two plaintext blocks

Counter mode (CTR)

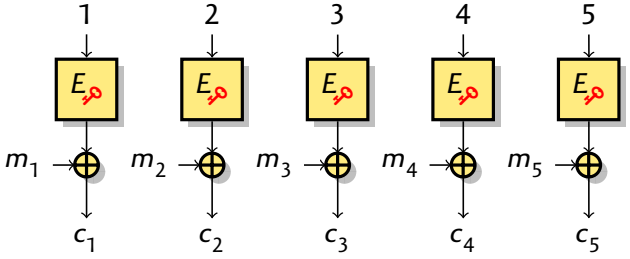
- CTR is a more modern mode with birthday security (used in GCM) [NIST'01]
- Well known distinguisher against CTR



- All block-cipher inputs are distinct
- For all $i \neq j$, $m_i \oplus c_i \neq m_j \oplus c_j$
  - $m_i \oplus m_j \neq c_i \oplus c_j$
  - Hard to extract plaintext information from inequations
- Distinguisher: collision after $2^{n/2}$ blocks with random ciphertext

Counter mode (CTR)

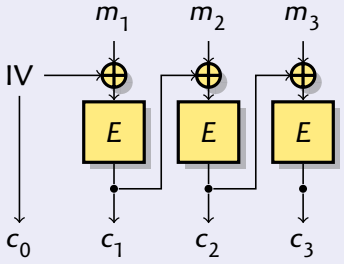
- CTR is a more modern mode with birthday security (used in GCM) [NIST'01]
- Well known distinguisher against CTR



- All block-cipher inputs are distinct
- For all $i \neq j$, $m_i \oplus c_i \neq m_j \oplus c_j$
  - $m_i \oplus m_j \neq c_i \oplus c_j$
  - Hard to extract plaintext information from inequations
- Distinguisher: collision after $2^{n/2}$ blocks with random ciphertext

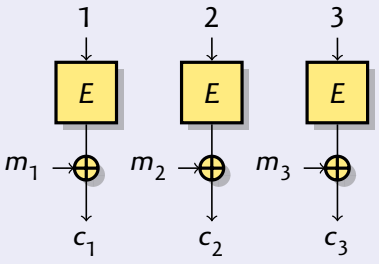
CBC and CTR

CBC mode



- Security proof up to $2^{n/2}$ blocks
- $m_i \oplus m_j = c_{i-1} \oplus c_{j-1}$ if $c_i = c_j$
- Collisions reveals xor of two plaintext blocks

CTR mode



- Security proof up to $2^{n/2}$ blocks
- $m_i \oplus m_j \neq c_i \oplus c_j$ $\forall\, i,j$
- Distinguisher: Key stream doesn't collide

Birthday security in practice

Block size does matter

- State size is an important security parameter
  - Hash functions and stream ciphers use large state size $n \geq 160$
- Modern block ciphers have a 128-bit block size (*e.g.* AES)
  - 2^{64} blocks correspond to 256 EB
- Block ciphers from the 90's have a 64-bit block size (Blowfish, 3DES)
  - 2^{32} blocks correspond to 32 GB

- In 2016, 64-bit block ciphers were still used in practice
  - Around 1−2% of HTTPS connections used 3DES-CBC in 2015−2017
    - Mandatory support in TLS 1.0 and TLS 1.1
    - Supported for compatibility with old client/server
    - Many servers supported AES but preferred 3DES
  - OpenVPN used Blowfish-CBC by default

Proof-of-concept Attack Demo: Sweet32 [Bhargavan & L, CCS'16]

The BEAST setting

- Man-in-the-browser: chosen plaintext
- Authentication cookie: repeated secret
- Modelled as chosen-prefix secret-suffix oracle:
  $M \mapsto \mathcal{E}(M \parallel S)$ with secret S

- Wait for collision between blocks
  from secret cookie and known plaintext

$$\underbrace{\texttt{cookie}}_{\textit{unknown}} \oplus \underbrace{\texttt{header}}_{\textit{known}} = \underbrace{c_{i-1} \oplus c_{j-1}}_{\textit{known}}$$

- Demo with HTTPS traffic with 3DES-CBC
  - Attack successful with ≈ 800 GB of data, collected over 40 hours

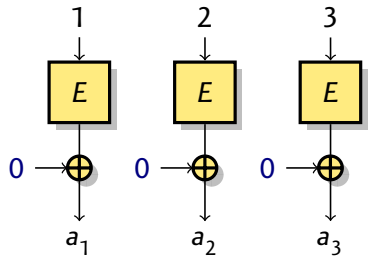
worker.js

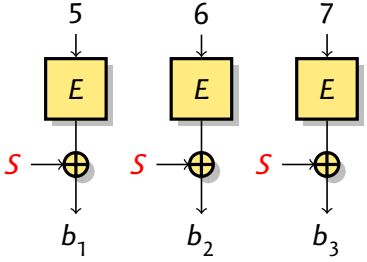
```
var url = "https://target";
var xhr = new XMLHttpRequest;

while(true) {
  xhr.open("HEAD", url, false);
  xhr.withCredentials = true;
  xhr.send();
  xhr.abort();
}
```

Plaintext recovery against CTR

- Collect two kinds of blocks





Chosen plaintext blocks $a_i = E(i)$

Repeated secret $b_j = E(j) \oplus S$

- $\forall i,j,\ a_i \neq S \oplus b_j$
- $\forall i,j,\ S \neq a_i \oplus b_j$

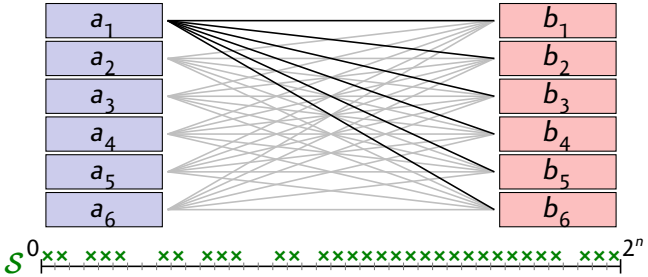
Missing difference problem

Given sets $\mathcal{A}, \mathcal{B} \subset \{0,1\}^n$
Find S such that

$$\forall (a,b) \in \mathcal{A} \times \mathcal{B},\ S \neq a \oplus b$$

Sieving algorithm

[McGrew, FSE'13]

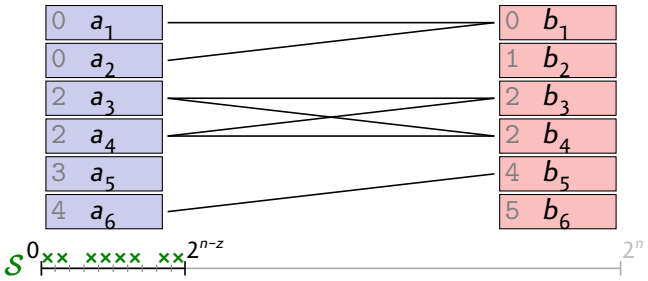


- Compute all $a_i \oplus b_j$, remove from a sieve $\mathcal{S}$

Analysis: Coupon collector problem

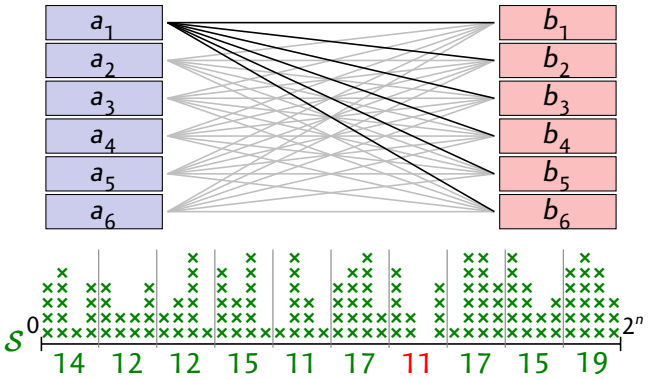
- To exclude 2^n candidates S, we need $n \cdot 2^n$ values $a_i \oplus b_j$
  - Lists $\mathcal{A}$ and $\mathcal{B}$ of size $\sqrt{n} \cdot 2^{n/2}$. Complexity: $\tilde{\mathcal{O}}(2^n)$

Known-prefix sieving [L & Sibleyras, EC'18]



- Assume S starts with z zero bits
  - Smaller sieve
- Sort lists, consider a_i's and b_j's with matching prefix
- Complexity: $\tilde{\mathcal{O}}(2^{n/2})$ when $z \geq n/2$

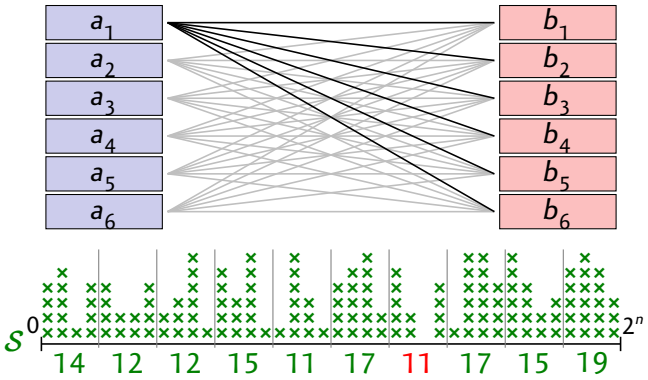
Fast-convolution sieving [L & Sibleyras, EC'18]



- Use $2^{2n/3}$ queries, sieving with $2^{2n/3}$ buckets of $2^{n/3}$ elements
  - With high probability, smallest bucket corresponds to missing difference

$$\#\left\{(a,b) \in \mathcal{A} \times \mathcal{B} \mid a \oplus b \in \mathcal{X}_u\right\} = \sum_{v \in \{0,1\}^{2n/3}} \#\left\{a \in \mathcal{A} \mid a_{[0..2n/3]} = v\right\} \times \#\left\{b \in \mathcal{B} \mid b_{[0..2n/3]} = u \oplus v\right\}$$

Fast-convolution sieving [L & Sibleyras, EC'18]



- Use $2^{2n/3}$ queries, sieving with $2^{2n/3}$ buckets of $2^{n/3}$ elements
  - With high probability, smallest bucket corresponds to missing difference
- Sieving can be computed with Fast Walsh-Hadamard transform!
- Complexity: $\tilde{\mathcal{O}}(2^{2n/3})$ for arbitrary S

Application of missing difference algorithms

Application to CTR mode

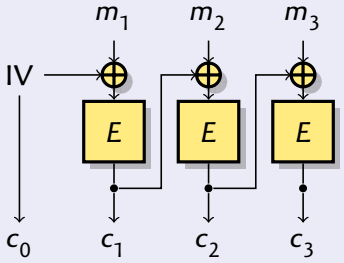
- Message recovery in the BEAST setting
  - Fixed secret encrypted repeatedly
  - Adversary can control the position of a fixed secret
- Target a block with $n/2$ secret bits and $n/2$ known bits
- Message recovery attack with complexity $\tilde{\mathcal{O}}(2^{n/2})$ using known-prefix sieving

Applications to Wegman-Carter MAC

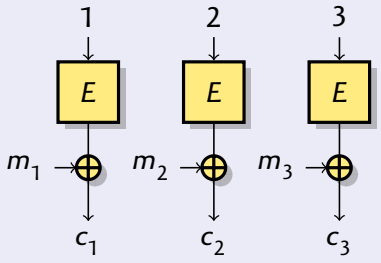
- Recovery of hash key is a missing difference problem
- Complexity $\tilde{\mathcal{O}}(2^{2n/3})$ using fast-convolution sieving
- First partial key-recovery below 2^n

Summary: CBC and CTR

CBC mode



CTR mode



- CTR and CBC both leak plaintext data at the birthday bound
- Birthday attacks are practical against 64-bit block ciphers [Sweet32]
- Alternative: beyond-birthday-bound modes (eg. CENC)

Outline

Expected impact of quantum computers

- Recent progress toward building a large-scale quantum computer
- Some problems can be solved much faster with quantum computers
  - Up to exponential gains
  - But we don't expect to solve all NP problems

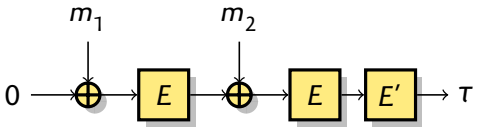
Impact on public-key cryptography

- RSA, DH, ECC broken by Shor's algorithm
  - Breaks factoring and discrete log in polynomial time
  - Large effort to develop quantum-resistant algorithms (*e.g.* NIST)

Impact on symmetric cryptography

- Exhaustive search of κ-bit key in time $2^{\kappa/2}$ with Grover's algorithm
  - Common recommendation: double the key length (AES-256)
  - What is the security of modes of operation in the quantum setting?

CBC-MAC



- One of the earliest MACs, based on CBC encryption mode
- Security proof up to the birthday bound [Bellare, Kilian & Rogaway '94]

Collision attack using two sets of $2^{n/2}$ messages

- $A_x = [0] \| x$
- $\mathtt{MAC}(A_x) = E'(E(x \oplus E([0])))$
- $B_y = [1] \| y$
- $\mathtt{MAC}(B_y) = E'(E(y \oplus E([1])))$
- $\mathtt{MAC}(A_x) = \mathtt{MAC}(B_y)$ iff $x \oplus E([0]) = y \oplus E([1])$
  - Deduce $\delta = E([0]) \oplus E([1]) = x \oplus y$
  - Produce forgeries: $\mathtt{MAC}([0] \| m) = \mathtt{MAC}([1] \| m \oplus \delta)$ for all m

Simon's Algorithm [Simon, SIAM'97]

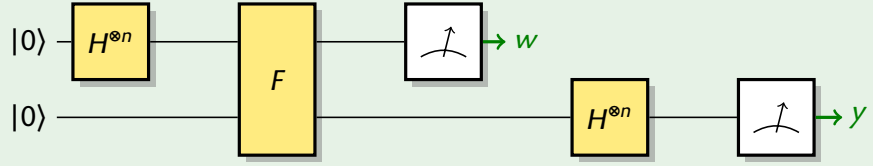
- Quantum algorithm to find collisions with extra structure
- First used in symmetric cryptography by Kuwakado and Morii.

Definition (Simon's problem)

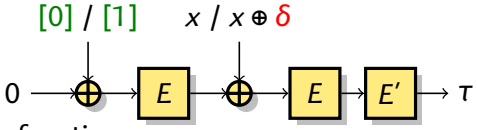
Given $f : \{0,1\}^n \to \{0,1\}^n$ such that there exists $\delta \in \{0,1\}^n$
with $f(x) = f(y) \Leftrightarrow x \oplus y = \delta$ or $x = y$, find δ.

- Classical algorithms require $\mathcal{O}(2^{n/2})$ queries (finding collisions)
- Simon's algorithm require $\mathcal{O}(n)$ quantum queries

One step of Simon's algorithm returns $y \perp \delta$



Quantum attack against CBC-MAC [Kaplan, L, Leverrier, Naya-Plasencia, C'16]



1. Consider the following function:

$$f : \{0,1\} \times \{0,1\}^n \to \{0,1\}^n$$
$$b, x \mapsto \texttt{MAC}([b] \,\|\, x) = E'\left(E\left(x \oplus E([b])\right)\right)$$

$$f(b,x) = f(b',x') \iff \begin{cases} b = b' \text{ and } x = x' \\ b \neq b' \text{ and } x \oplus x' = E([0]) \oplus E([1]) \end{cases} \quad \text{or}$$

   - f has period $1 \,\|\, \delta$, with $\delta = E([0]) \oplus E([1])$
2. Use Simon's algorithm to recover $1 \,\|\, \delta$
3. Produce forgeries: $\texttt{MAC}([0] \,\|\, m) = \texttt{MAC}([1] \,\|\, m \oplus \delta)$

Summary: Quantum security of modes of operation

Applications of Simon's algorithm

- Generalization breaks most common MAC and AEAD modes
  - `CBC-MAC`, `PMAC`, `GMAC`, `GCM`, `OCB`, ... [KLLNP, Crypto'16]
  - `θCB`, `LightMAC`, `LightMAC+`... [BLNS, AC'21]
- Corresponds to classical attacks with $2^{n/2}$ queries
  - Query f with $2^{n/2}$ values, look for collisions
- Strong assumption: superposition queries

- Surprising, because common encryption modes are quantum-secure (CBC, CTR) [Unruh, Targhi, Tabia & Anand, PQC'16]
- Some MAC are also quantum-secure: Cascade/HMAC is [Song & Yun, Crypto'17]

Conclusion: Cryptanalysis beyond primitives

Fun research area

- Interesting algorithmic problems for generic attacks
- Concrete attacks with practical impact

- Modes and protocols usually studied with proofs but cryptanalysis is useful
  - Mistakes in proofs
  - Gap between proofs and attacks
  - Different security degradation after the birthday bound
  - Usage when the proof does not apply

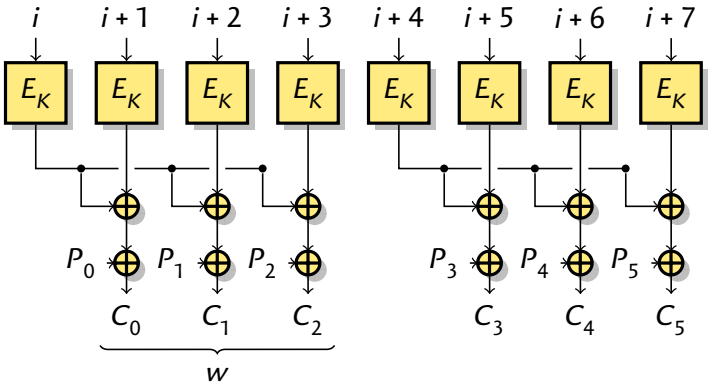
Take away

- Don't assume security above the birthday bound without a proof
- Many generic attacks exploit collisions in smart ways

CENC
Chosen-prefix collisions

CENC

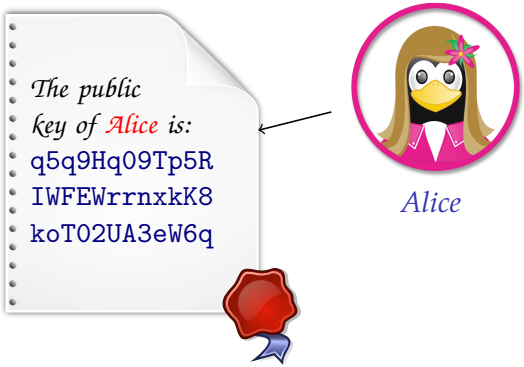
[Iwata, FSE '06]



- Variant of the counter mode
- Designed by Iwata [FSE '06]
- Security $2^n/w$ [Iwata, Mennink & Vizár '16]

Attacking key certification

[Stevens, Lenstra & de Weger, EC'07]



PKI Infrastructure

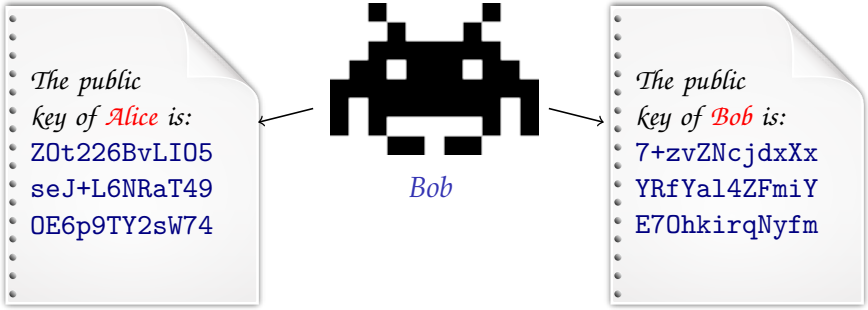
- Alice generates key
- Asks CA to sign
- Certificate proves ID

Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||\mathcal{P}_A) = H(\texttt{Bob}||\mathcal{P}_B)$
2. Bob asks CA to certify his key $\mathcal{P}_B$
3. Bob copies the signature to $\mathcal{P}_A$, impersonates Alice

Attacking key certification

[Stevens, Lenstra & de Weger, EC'07]



PKI Infrastructure

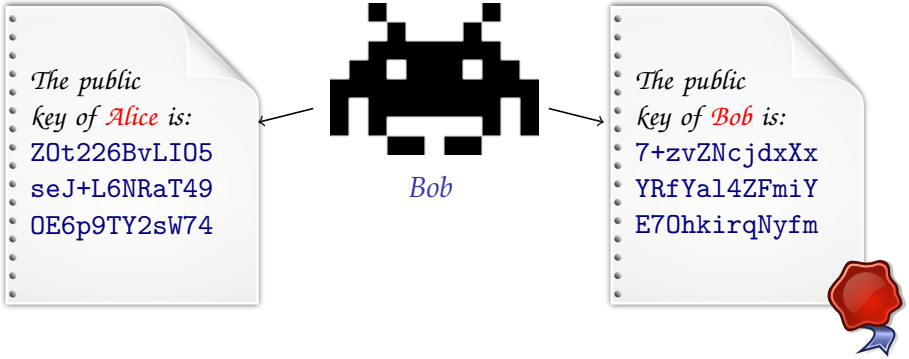
- Alice generates key
- Asks CA to sign
- Certificate proves ID

Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||\mathcal{P}_A) = H(\texttt{Bob}||\mathcal{P}_B)$
2. Bob asks CA to certify his key $\mathcal{P}_B$
3. Bob copies the signature to $\mathcal{P}_A$, impersonates Alice

Attacking key certification

[Stevens, Lenstra & de Weger, EC'07]



PKI Infrastructure

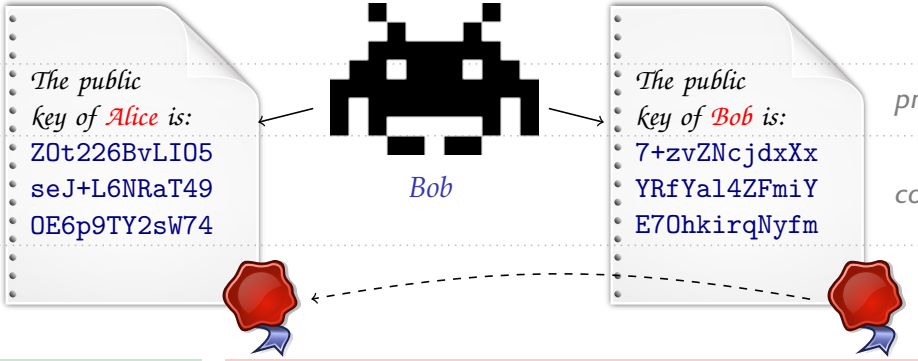
- Alice generates key
- Asks CA to sign
- Certificate proves ID

Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||\mathcal{P}_A) = H(\texttt{Bob}||\mathcal{P}_B)$
2. Bob asks CA to certify his key $\mathcal{P}_B$
3. Bob copies the signature to $\mathcal{P}_A$, impersonates Alice

Attacking key certification

[Stevens, Lenstra & de Weger, EC'07]



PKI Infrastructure

- Alice generates key
- Asks CA to sign
- Certificate proves ID

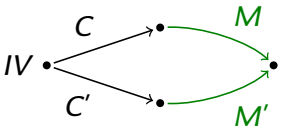
Impersonation attack

1. Bob creates keys s.t. $H(\texttt{Alice}||\mathcal{P}_A) = H(\texttt{Bob}||\mathcal{P}_B)$
2. Bob asks CA to certify his key $\mathcal{P}_B$
3. Bob copies the signature to $\mathcal{P}_A$, impersonates Alice

Chosen-prefix collisions

Chosen-prefix collision [Stevens, Lenstra & de Weger, EC'07]

Given challenges C, C',
Find M, M' s.t. $H(C \parallel M) = H(C' \parallel M')$

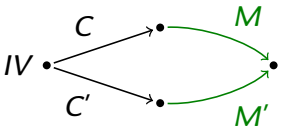


- Application to certificate
  - C = `Alice`, C' = `Bob`
  - Collision M, M' hidden in the public key

Chosen-prefix collisions

Chosen-prefix collision [Stevens, Lenstra & de Weger, EC'07]

Given challenges C, C',
Find M, M' s.t. $H(C \| M) = H(C' \| M')$



Generic

- Complexity $2^{n/2}$
- Compute $H(C \| r)$ and $H(C' \| r)$ for $2^{n/2}$ choices r
- Memoryless with helper function

$$\phi(x) = \begin{cases} H(C \| x) & \text{if } x \text{ is odd} \\ H(C' \| x) & \text{if } x \text{ is even} \end{cases}$$

Dedicated

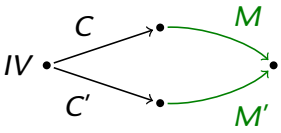
- More difficult than plain collision
- Still possible using similar techniques

Function	MD5	SHA-1
Hash length	128	160
Collision	2^{16}	$2^{61.6}$
CP collision	$2^{39.1}$	$2^{63.5}$

Chosen-prefix collisions

Chosen-prefix collision [Stevens, Lenstra & de Weger, EC'07]

Given challenges C, C',
Find M, M' s.t. $H(C \| M) = H(C' \| M')$



Generic

- Complexity $2^{n/2}$
- Compute $H(C \| r)$ and $H(C' \| r)$ for $2^{n/2}$ choices r
- Memoryless with helper function

$$\phi(x) = \begin{cases} H(C \| x) & \text{if } x \text{ is odd} \\ H(C' \| x) & \text{if } x \text{ is even} \end{cases}$$

Dedicated

- More difficult than plain collision
- Still possible using similar techniques

Function	MD5	SHA-1
Hash length	128	160
Collision	2^{16}	$2^{61.6}$
CP collision	$2^{39.1}$	$2^{63.5}$