Hashing and MACs

Aggelos Kiayias

What is a hash function?

- A way to produce a "fingerprint" of a file

$$\mathcal{H} : \{0,1\}^* \rightarrow \{0,1\}^v$$

what are the required properties (traditionally):

1. Efficiency.
2. A good spread for various input distributions.

What are Security/Cryptographic considerations?

Applications

1. A short representation of a file (e.g., in fingerprints, digital signatures etc.)
2. Can be used for detection of channel errors or malicious errors.
3. Equality testing with privacy (passwords).
4. Can be used to commit to data.

Attacks with Collisions

- When hashing is used to "commit" to a certain value.

Collision attack Find $x, y : \mathcal{H}(x) = \mathcal{H}(y)$

Second Pre-image attack

Given x Find $y : \mathcal{H}(x) = \mathcal{H}(y)$

Attacks against Secrecy

- When hashing is used to hide data.

General Problem statement:

Given $\mathcal{H}(m) = h$

the goal is to recover m

Also called "(first) pre-image attack"

Important, e.g., for password hashing, and other applications

Hashing for error detection

- Verification of downloaded software.
- Verification of transmitted packets.

Hashing for Committing

- Sealed bid Auctions : submit (private) bids independently, then highest bid wins.
  - Implementation : bidders publish Hash(bid)
  - after all bidders publish, they announce their bids.
  - Is this private (does it implement sealed bids)?

Second pre-image attack resistance ensures **binding** ?
First pre-image attack resistance ensures **hiding** ?

The birthday paradox

- How many people should be in a room so that the probability that two of them share a birthday becomes larger than 50%?

The paradox explained

$$\Pr[\neg Col] =$$

$$\frac{n}{n}\frac{n-1}{n}\frac{n-2}{n}\cdots\frac{n-k+1}{n} = \prod_{\ell=1}^{k}(1-\frac{\ell}{n})$$

$$\leq \exp(-\frac{1}{n}\sum_{\ell=1}^{k}\ell) = \exp(-k(k+1)/2n)$$

$$\Pr[Col] = \frac{1}{2} \Rightarrow k \approx 1.177\sqrt{n}$$

Collision finding using B.P.

- Random sampling $x, \mathcal{H}(x)$
- Store *n* such pairs in a table.
- Sort the table according to the second coordinate.
- Linear pass to find if any entries have equal second coordinate.
  - what is the complexity of this algorithm?

Complexity

- Storage : k
- Time : ~ $2k + k \log k$
- How to choose k for 50% probability of success: $\lceil 1.177\sqrt{n} \rceil$
- This is why the length of hash function output is typically selected to be double the length of the key of an encryption algorithm.

Importance of B.P

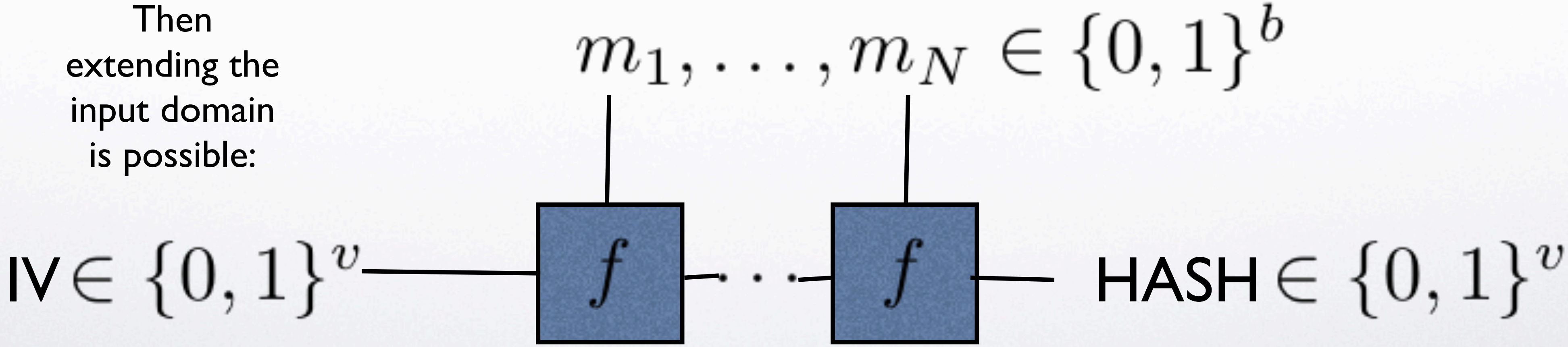
- Consider a developer that distributes software and tells you that the first 20 Hex characters of the hash is: **4E9A51FD8A35EC69AFAC**
- You find this file in a torrent and the hash matches. Are you convinced this is correct file?

Merkle Damgård Design

- Suppose that you have a good “compression” function.

$$f : \{0,1\}^v \times \{0,1\}^b \rightarrow \{0,1\}^v$$

Then extending the input domain is possible:



Observation

If the previous construction is used then given $\mathcal{H}(m)$

One can easily find the hash of $m||m'$ How?

Hash functions constructed as above are called
Iterated Hash Functions

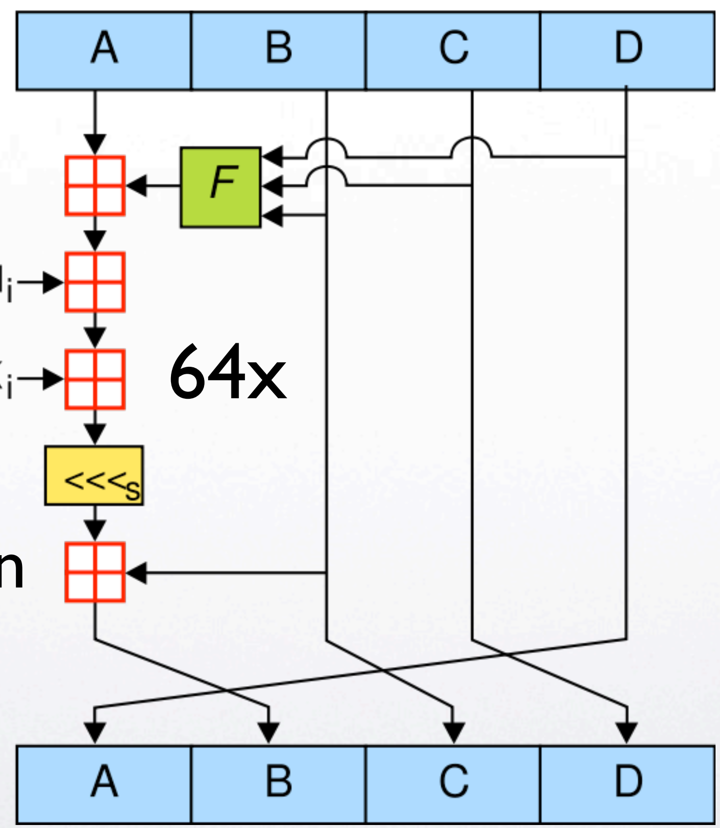
MD5

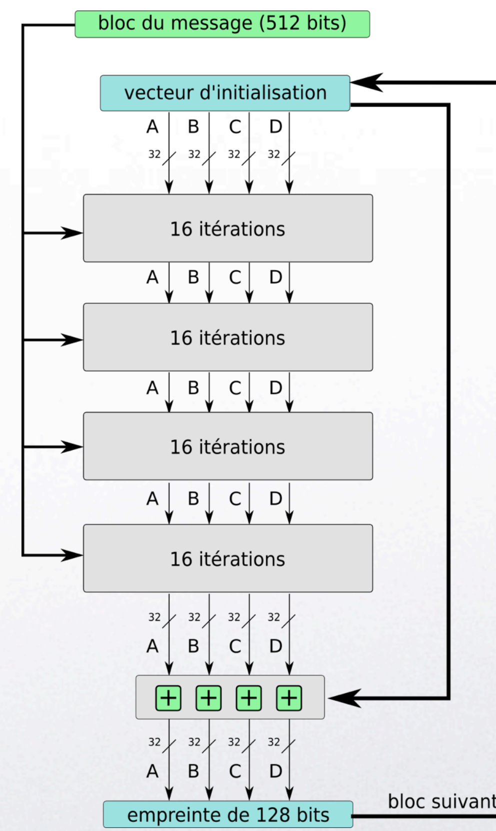
128 bit output

⊞ addition mod 2^{32}

⋘ rotation

F : simple non-linear function





MD5 attacks

- MD4: 128 bits, 1990. Broken
- MD5: 128 bits. 1992. Wide Usage. Collision attacks in 2004.
- Current best c.a. : Xie-Feng (2009) in 2^{20}
- preimage attacks : still hard ~ $2^{123.4}$ (Sasaki-Aoki)

The SHA Family

- SHA-0 was made a standard by NIST in 1993. It was designed by NSA. Also based on Merkle Damgard design. 160 bits.
- In 1995 a technical revision was added that added an additional rotation operation. This was SHA-1.
- In 1998 collisions against SHA-0 were demonstrated in 2^{61} steps (Chabaud - Joux Crypto 1998)

The SHA Family, II

- SHA-1 was thought to be secure but evidence to the contrary was emerging (near-collisions, collisions on "reduced round" versions etc.)
- Finally: Collisions were found in 2^{69} steps Wang, Yin, Yu, Crypto 2005.
- SHA-1 is considered broken and its usage will be discontinued.

The SHA Family, III

- Isn't 2^{69} still too high?
- [Wang-Yao-Yao'05] announced new collision attacks of complexity: 2^{63}
- [De Caniere and Rechberger'06] - towards more structured collisions for SHA-1
- In 2009 claims for 2^52 were made (?).

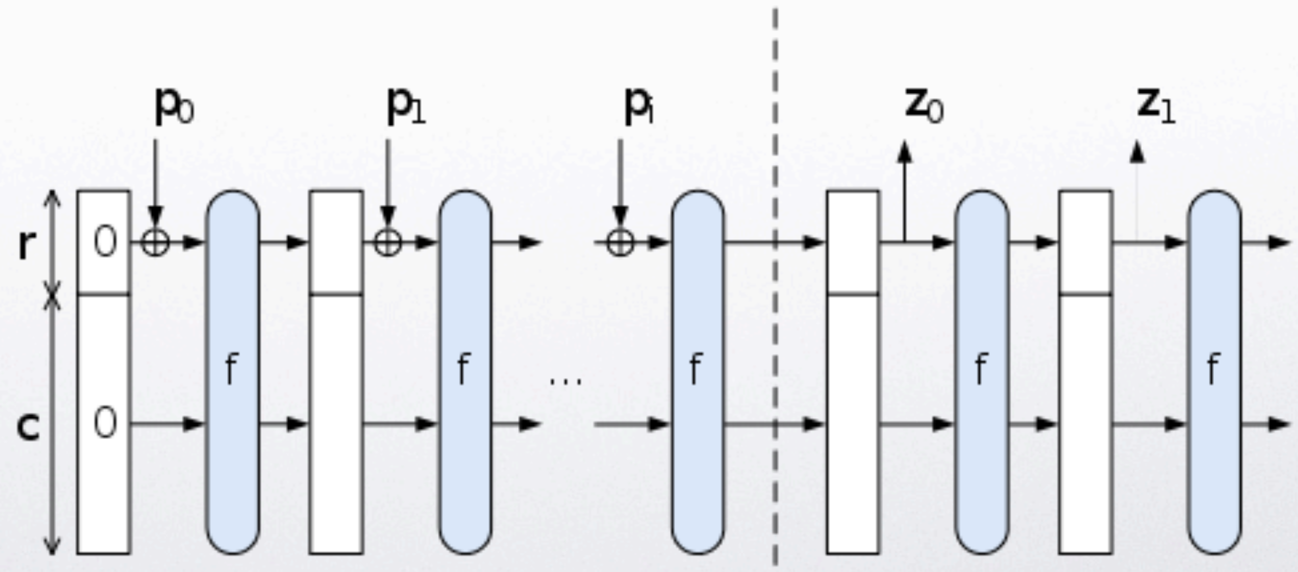
Still...

- The above results do not necessarily mean that current products that use SHA-1 are insecure [remember our definition of security]. Moreover
  - There are ways to employ hash functions so that collision finding is made harder(e.g., "salting" techniques).
  - Modify hash functions in a modular way.

SHA3

224, 256, 384 and 512-bit output

- started 2007 - a competition by NIST
- In 2012 Keccak was selected as the SHA-3 winner
  - Blake
  - Grostl
  - JH
  - Keccak
  - Skein



MACs

message authentication codes

- Keyed hash functions: $\mathcal{H}_k : \{0,1\}^* \rightarrow \{0,1\}^n$

Required Property
Computational Resistance against MAC forgery:

Given any sequence of pairs

$$\langle m_1, \mathcal{H}_k(m_1)\rangle, \ldots, \langle m_v, \mathcal{H}_k(m_v)\rangle$$

It is hard to produce an additional pair: $\langle m, \mathcal{H}_k(m)\rangle$

Usage of MACs

- Data Origin Authentication.
  - Sender and Receiver share the key *k*.
  - All messages have a MAC appended by Sender.
  - Receiver verifies the MACs (by recomputing them).

Constructing MACs

- Generic construction based on a hash function: $\mathcal{H}_k(m) = \mathcal{H}(k||m)$
- Not good: if an iterated hash function is employed this ***will allow an attack***.

a possible implementation:

Constructing MACs

- Generic construction based on a hash function: $\mathcal{H}_k(m) = \mathcal{H}(k||m)$
- Not good: if an iterated hash function is employed this ***will allow an attack***.

a possible implementation:

$$\mathcal{H}_k(m) = \mathcal{H}(k||padding||m||k)$$

Assigned Reading

- I. Mironov, Hash functions: Theory attacks and applications.

http://research.microsoft.com/en-us/people/mironov/hash_survey.pdf