Design of Hash functions and Some Attacks.

Mridul Nandi

Indian Statistical Institute, Kolkata.

What is a hash function?

(Cryptographic) Hash Function

- It uses some atmoic operations, e.g. bit-wise rotation, xor, shift, modular addition, multiplication, S-box etc.

- $H : \mathcal{M} \rightarrow \{0,1\}^n$. A public function (anybody can compute)

- $\mathcal{M}$ is called message space, n is called the hash-size.
  Message space: $\{0,1\}^*$, $\{0,1\}^{2^{64}}$, $(\{0,1\}^w)^*$ where w is word-size.
  Hash size: n = 128, 160, 224, 256, 384, 512 etc.

What we demand from a good hash function?

Hash Function

- All cryptographic objects or building blocks have two features (in general)

(1) **correctness**: what we want to achieve minimally (good or bad)?

(2) **Security**: What we achieve extra features from a good building blocks to protect us from bad people?

Hash Function

- All cryptographic objects or building blocks have two features (in general)

(1) **correctness**: what we want to achieve minimally (good or bad)?

- E.g. encryption should have inverse and both encryption and decryption (inverse) should be efficiently computable. Similarly, hash function should take an input of a specified message space and gives an output of fixed specified length.

(2) **Security**: What we achieve extra features from a good building blocks to protect us from bad people?

Hash Function

- All cryptographic objects or building blocks have two features (in general)

(1) correctness: what we want to achieve minimally (good or bad)?

- E.g. encryption should have inverse and both encryption and decryption (inverse) should be efficiently computable. Similarly, hash function should take an input of a specified message space and gives an output of fixed specified length.

(2) Security: What we achieve extra features from a good building blocks to protect us from bad people?

- In case of Encryption, given plaintexts and the corresponding ciphertexts, the key should not be revealed.
- Similarly, we have (many) security goals from a good cryptographic hash function. We will make a list later.

Examples of Hash Functions?

Examples of Hash Function

- Traditional hash: MD4, MD5 (Ron Rivest)
  - Widely used... Some weakness observed.
  - finally MD6 (again by Rivest and his team)
- SHA-0, SHA-1, SHA-2 (**SHA-224, SHA-256, SHA-384, SHA-512**) designed by the National Security Agency (NSA) and published by the NIST.
  - Again some weakness in SHA-0 and even SHA-1 are observed.
- SHA-3 competition called by NIST.
  - History of the SHA3-competition.
  - Current status: five finalists have been selected.
  - They are: Blake, Grostl , JH, Keccak, and Skein
  - In 2012, the winner will be announced.
  - For more information: SHA3-zoo, NIST web-page.

Some Applications of Hash Function

Application: Hash Function

- 1. Digital Signature : Let sig_{SK} be a signature algorithm over $\{0,1\}^n$. We define SIG_{SK} over $\mathcal{M}$ as

-- $\boxed{SIG_{SK}(M) = sig_{SK}(H(M)).}$

1. Make message compatible with signature algorithm.

2. Random looking hash output

3. Much faster algorithm

Hash →

Digital Signature Algorithm sig_{SK}

→ Value of Digital Signature

Application: Hash Function

- 1. Digital Signature : Let sig_{SK} be a signature algorithm over $\{0,1\}^n$. We define SIG_{SK} over $\mathcal{M}$ as

-- $\boxed{SIG_{SK}(M) = sig_{SK}(H(M)).}$

1. Make message compatible with signature algorithm.

2. Random looking hash output

3. Much faster algorithm

Hash Function should be

Collision resistance:

Hard to find $M \neq M'$ such that $H(M) = H(M')$.

Hash →

Digital Signature Algorithm sig_{SK}

→ Value of Digital Signature

If H is not CR then what is the problem??

Application: Hash Function

- 2. Bit-commitment: To commit a message M, make c = H(M) public.
  - **Hiding property**: **preimgae resistance:** (given c, hard to find M).
  - **Binding property:** hard to change the commitment i.e. to find a message M' such that H(M') = c.

    - **collision resistance,** **2^{nd} preimage resistant** (given M hard to find $M \neq M'$ such that H(M) = H(M')).

- This works for long message. How one can commit for a single bit b?

Application: Hash Function

- 2. Bit-commitment: To commit a message M, make c = H(M) public.
  - **Hiding property**: **preimgae resistance:** (given c, hard to find M).
  - **Binding property:** hard to change the commitment i.e. to find a message M' such that H(M') = c.

    - **collision resistance**, **2nd preimage resistant** (given M hard to find $M \neq M'$ such that H(M) = H(M')).

- This works for long message. How one can commit for a single bit b?
  - Choose a long random string r and commit b||r instead of b.
  - One can append the random string for M also.

Application: Hash Function

- Message authentication: E.g. HMAC (Bellare et al.)
  - Keyed hash function: (1) classical: H(K|| M), (2) sandwich: H(K||M||K) etc.

- Public Key Encryption (Kurosawa-Desmedt, Cramer-Shoup, DHIES etc.).

- Identity based Public Key Encryption (Boneh et al.) (public-key encryption with identity (e.g. gmail-id) as a public key).

Application: Hash Function

- Key extraction: Given a long key-stream (e.g. biometric data) with less entropy how one can compute a smaller key-size with full entropy.
  - A possible solution is to apply a good hash function to the long key-stream.

We have already heard some security requirements, now

Can we make a list?

Security Requirements: Hash Function

(The MOST POPULAR)

- (1) collision resistance.
- (2) Preimage resistant.

The others

- (3) 2nd preimage resistant
- (4) multicollision resistant.
- (5) Target collision resistant
  or UOWHF.
- (6) resistant against length-extension attack
- (7) Herding attack.
- (8) indistinguishability in outputs. **PRF, PRO..**
- ETC…

Security Requirements: Hash Function

(The MOST POPULAR)

- (1) collision resistance.
- (2) Preimage resistant.

The others

- (3) 2^{nd} preimage resistant
- (4) multicollision resistant.
- (5) Target collision resistant or UOWHF.
- (6) resistant against length-extension attack
- (7) Herding attack.
- (8) indistinguishability in outputs. **PRF, PRO..**
- ETC…

Security Requirements: Hash Function

(The MOST POPULAR)

- (1) collision resistance.
- (2) Preimage resistant.

The others

- (3) 2nd preimage resistant
- (4) multicollision resistant.
- (5) Target collision resistant or UOWHF.
- (6) resistant against length-extension attack
- (7) Herding attack.
- (8) indistinguishability in outputs. PRF, PRO..
- ETC…

SWISS ARMY KNIFE

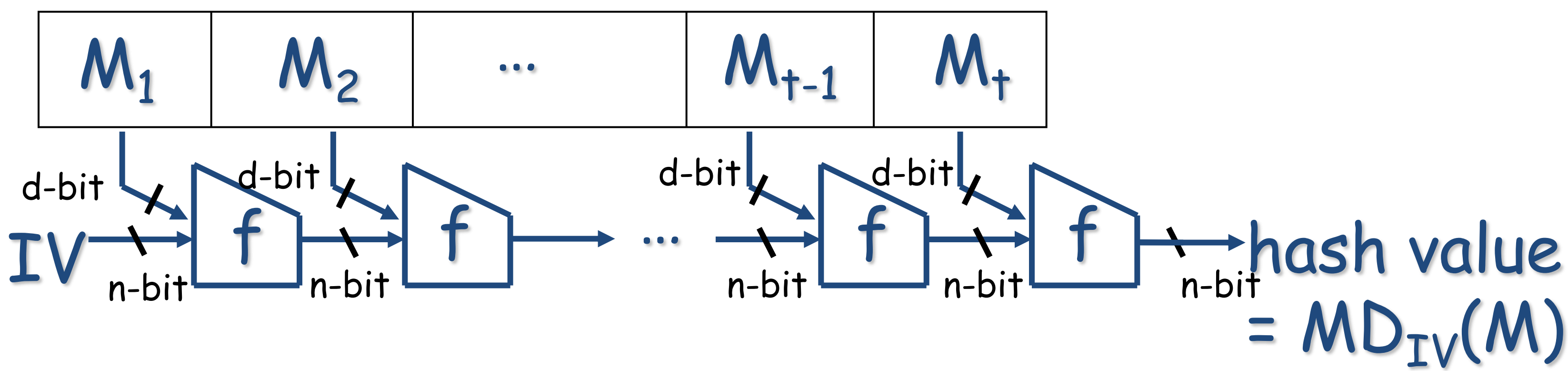
We have a long list of security requirements, so let's begin with

collision resistant

Merkle-Damgård (Crypto-89)

- Let $f: \{0,1\}^{n+d} \rightarrow \{0,1\}^n$ be a compression function.

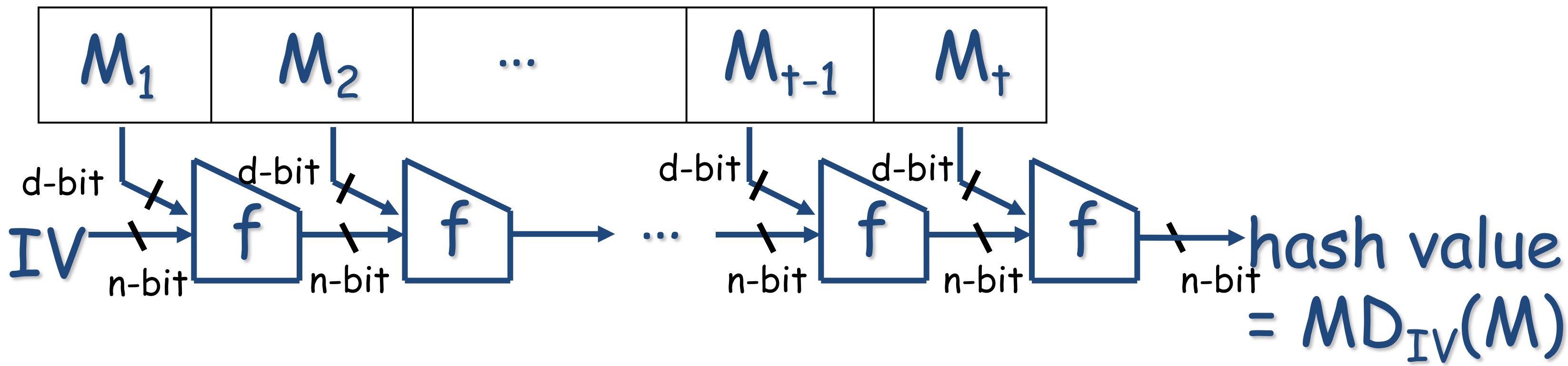
  - $\mathrm{Pad}(M) = M \,||\, 10\ldots00 \,||\, \mathrm{binary}(|M|)_{64} = M_1 \,||\, \ldots \,||\, M_t$



Merkle-Damgård (Crypto-89)

- Let $f: \{0,1\}^{n+d} \rightarrow \{0,1\}^n$ be a compression function. Given a message M find smallest $r \geq 0$ such that $|M| + 65 + r$ is multiple of d. Append $10^r \,||\, \text{binary}(|M|)_{64}$.

  - $\text{Pad}(M) = M \,||\, 10...00 \,||\, \text{binary}(|M|)_{64} = M_1 \,||\, ... \,||\, M_t$

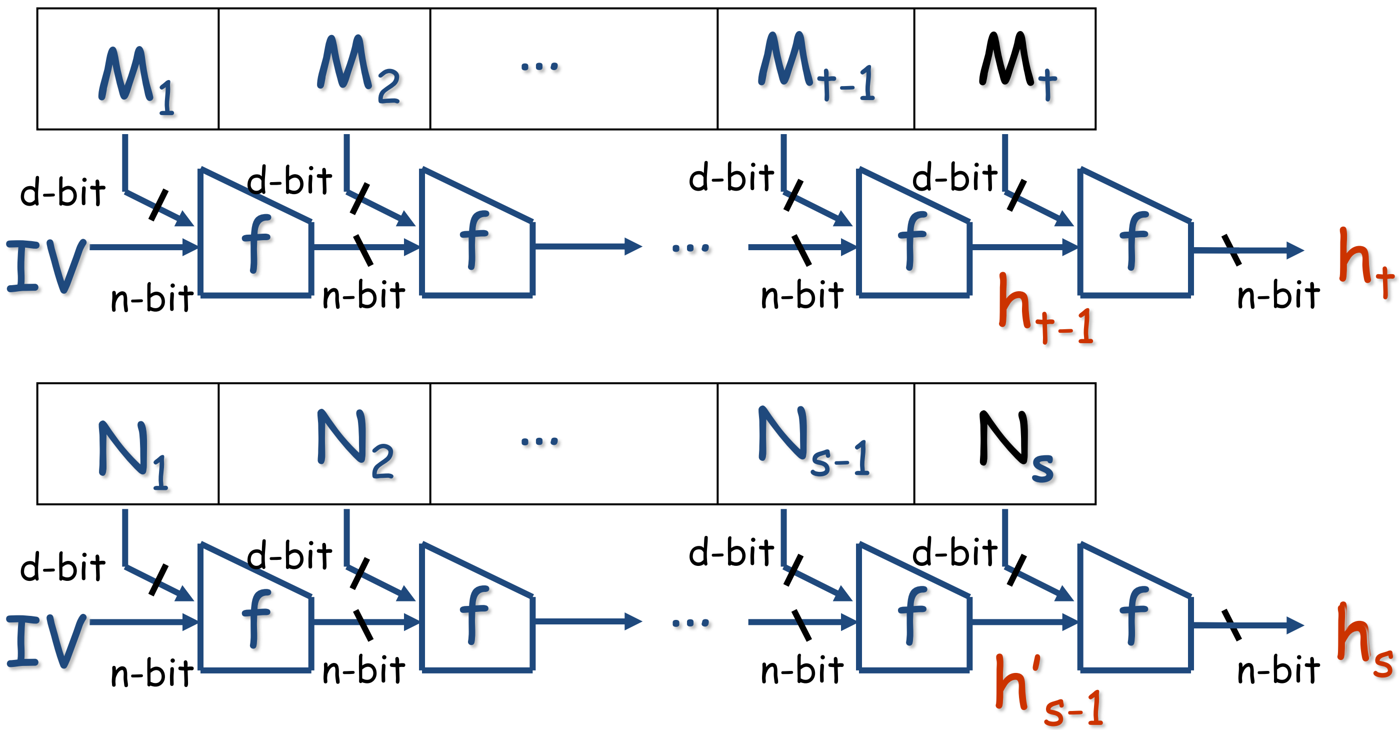


f is CR $\Longrightarrow$ F is CR. (Merkle,Damgård 1989).

Can we prove this?

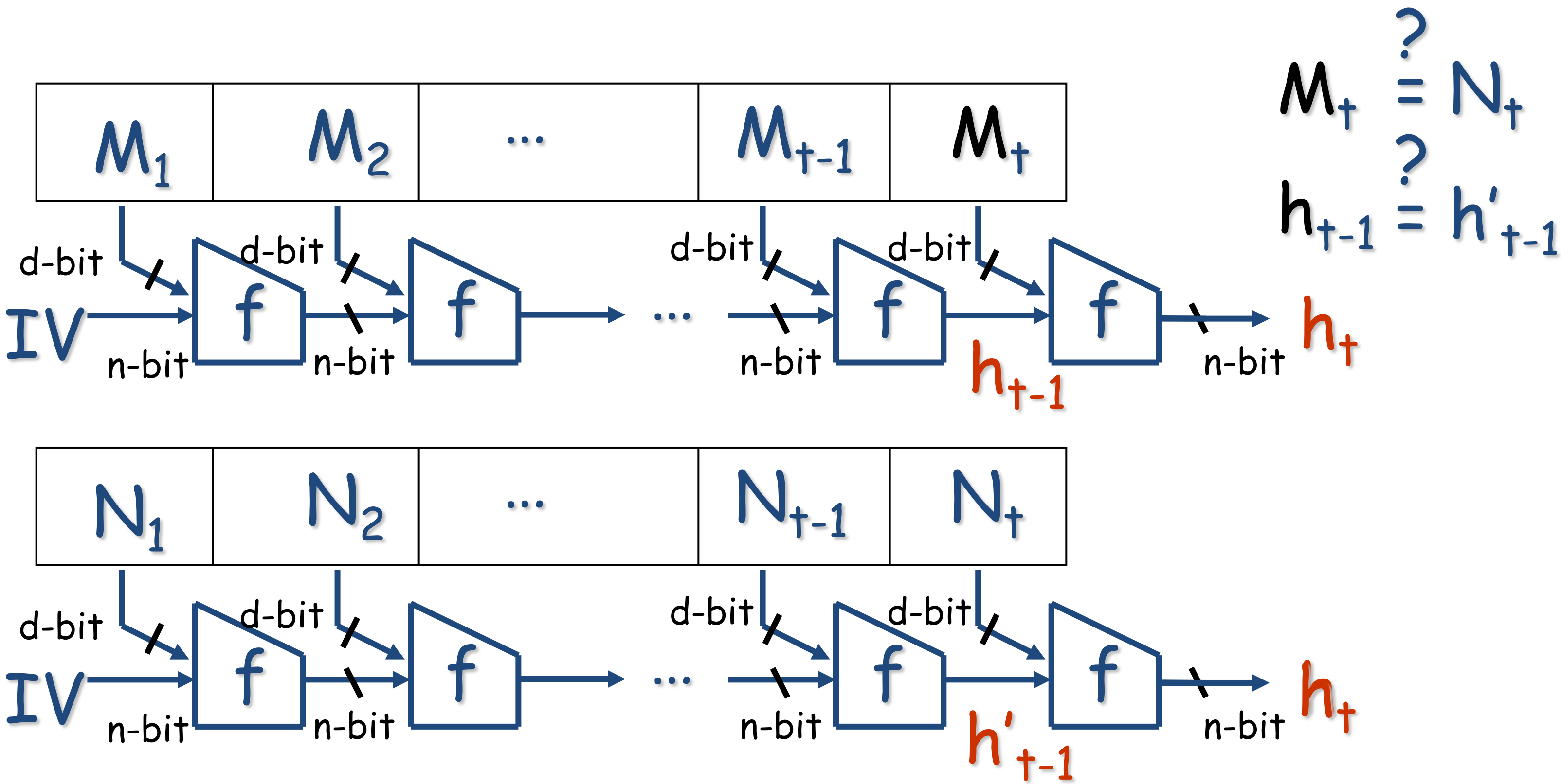
Merkle-Damgård preserves CR (Proof)

If $|M| \neq |N|$ then $M_t \neq N_s$ (both contain the length) hence $f(h_{t-1}, M_t) = f(h'_{s-1}, N_s)$ is collision.



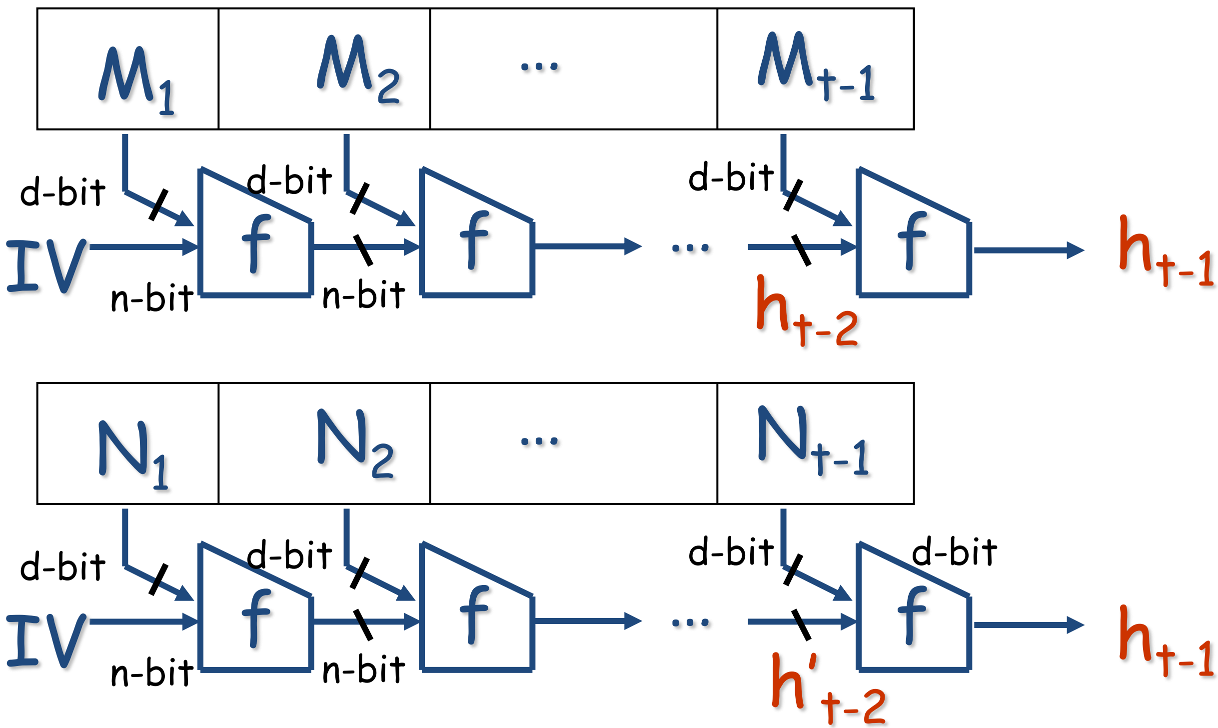
Merkle-Damgård preserves CR (Proof)

So assume |M| = |N|, i.e. s = t.



Merkle-Damgård preserves CR (Proof)

So assume |M| = |N|, i.e. s = t.



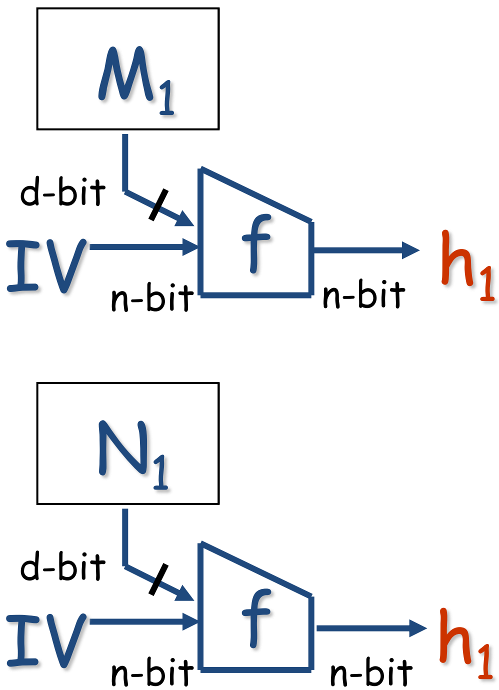
$M_t = N_t$

$M_{t-1} \stackrel{?}{=} N_{t-1}$

$h_{t-2} \stackrel{?}{=} h'_{t-2}$

Merkle-Damgård preserves CR (Proof)

So assume $|M| = |N|$, i.e. $s = t$.



$$M_t = N_t$$

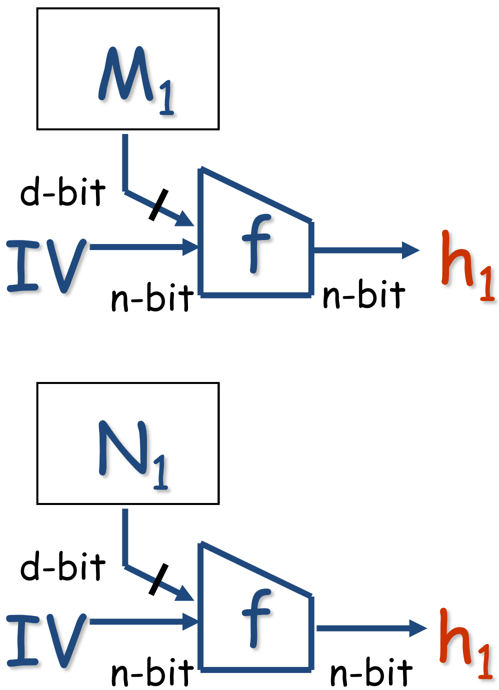
$$M_{t-1} = N_{t-1}$$

$$\vdots$$

$$M_1 \stackrel{?}{=} N_1$$

Merkle-Damgård preserves CR (Proof)

We have $(M_1, \ldots, M_t) = (N_1, \ldots, N_t)$. This implies that $M = N$ (see the padding rule) and hence contradiction.



$$M_t = N_t$$

$$M_{t-1} = N_{t-1}$$

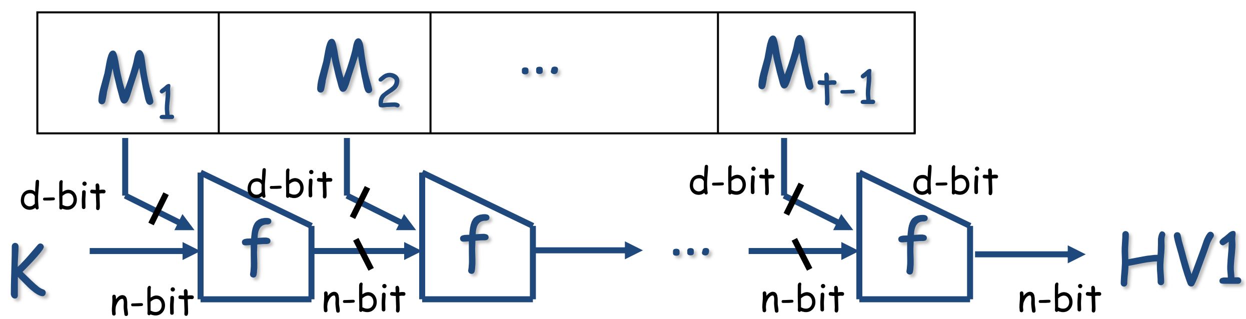
$$\vdots$$

$$M_1 = N_1$$

Keyed Merkle-Damgård ('89)

a. Use key as an initial value: $MD_K(M)$.

b. Prepend key to the message block: $MD_{IV}(K||M)$

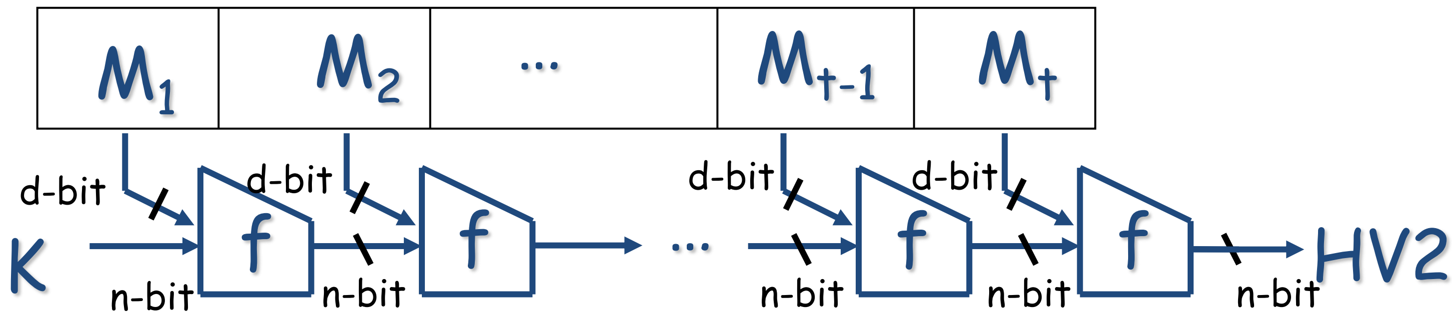


H has length extension attack

Keyed Merkle-Damgård ('89)

a. Use key as an initial value: $MD_K(M)$.

b. Prepend key to the message block: $MD_{IV}(K||M)$



H has length extension attack

$= f(HV1, M_t)$

- Hash based Password authentication is vulnerable to Length extension attack.
- Similar attack can be obtained for $MD_{IV}(K || M)$.

MD is good and bad both…

Are there any other examples?

MD is good and bad both…

Are there any other examples?

YES

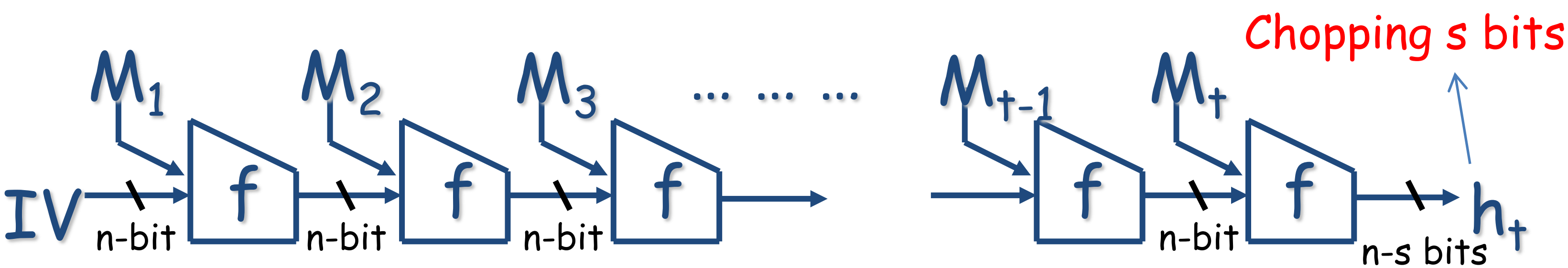
Designs of Hash Function

- We design hash function from a smaller domain function called base function (e.g. compression function).
- E.g. Merkle-Damgård.
- There are other variants of MD.
  - Chop-MD, MD with post-processor.
  - Haifa.
  - Concatenated MD.
  - Doubly iterated, Zipper hash, Generalized MD.
- A sequential design based on non-compressing function
  - **Sponge** Hash function.
- **Now we study the above design of hash functions one by one.**

1. Chop-MD

- Chop Construction (Coron et al. Crypto 2005)

  - any padding rule : for any $M \neq M'$, $Pad(M) \neq Pad(M')$.
  - chopping s bits of output.
  - i.e. the hash size is (n-s).

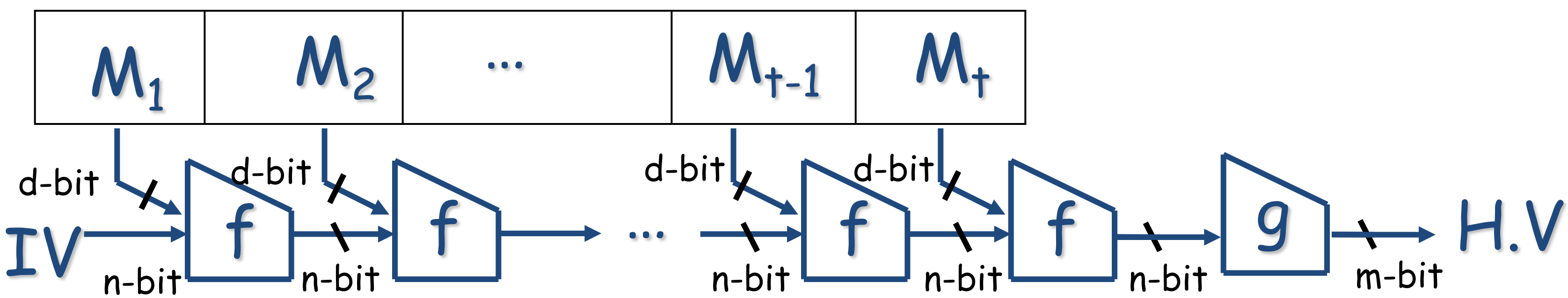


Theorem (Coron et al.)
If f is "Random Oracle" then F (chopping s-bits on MD) has no length extension attack.

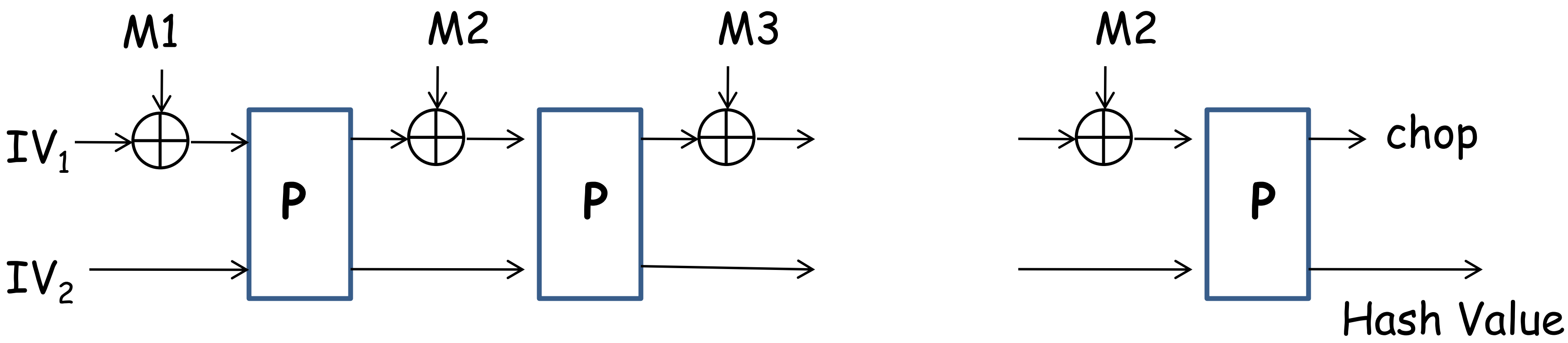
2. MD with Post Processor

- $g: \{0,1\}^n \rightarrow \{0,1\}^m$ be any function, $m \leq n$, called post-processor (chop is one example).

  - $Pad(M) = M \,||\, 10...00 \,||\, binary(|M|)_{64} = M_1 \,||\, ... \,||M_t$



Sponge Mode



- P: $\{0,1\}^{r+c} \rightarrow \{0,1\}^{r+c}$ be a function (permutation).
- $|IV_1| = r$ (bit-rate or hash rate) and $|IV_2| = c$ (capacity measure security guarantee).
- Can be used for
  - Arbitrary length hash outputs.
  - stream-cipher.

- In some application we need <u>larger</u> hash size.

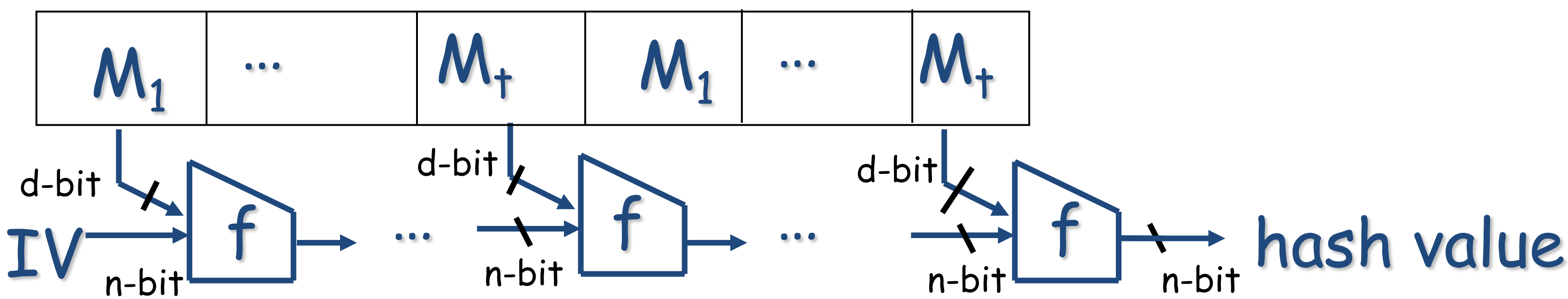
How we can make larger hash size?

3. Concatenated Hash function

- In some application we need <u>larger</u> hash size.
  - One solution: Design a compression function with large n.
  - Double block length hash function: From n-bit compression function how to design 2n-bit hash function.
  - Range extension vs Domain Extension.
- H and G n-bit hash function then H(M) || G(M) is a 2n-bit hash function.
  - Widely used in many industries.
  - Common belief: If H and G are good hash functions and independently designed then H || G has strength like an ideal 2n-bit hash function. - NOT TRUE ALWAYS... (we will see later)

4. Generalized MD

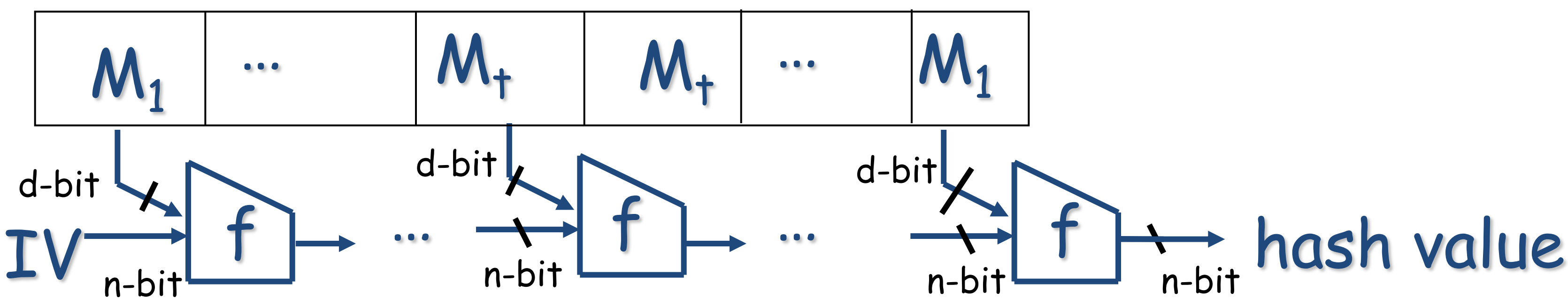
- 1. **Doubly iterated** Merkle-Damgård Construction.



- The sequence is ‹1,2,…, t, 1,2, …, t›

4. Generalized MD

- 2. **Zipper** hash function.



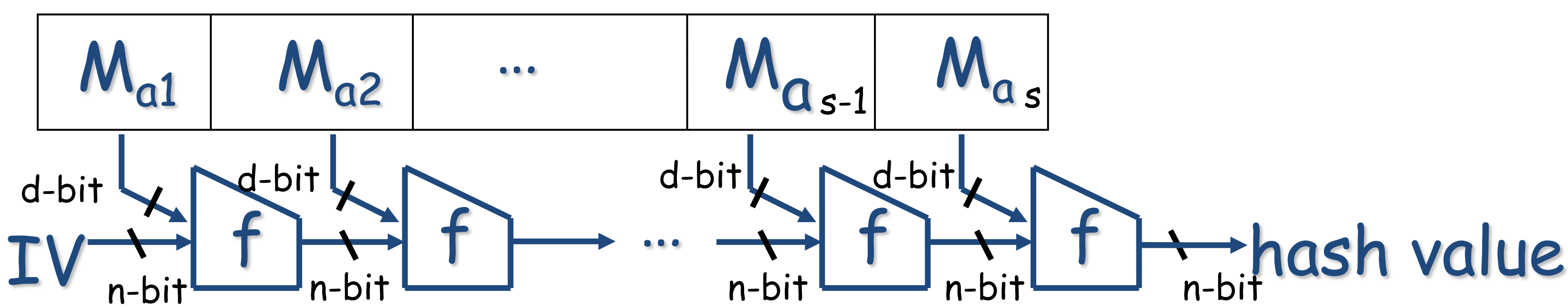
- The sequence is <1,2,…, t, t,t-1, …, 1>
- Classical MD hash function can be characterized by a sequence <1,2,…,t>.

Generalized MD (Nandi, Stinson IEEE'07)

- Generalization of MD (sequence-based): Given any sequence $a = \langle a_1,\dots,a_s \rangle$ of $\{1,2,\dots,t\}$ we can define a hash function $F_a : \{0,1\}^{dt} \rightarrow \{0,1\}^n$ as $F_a(M) = h_s$ where

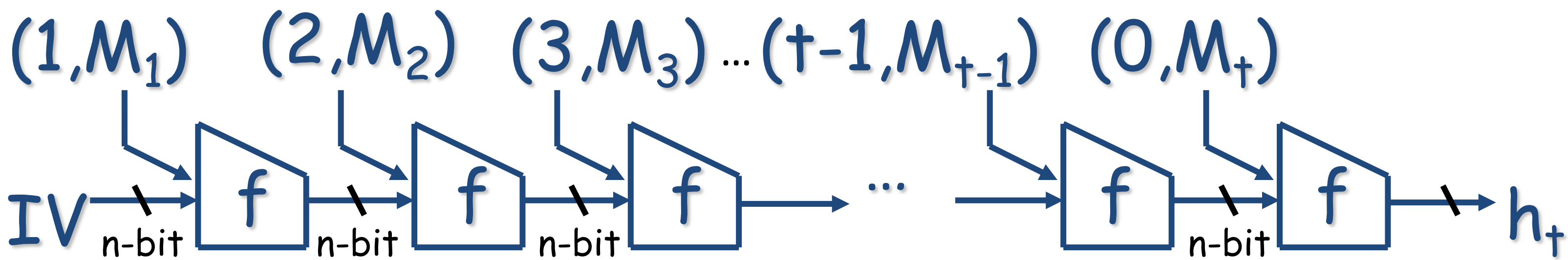
$$h_0 = IV,\ h_i = f(h_{i-1}, M[a_i]),\ i = 1,\dots,s,\ \ M = M[1] || \dots || M[t],$$

- $Pad(M) = M \,||\, 10\dots00 \,||\, binary(|M|)_{64} = M_1 \,||\, \dots \,||\, M_t$



5. HAIFA

Compression function can take counter along with message block and chaining value.



It protects from length extension attack (recall it for MD) and long-message 2nd preimage attack (we describe later).

We know some hash designs …

Which designs SHA-3 finalists use?

SHA-3 Five finalists

- Blake: HAIFA MD design.
  - f(h, m, ctr) = h'. We increase counter one by one in MD chain.
  - Salt can be incorporated.
- Grostl: MD with non-trivial post processor .
  - Chain size: 2n. Post-processor: 2n → n.
- JH: chop-MD
  - Chain size: 2n. chop: 2n → n.
- Keccak: Sponge mode.
- Skein: MD with post-processor.

We talked about a lot of designs. Let's go back to security... rather

some generic attacks...

Birthday Attacks

- **Fact** : If $z_1, \ldots, z_q$ are chosen "*randomly*" (uniformly and independently) from a set A with $|A| = N$ then

  probability of collision (i.e. $z_i = z_j$) is roughly q^2/N.

  - $N=365$, $q=23$ then collision probability is more than $\frac{1}{2}$. In other words, it is more likely that among 23 person, two share same birthday.
  - If $z_i = f(x_i)$, $1 \leq i \leq q$ where x_i is chosen randomly from X and any $f : X \rightarrow A$ with $|X| > |A|$ then collision probability is no less than the birthday collision probability.

Birthday Attcks

- Suppose H is a hash function with hash size n.
- **Collision Attack:**
  - Choose $M_1, \ldots, M_q$ at random and compute $z_i = H(M_i)$.
  - Find collision on z_i's (i.e. $M_i \neq M_j$ but $z_i = z_j$).
  - On the average we expect at least one collision in $2^{n/2}$ tries.

Birthday Attcks

- Suppose H is a hash function with hash size n.
- **Collision Attack:**
  - Choose $M_1, \ldots, M_q$ at random and compute $z_i = H(M_i)$.
  - Find collision on z_i's (i.e. $M_i \neq M_j$ but $z_i = z_j$).
  - On the average we expect at least one collision in $2^{n/2}$ tries.
- **Preimage Attack:** given z, choose $M_1, \ldots, M_q$ at random until we get $z = H(M_i)$ for some i.

Birthday Attcks

- Suppose H is a hash function with hash size n.
- **Collision Attack:**
  - Choose $M_1,\ldots, M_q$ at random and compute $z_i= H(M_i)$.
  - Find collision on z_i's (i.e. $M_i \neq M_j$ but $z_i = z_j$).
  - On the average we expect at least one collision in $2^{n/2}$ tries.
- **Preimage Attack:** given z, choose $M_1,\ldots, M_q$ at random until we get $z= H(M_i)$ for some i.
- **2^nd-Preimage Attack:** given M first compute $z = H(M)$ then choose $M_1,\ldots, M_q$ at random until we get $z= H(M_i)$ for some i and $M \neq M_i$.

Complexity of the birthday (2^{nd}-) preimage attack?

- **2^n hash outputs are required to succeed.**

Multicollision

- Generalization of collision : (distinct) $x_1,\ldots, x_k \in X$ are said to be k-multicollision tuple of $f : X \rightarrow \{0,1\}^n$ if

$$f(x_1) = \ldots = f(x_k).$$

k=2, simply called collision.

- Birthday attack: If f is RO then for any $x_1, \ldots, x_q$ there is a k-multicollision of f with probability $O(q^k/2^{n(k-1)})$. In other words, **we need at least $2^{n(k-1)/k}$ queries** to get a multicollision.

Nosterdamus Attack

- Commit h.
- Given any M, find r such that H(M, r) = h
- Finding r: a kind of preimage?
  - Not exactly, in case of MD (we will see later).
  - **Generic attack:** choose r at random until we find H(M, r) = h. complexity: 2^n.

- **Why it is called Nosterdamus attack?**

- Commit for future event and reveal once we reach that future time point. - used for Prediction.

Ideal Hash Function: Random Oracle

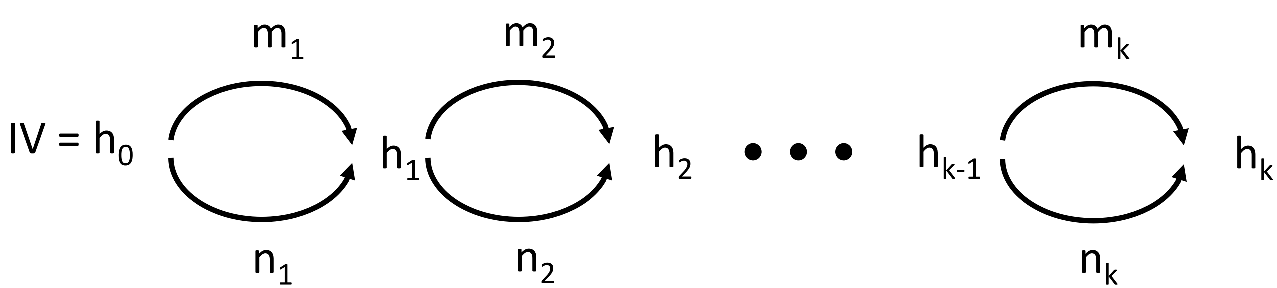
Random Oracle

- An n-bit hash function H is called random oracle if for any distinct inputs $M_1, \ldots M_q$, $H(M_1), \ldots, H(M_q)$ are uniformly and independently distributed over $\{0,1\}^n$.
- Hash functions are usually assumed to be a random oracle. For any distinct choices of $x_1, \ldots, x_q$ we have the birthday collision probability.
- Random oracle is ideal: For any attack (not necessarily birthday attack)
  - Collision. **query : $2^{n/2}$**.
  - Preimage, Nosterdamus attack, second preimage- **2^n**.
  - k-multicollision **$2^{n(k-1)/k}$ queries**

Can we have attacks better than generic attacks ??

Joux's Multicollision

Notation : $h_0 \xrightarrow{m_1} h_1$ $f(h_0, m_1) = h_1$



- k successive birthday attacks.
- $H(M) = h_k$ for any $M = x_1x_2\ldots x_k$ where $x_i = m_i$ or n_i.
- 2^k-multicollision based on $k2^{n/2}$ queries.

Joux's Multicollision

- What we will do if we do not get collision at some stage after $2^{n/2}$ tries?.
- We make sufficient number of queries.. Even if we do not get we abort
  - step back or
  - change chaining value
- What we will do if $d < n/2$?
  - We combine two or more blocks so that message size in combined block is at least $n/2$.

Application: Joux's Multicollision

- Collision for concatenated DBL Hash $H \,||\, G$:
  - $2^{n/2}$ -way multicollision for H in $O(n2^{n/2})$ complexity.
  - Assume G as RO. We expect a collision pair (M, M') for G. So,

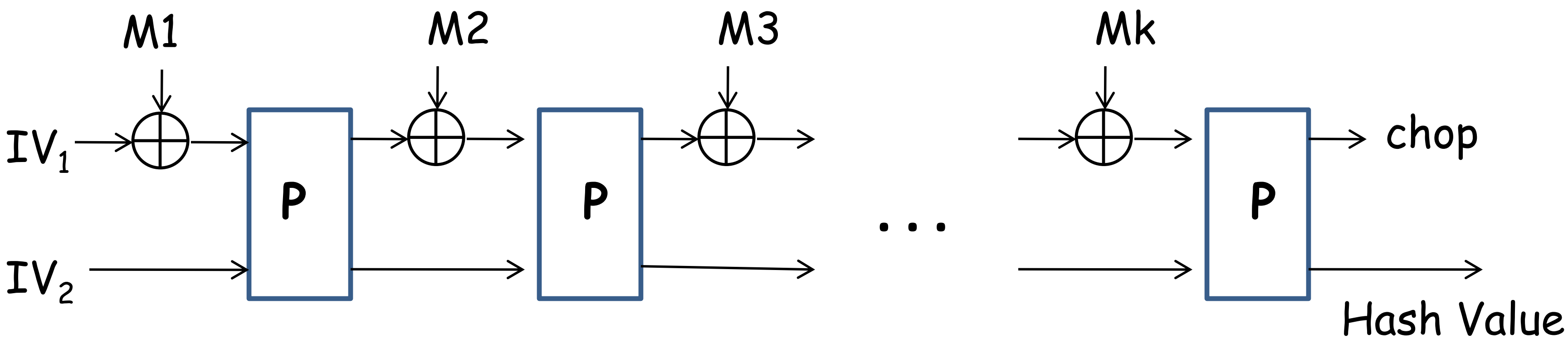
    $$H(M) \,||\, G(M) = H(M') \,||\, G(M').$$

  - $n2^{n/2}$ complexity for collision of 2n-bit hash function.
  - **Open Problem** : To find collision without assuming ROM.

Application: Joux's Multicollision

- Find 2^{nd} preimage of long messages (2^k blocks).
  - Query complexity: $2^{n-k} + k\ 2^{n/2}$.
- **Pre-processing step:** We can use the idea of Joux's attack to find expandable messages.
- **Step-1**: Find a link message to the chain of the given message M.
- **Step-2**: Use appropriate length message from the expandable message set.

Sponge Mode

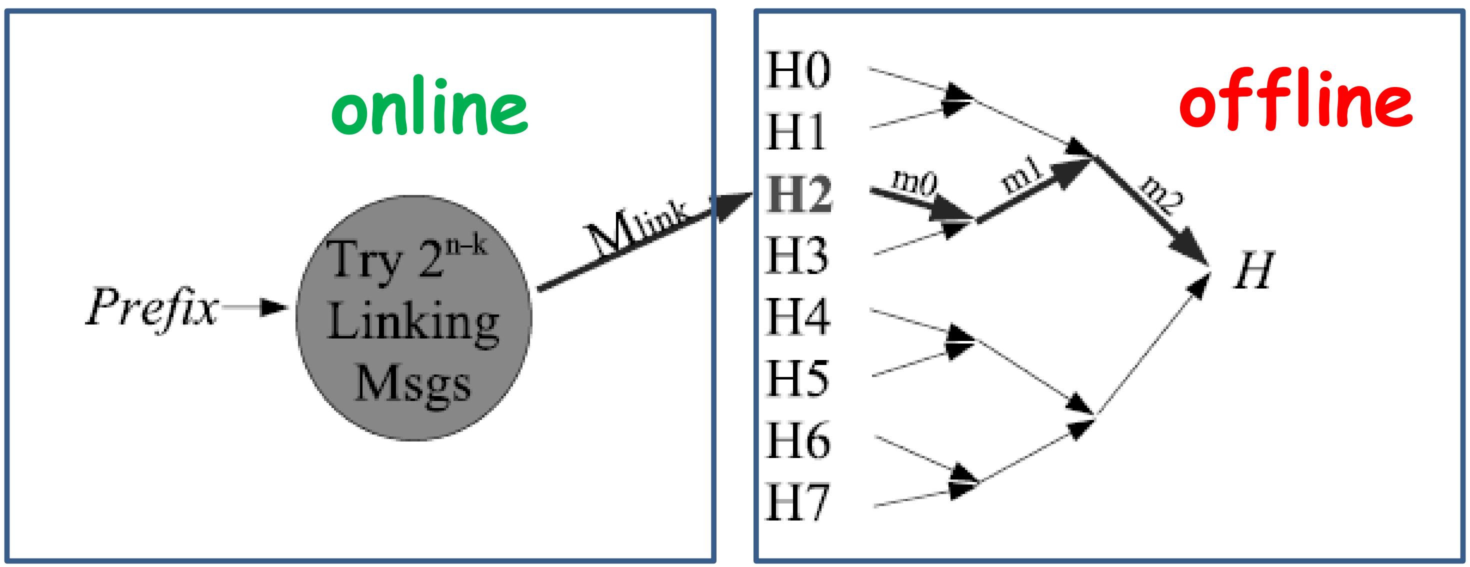


- We have multicollision similar to MD.
  - Can you see this?
- Preimage attack: Meet-in-the middle attack (given M and $E_{K2}(E_{K1}(M)) = C$, how to find K1 and K2 ?) . Here given h, find M:
  - $2^{c/2}$ **queries** to find a preimage.

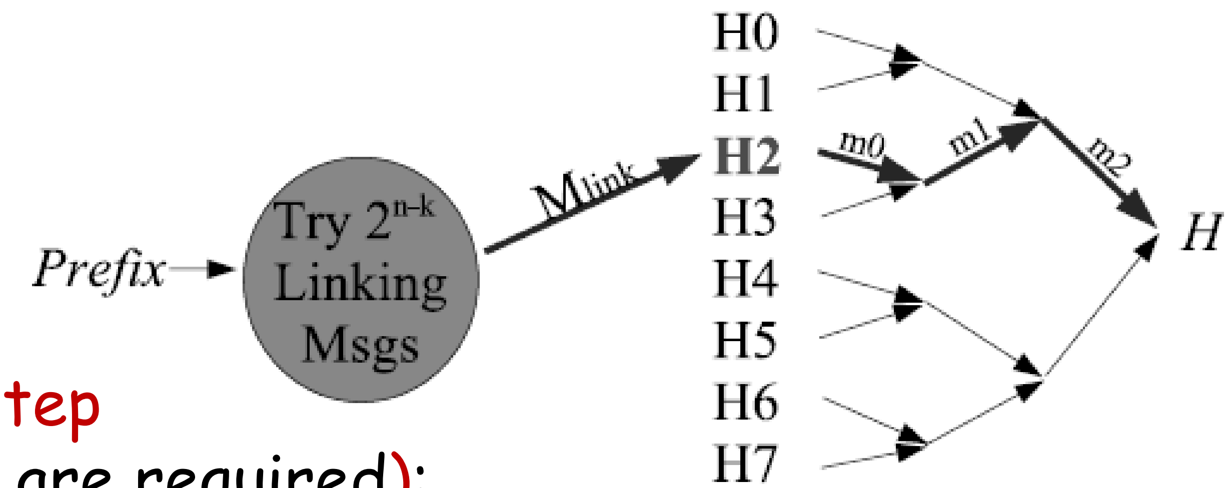
Nosterdamus Attack for MD

- Commit h.
- Given any M find r such that H(M, r) = h

- We have Diamond attack.
  - roughly $2^{n-k} + 2^{k/2+n/2}$ hash queries.

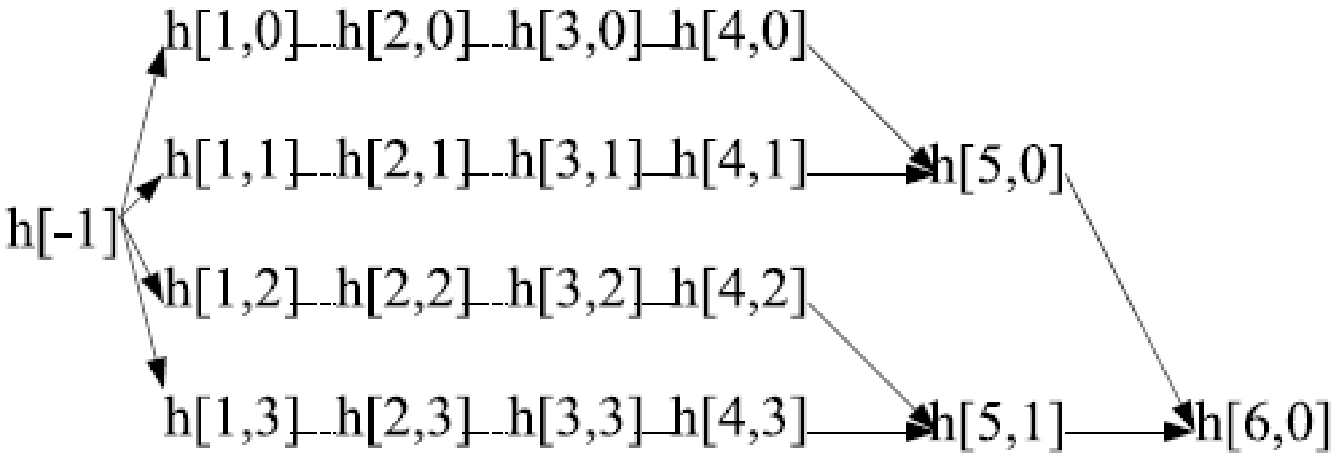


Nosterdamus Attack for MD

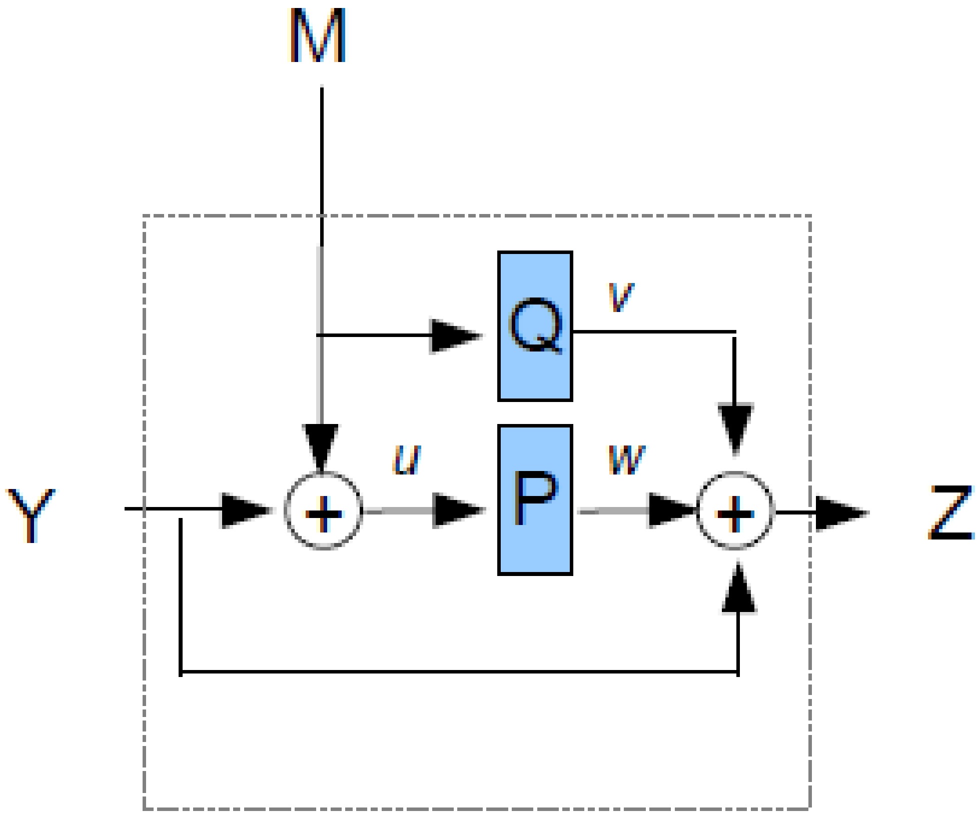


- preprocessing step ($2^{n/2 + k/2}$ queries are required): Make the tree to obtain H (root node) and commit it.
- Given M compute the partial chain value h: $IV \rightarrow_M h$.
- Find a link Mlink s.t. $h \rightarrow_{Mlink} Hi$ for some i (here i = 2).
  - 2^{n-k} queries are required
- Let $Hi \rightarrow_N H$. Then we have, $IV \rightarrow_{M \,||\, Mlink \,||\, N} H$
  - i.e. MD(M || Mlink || N) = H. So r = Mlink || N.

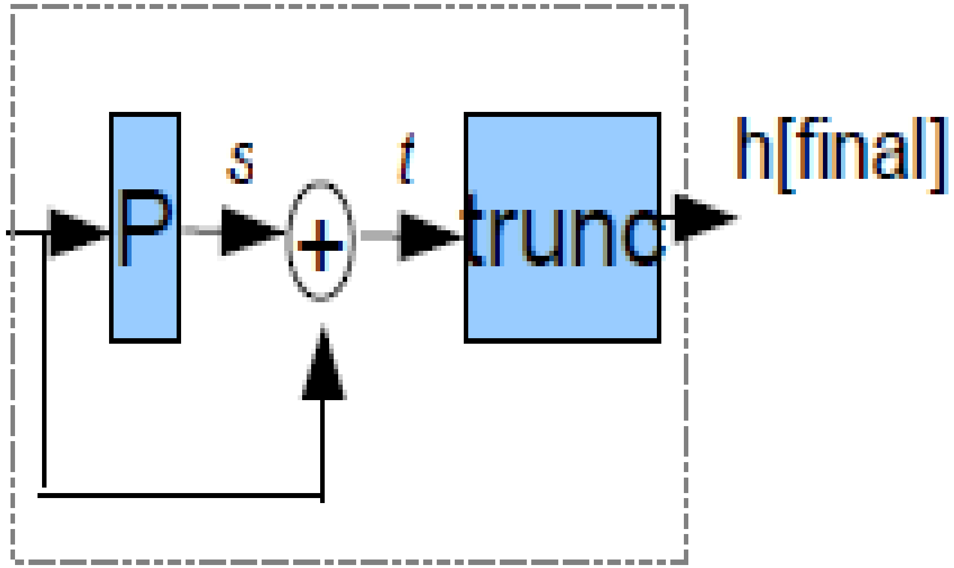
Elongated Diamond Structure



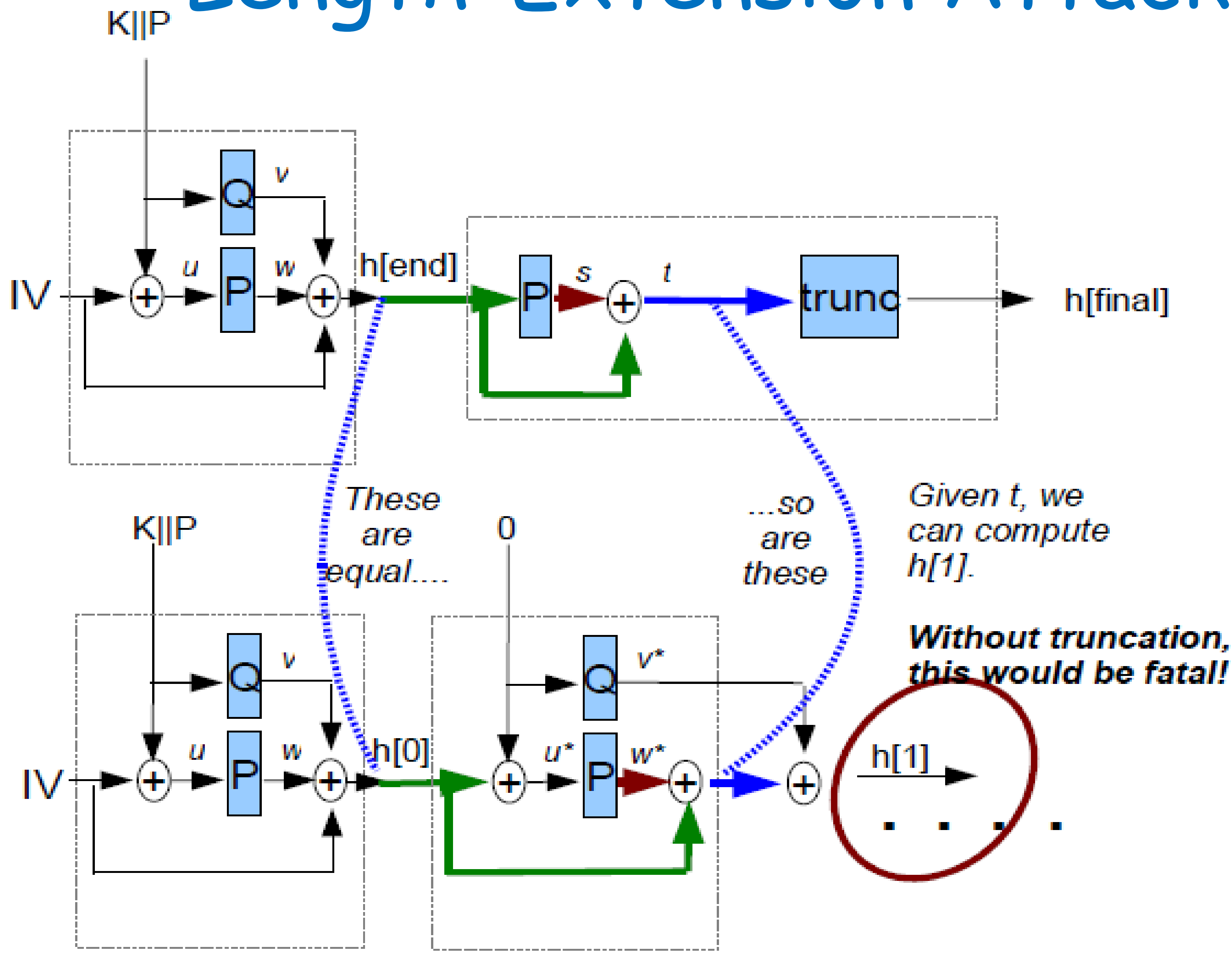
Grostl Compression function



Grostl post-processor



Length-Extension Attack



Thank you

Comments and Questions ?

Security Requirements: Hash Function

(The MOST POPULAR)

- (1) collision resistance.
- (2) Preimage resistant.

The others

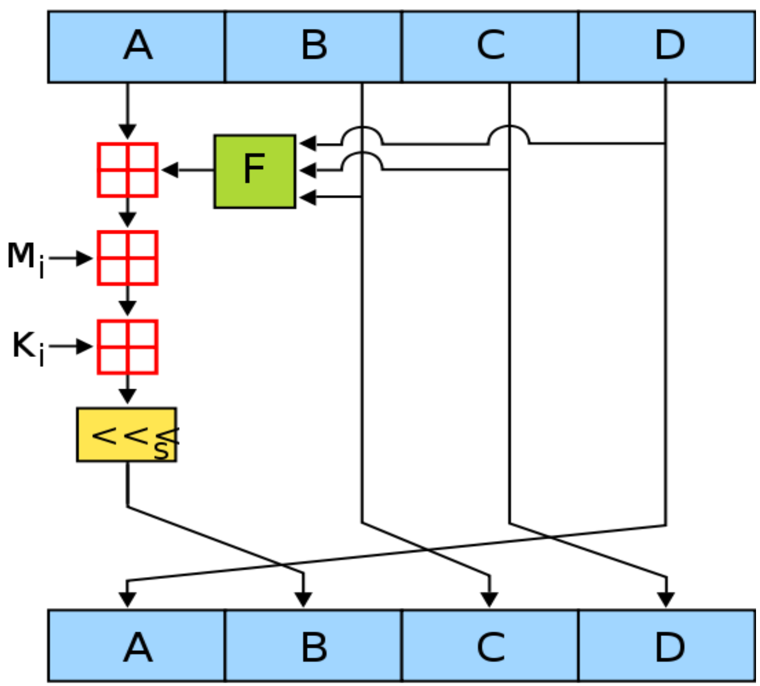
- (3) 2nd preimage resistant
- (4) multicollision resistant.
- (5) Target collision resistant or UOWHF.
- (6) resistant against length-extension attack
- (7) Herding attack.
- (8) indistinguishability in outputs. PRF, PRO..
- ETC...

SWISS ARMY KNIFE

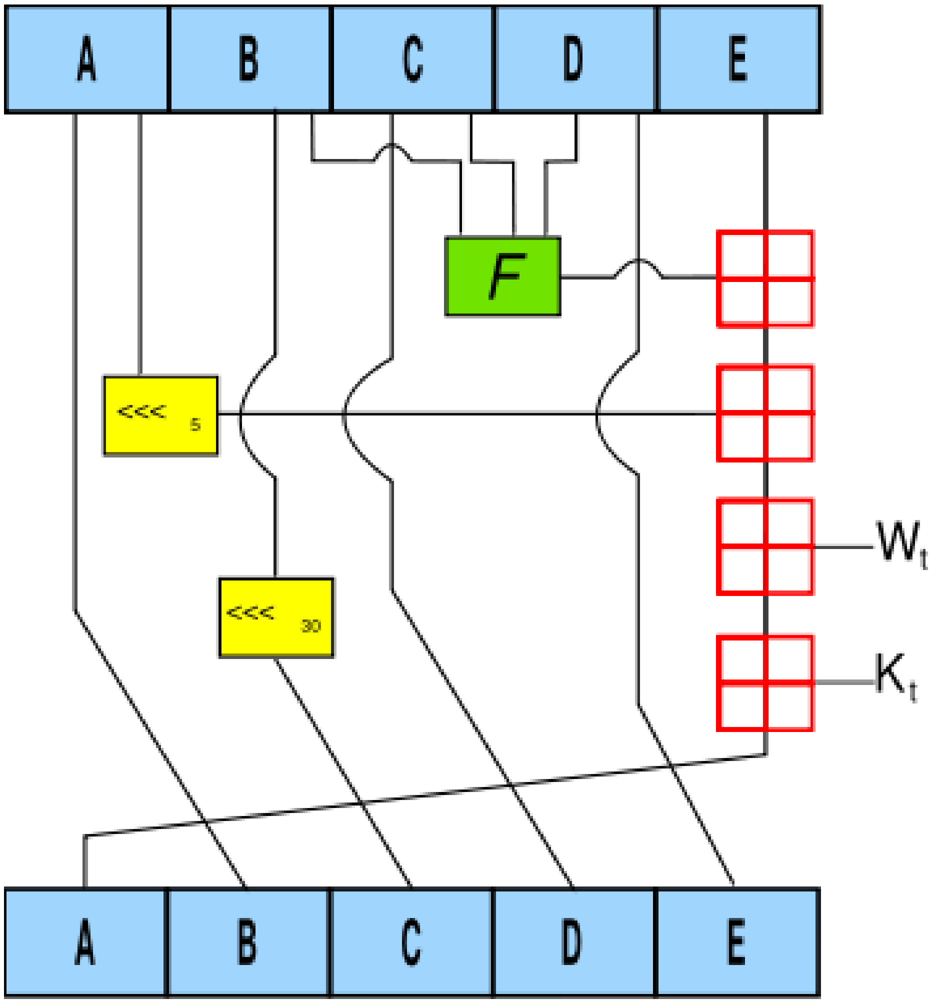
Some compression functions…

MD4 compression function

- MD4 compression function f: $\{0,1\}^{128} \times \{0,1\}^{512} \rightarrow \{0,1\}^{128}$.
  - F(h, m) = h'
  - message expansion $M_1 \,||\, \ldots\, M_{16} \,||\, M_1 \,||\, ..\, M_{16} \,||\, M_1 \,||\, \ldots \,||\, M_{16}$.
  - h = A || B || C || D. Update h 48 times as shown in figure.



SHA-1 compression function



SHA-2 compression function