Selected Areas in Cryptology
Cryptanalysis
Week 6

Marc Stevens

stevens@cwi.nl

https://homepages.cwi.nl/~stevens/mastermath/2021/

Cryptographic Hash Functions

Theoretical Cryptology:

Hash function family with the same range $\mathcal{H}$ (e.g. $\{0,1\}^{256}$)

$$\mathcal{F} = \{f : \{0,1\}^* \to \mathcal{H}\}$$

Security games for any PPT adversary A

- Pre: pre-image resistance:
  $f \leftarrow \mathcal{F}, h \leftarrow \mathcal{H}$, wins if $M \leftarrow A(f,h)$ and $f(M) = h$
- ePre: everywhere Pre: $h \leftarrow A$, $f \leftarrow \mathcal{F}$ instead
- aPre: always Pre: $f \leftarrow A$, $h \leftarrow \mathcal{H}$ instead
- Sec: second preimage resistance:
  $f \leftarrow \mathcal{F}, M \leftarrow \{0,1\}^{\leq n}$, win if $M' \leftarrow A(f,M), f(M) = f(M')$ and $M \neq M'$
- eSec: everywhere Sec: $M \leftarrow A, f \leftarrow \mathcal{F}$ instead
- aSec: always Sec: $f \leftarrow A, M \leftarrow \{0,1\}^{\leq n}$ instead
- Coll: collision resistance
  $f \leftarrow \mathcal{F}$ win if $M, M' \leftarrow A(f)$ and $f(M) = f(M')$ and $M \neq M'$

Cryptographic hash function

Hash function standards $H: \{0,1\}^* \to \{0,1\}^n$:

- MD5:
  - 128-bits hash function published in 1992
  - Widely used till ~~2010
  - Broken in 2004: first collision found [WY05],
    real world attacks in 2009: rogue certificate authority [SSA+09]
    & 2012: windows update forged certificate [FS15]
- SHA-1:
  - 160-bit hash function published in 1995
  - Widely used even today (TLS1.2, Git, …)
  - 'Broken' in 2005: first theoretical collision attack [WYY05]
    practical attack in 2017: first collision [SBKAM17]
- SHA-2 family:
  - 224/256/384/512-bit hash functions published in 2001
- SHA-3 family:
  - 224/256/384/512-bit hash functions published in 2015

Cryptographic hash functions

Fixed n-bit hash functions: $f: \{0,1\}^* \rightarrow \{0,1\}^n$

- Pre, ePre, Sec, eSec, Coll security notions ill-defined

- aPre: always pre-image resistance:
  - Given random $h \leftarrow \{0,1\}^n$ find M s/t $f(M) = h$
- aSec: always second pre-image resistance:
  - Given random $M \leftarrow \{0,1\}^{\leq n}$ find $M' \neq M$ s/t $f(M) = f(M')$

- Secure if there is no attack faster than a generic attack
  - aPre/aSec brute-force search:

    While true

    $M' \leftarrow \{0,1\}^{\leq n}$, $h' = f(M')$

    If $h' = h$ then return M'

    - Cost $O(2^n)$

      Geometric distribution with $p = 2^{-n} \Rightarrow$ mean success cost $1/p = 2^n$

Collision conundrum

How to define collision resistance for fixed hash functions?

Mathematical existential security definitions?:

> "There should exist no attack that is feasible/faster than generic attack/PPT that finds a collision with non-negligible probability"

Conundrum:

> Pigeon-hole principle ⇒ collisions exist
>
> Any collision $f(M) = f(M')$ with $M \neq M'$ leads to a trivial attack:
>
> > Algorithm $A_{M,M'}$: simply outputs the pair M, M'
> >
> > Such algorithms exist and break security definitions
> >
> > However, we can't actually write down such algorithms unless we first compute a collision… (i.e., its non-uniform)

Foundations of Hashing Dilemma:

> No formal definition of collision resistance exists
>
> Informal definition relies on human ignorance:
>
> > "There exists no *known* attack that …"

Generic collision attack

- Generic collision attack

  For $i = 1, \ldots$

  Sample $M_i \leftarrow \{0,1\}^{\leq n}$, $h_i = f(M_i)$

  If $\exists j < i: h_j = h_i$ then return (M_j, M_i)

- Cost analysis:
  - let X be the number of samples needed before a collision is found
  - $E[X] = \sum_{k=1}^{\infty} k \cdot \Pr[X = k] = \sum_{k=1}^{\infty} k \cdot (\Pr[X > k-1] - \Pr[X > k])$
    $= \sum_{k=0}^{\infty} (k+1) \cdot \Pr[X > k] - \sum_{k=1}^{\infty} k \cdot \Pr[X > k]$
    $= \sum_{k=0}^{\infty} \Pr[X > k]$
  - Week 2: $\Pr[X > k] \approx e^{-k^2 2^{-n-1}}$ ("no collision after k samples")
  - Estimate:
    $E[X] \approx \sum_{k=0}^{\infty} e^{-k^2 2^{-n-1}} \approx \int_0^{\infty} e^{-k^2 2^{-n-1}} = \sqrt{\pi/2} \cdot 2^{n/2}$ time cost
  - Memory cost: $O\left(2^{n/2}\right)$

Generic collision attack

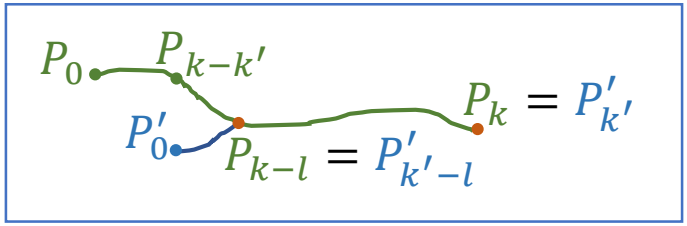
- Memory cost improvements
  - Idea: compute trails and only store begin/end-points like Hellman's time-memory trade-off attack

  - Define search space: $\mathcal{H} := \{0,1\}^n$
  - Choose injective embedding $\phi: \mathcal{H} \to \{0,1\}^*$
    - Let $g := f \circ \phi: \mathcal{H} \to \mathcal{H}$
    - Hence a collision of $H \neq H'$ of g (i.e., $g(H) = g'(H)$), is a collision $\phi(H) \neq \phi(H')$ of f
  - Choose set of 'distinguished points' $S \subset \mathcal{H}$:
    - Easily distinguishable: e.g. last l-bits are zero

  - Compute trails:
    - Choose random starting point P_0
    - Iterate $P_i = g(P_{i-1})$ until a distinguishable point $P_i \in S$ is encountered
    - Then only store begin/end-point & length (P_0, P_i, i)

Generic collision attack

- Compute trails:
  - Choose random starting point P_0
  - Iterate g: $P_i = g(P_{i-1})$
    until a distinguishable point $P_i \in S$ is encountered
  - Then only store begin/end-point & length (P_0, P_i, i)
- What happens when a collision occurs:
  - $P_i \neq P_j$ and $g(P_i) = g(P_j)$
  - Since g is deterministic, the two trails merge:
    - $g^k(P_i) = g^k(P_j)$
    - End at the same distinguished point: $g^k(P_i) = g^k(P_j) \in S$
- Resolving a collision:
  - Consider two trails (P_0, P_k, k), $(P'_0, P'_{k'}, k')$
    with $P_k = P'_{k'} \in S$ (wlog $k \geq k'$)
  - Assume collision occurs l iterations before end
  - First synchronize: iterate longest trail $k - k'$ iterations
  - Exceptional case: $P_{k-k'} = P'_0 \Rightarrow$ 'robin-hood' failure
  - Iterate for $i = k' - 1, \ldots, 0$:
    - If $P_{k-i} = P'_{k'-i}$ then return $\phi(P_{k-i-1}), \phi(P'_{k'-i-1})$



Generic collision attack

Memory cost

- Expected total evaluations before collision occurs:
  $E[X] = \sqrt{\pi/2} \cdot 2^{n/2}$
- Expected trail length $t := |\mathcal{H}|/|S|$ (geometric distribution with $p = |S|/|\mathcal{H}|$)
- We expect $\approx \sqrt{\pi/2} \cdot 2^{n/2}/t$ trails to store
  - If S consists of points with last l-bits zero
    then $t = 2^n/2^{n-l} = 2^l$ and $O\left(2^{n/2-l}\right)$ memory cost
- Additional costs:
  - Once a collision occurs, need to finish the trail:
    t evaluations (expected trail length is memoryless)
  - To compute the actual collision point:
    $2.5\ t$ evaluations (analysis see link in lecture notes)
  - Total $O(3.5t) = O(3.5 \cdot 2^l)$ time cost
- Suggested choice $l = n/2 - 20$
  - Memory cost ≈1M trails
  - Expected additional time cost: $O\left(2^{n/2}/2^{20}\right) \ll O\left(2^{n/2}\right)$

Generic collision attack

- See lecture notes for full collision attack algorithm
- Unlikely problematic case:
  - A trail enters a cycle without ever reaching a distinguished point
  - ⇒ collision attack would loop forever
- Solution:
  - Discard trail if $20\,t$ iterations is reached
  - Discard case 1: no cycle reached
    - Probability $\left(1-\frac{1}{t}\right)^{20\,t} \approx \left(e^{-\frac{1}{t}}\right)^{20t} = e^{-20} \approx 2^{-29}$
  - Discard case 2: cycle reached: internal collision
    - Probability: $1 - e^{-(20t)^2\,2^{-n-1}}$
    - Need $(20t)^2 \ll 2^n$ for this probability to be small enough
  - Both negligible losses

Wrap-up

- Cryptographic hash functions
  - Theoretical cryptography: hash function families
  - Practice: fixed hash function standards

- Foundations of Hashing Dilemma:
  - No security definition possible for collision resistance for fixed hash functions
  - Informal definition: “no known attack”

- Generic collision attack:
  - Birthday paradox
  - Use trails and distinguished points to reduce memory cost