# State-of-the-art of Hash Functions

*Thomas Peyrin*

Coding and Cryptography Research Group - NTU

**Applied Cryptography Workshop**

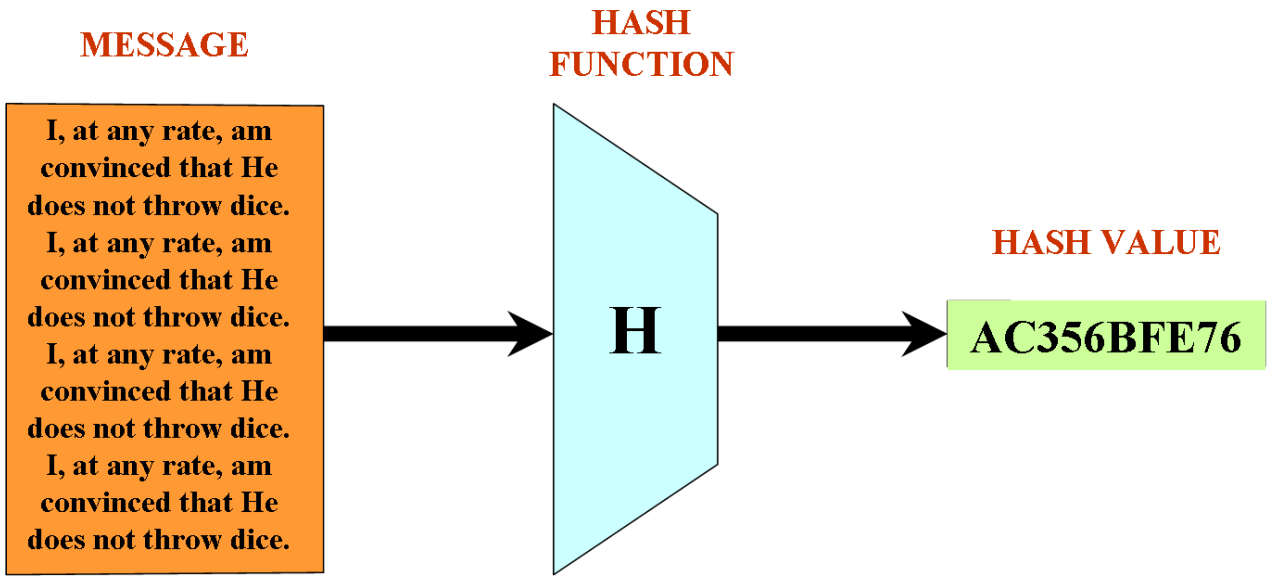
Singapore - December 3, 2010

NANYANG TECHNOLOGICAL UNIVERSITY

# Outline

# Outline

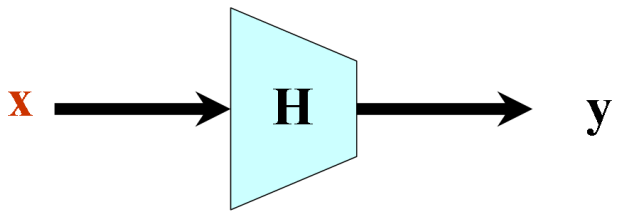
# What is a Hash Function ?




- $H$ maps an **arbitrary length input** (the message $M$) to a **fixed length output** (typically $n = 128$, $n = 160$ or $n = 256$).
- no secret parameter.
- $H$ must be easy to compute.

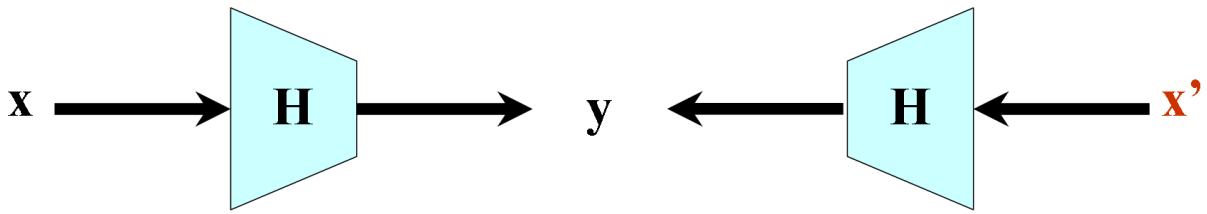
## The security goals

- **pre-image resistance:** given an output challenge $y$, the attacker can not find a message $x$ such that $H(x) = y$, in less than $\theta(2^n)$ operations.
- **2nd pre-image resistance:** given a challenge $(x, y)$ so that $H(x) = y$, the attacker can not find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.
- **collision resistance:** the attacker can not find two messages $(x, x')$ such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).



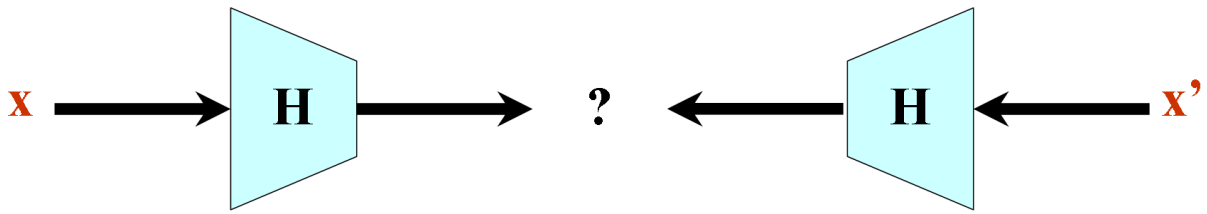
## The security goals

- **pre-image resistance:** given an output challenge $y$, the attacker can not find a message $x$ such that $H(x) = y$, in less than $\theta(2^n)$ operations.
- **2nd pre-image resistance:** given a challenge $(x, y)$ so that $H(x) = y$, the attacker can not find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.
- **collision resistance:** the attacker can not find two messages $(x, x')$ such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).



## The security goals

- **pre-image resistance:** given an output challenge $y$, the attacker can not find a message $x$ such that $H(x) = y$, in less than $\theta(2^n)$ operations.
- **2nd pre-image resistance:** given a challenge $(x, y)$ so that $H(x) = y$, the attacker can not find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.
- **collision resistance:** the attacker can not find two messages $(x, x')$ such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).



## The security goals

- **pre-image resistance:** given an output challenge $y$, the attacker can not find a message $x$ such that $H(x) = y$, in less than $\theta(2^n)$ operations.
- **2nd pre-image resistance:** given a challenge $(x, y)$ so that $H(x) = y$, the attacker can not find a message $x' \neq x$ such that $H(x') = y$, in less than $\theta(2^n)$ operations.
- **collision resistance:** the attacker can not find two messages $(x, x')$ such that $H(x) = H(x')$, in less than $\theta(2^{n/2})$ operations (a generic attack with the birthday paradox exists [Yuval-79]).

And other ones: near collisions, multicollisions, random oracle look-alike, ...

# Applications (1)

## Many applications:

- **Signatures.** a signature scheme allows users knowing the public key to verify the integrity and the authenticity of a message signed by the holder of the corresponding private key. Hash functions are used to speedup and increase the security of signature schemes.

- **Message Authentication Codes.** a MAC allows two users sharing a secret key to check the integrity and the authenticity of the message sent. HMAC is one famous way of building a MAC and it uses a hash function as main internal component (used for example in SSL/TLS, IPSec, ...)

- **Integrity check.** used for example in HTTP, FTP or P2P downloading.

# Applications (2)

**Many applications:**

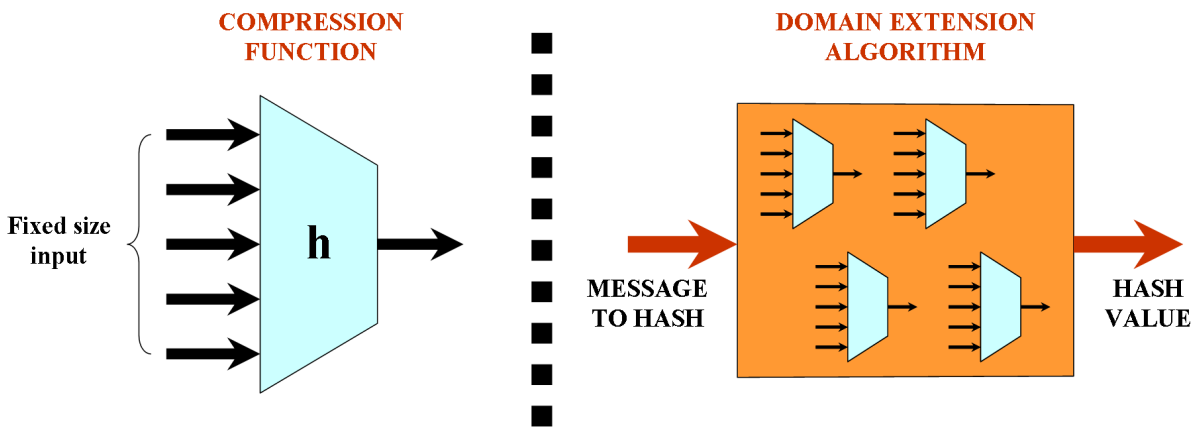
- **Passwords database protection.** instead of storing all passwords in a database on the server side for client authentication, one should instead store the hash value of the passwords.
- **Confirmation of knowledge/commitment on a value.** if someone would like to prove that he knows some information without actually revealing it, he can publish the hash value of this information. Once the information public, one can easily verify his commitment.
- **PRNG / key derivation.** hash functions are often utilized to destroy all potential structure that may exist on the input data, while preserving its entropy (for example algebraically structured objects).

# Outline

## General construction

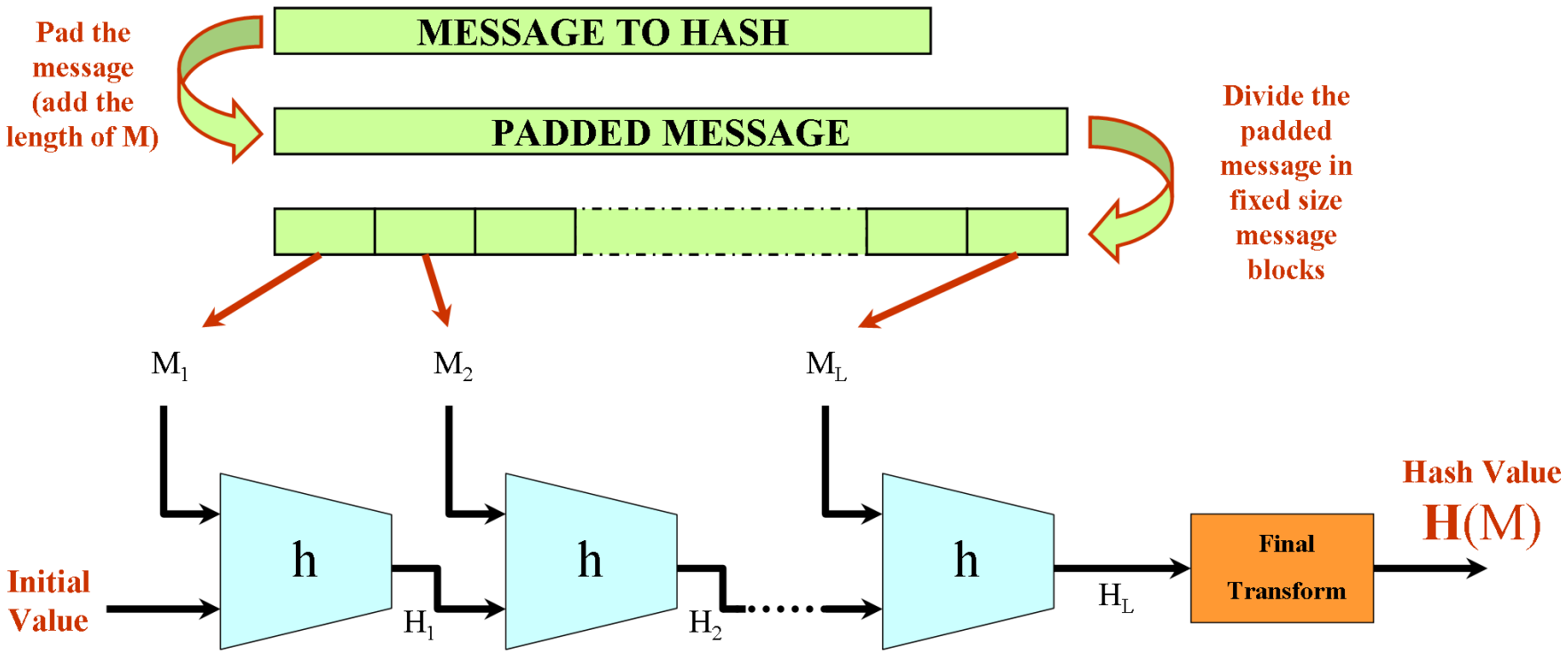
For historical reasons, most hash functions are composed of two elements:

- **a compression function** $h$**:** a function for which the input and output size is fixed.
- **a domain extension algorithm:** an iterative process that uses the compression function $h$ so that the hash function $H$ can handle inputs of arbitrary lenght.



# The Merkle-Damgård domain extension algorithm

The most famous domain extension algorithm used is called the **Merkle-Damgård** [Merkle Damgård-89] iterative algorithm.



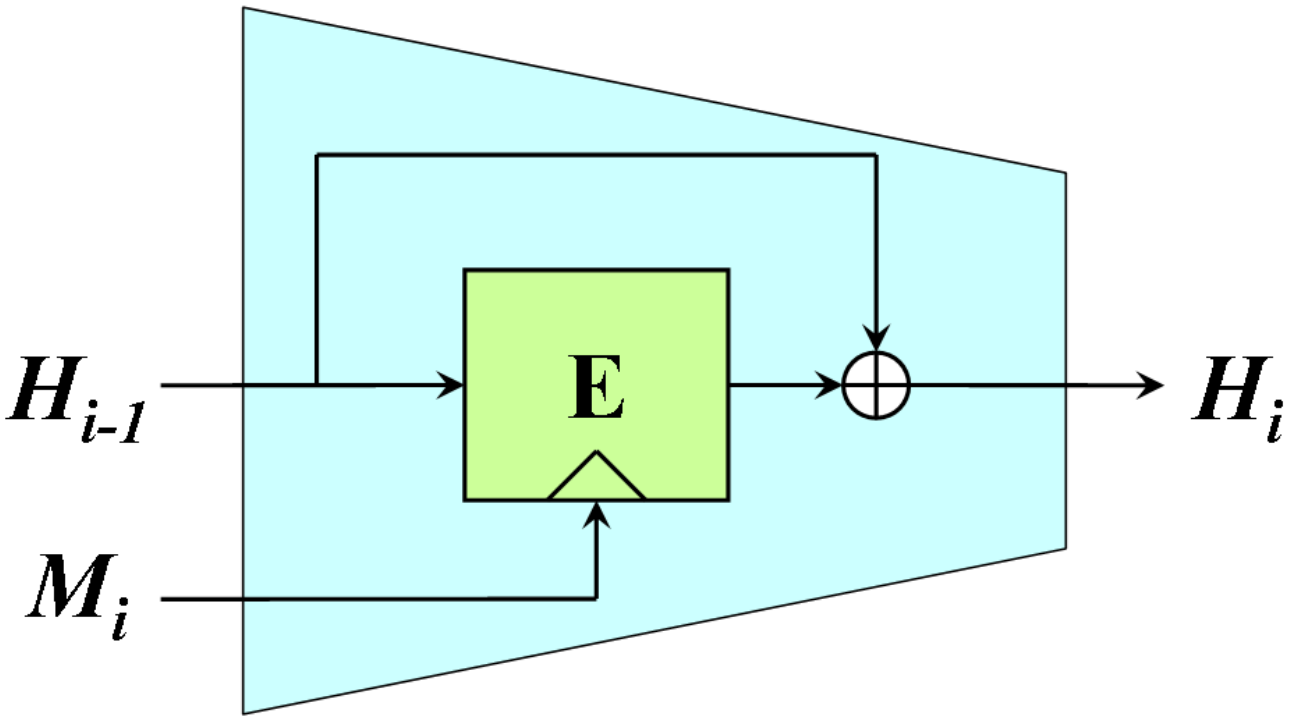
## Advantages and drawbacks of the Merkle-Damgård algorithm

- **Avantage:** the Merkle-Damgård algorithm allows to reduce the problem of building a collision/preimage-resistant hash function to the problem of building a collision-resistant compression function : an atatcker that found collision/preimage on $H$ necessarily found a preimage/collision on $h$.
- **Drawback:** collision/preimage resistance is not everything. Recently, it has been shown that Merkle-Damgård is very different from an ideal domain extension algorithm: multi-collisions, long 2nd preimages, ...

As a consequence, **many new and more complex algorithms have been recently proposed** (HAIFA, sponge functions, double-pipe, ...), leading to a better emulation of a random oracle.

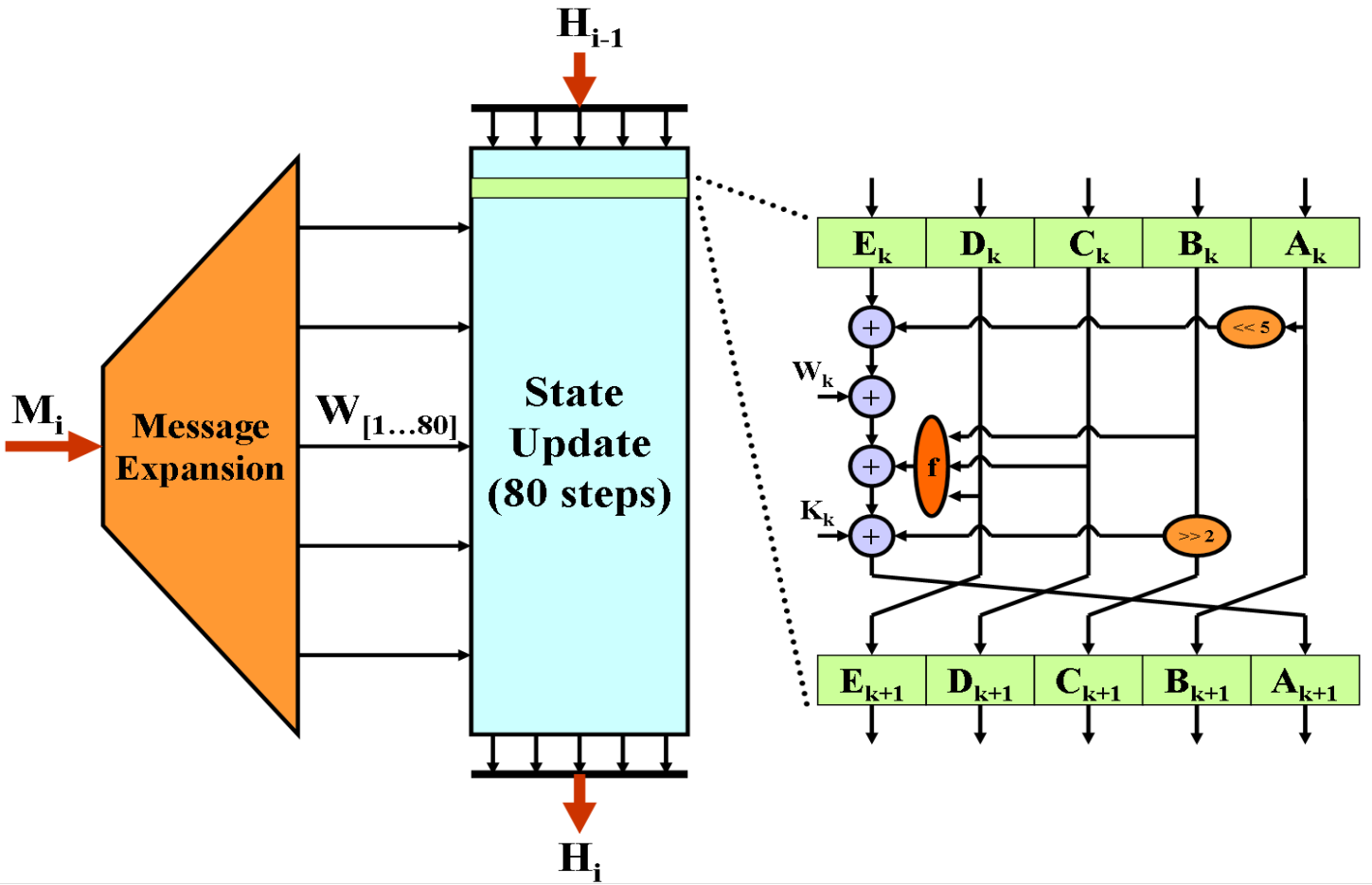
# The compression function

**The MD/SHA family** (MD4, MD5, SHA-0, SHA-1, SHA-2, ...)



# The compression function

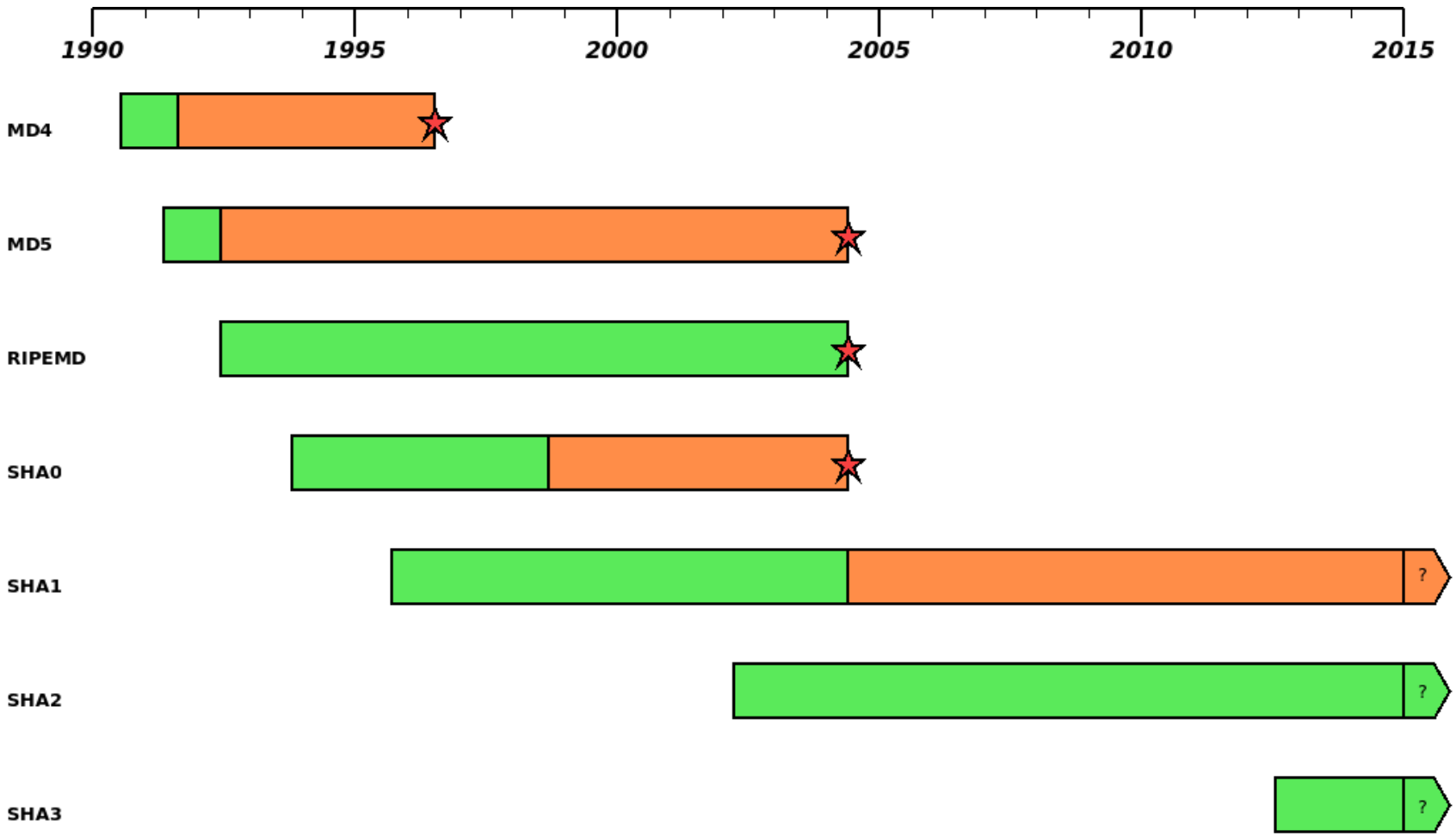
**The MD/SHA family** (MD4, MD5, SHA-0, SHA-1, SHA-2, ...)



# Hash functions efficiency

| Type | Algorithm | Speed (MiB/second) |
|---|---|---|
| Block Cipher | DES | 34 |
| Block Cipher | IDEA | 36 |
| Block Cipher | AES | 90 |
| Hash Function | CRC-32 | 256 |
| Hash Function | MD5 | 258 |
| Hash Function | SHA-1 | 155 |
| Hash Function | SHA-2 | 81 |
| MAC | HMAC(SHA-1) | 152 |
| MAC | CBC-MAC(AES) | 86 |

# Attacks on the MD/SHA family



## Collision attacks

**At the CRYPTO 2004 rump session, a collision is given for MD4, MD5 et RIPE-MD !**

Several months later (February 2005), **SHA-1 is broken** ! ... only theoretically, but nobody expected it !

- achievement after 10 years of work by the Wang et al.'s team ...
- ... most of the attack has been found by hand !
- the very same method applies to MD4, MD5, SHA-0, SHA-1, RIPEMD
- but for a long time everything remained blurry: very complex attack, bad explanations, some minor errors ...

## Recent improvements

The following years, researchers gave a **more sound and theoretical approach** of the Wang et al. attacks:

- better differential paths (perturbation vector, non-linear part)
- better freedom degrees utilization (message modifications, neutral bits, boomerang, ...)
- the automation of the tools led to better results

Cryptanalysts also looked at **other properties** of hash functions:

- preimages (using complex meet-in-the-middle approaches)
- HMAC (distinguisher, forgery or key-recovery)
- forgery of valid X.509 certificates signed using MD5 (Lenstra et al. 2009)

# The current security status of the MD/SHA family

| Algorithm | Output size | collision | preimage | HMAC |
|---|---|---|---|---|
| MD4 (1990) | 128 | $2^{1}$ | $2^{95}$ | $2^{58}$ |
| MD5 (1992) | 128 | $2^{16}$ | $2^{124}$ | $2^{47}$ |
| SHA-0 (1993) | 160 | $2^{33}$ | 52/80 rds | $2^{84}$ |
| SHA-1 (1995) | 160 | $2^{60}$ | 48/80 rds | 62/80 rds |
| SHA-2 (2002) | 256 | 24/64 rds | 43/64 rds | 52/64 rds |

# Outline

# SHA-3 competition

**The SHA-3 hash function competition:**

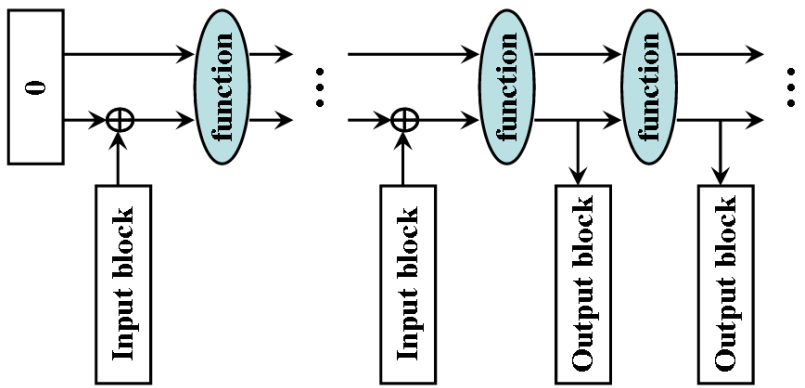
- started in October 2008, 64 submissions
- 51 candidates accepted for the first round
- 14 semi-finalists selected in 2009
- 4/5/6 finalists to be selected end 2010
- winner to be announced in 2012

**Motivated new designs and new attacks**

# New domain extension algorithms

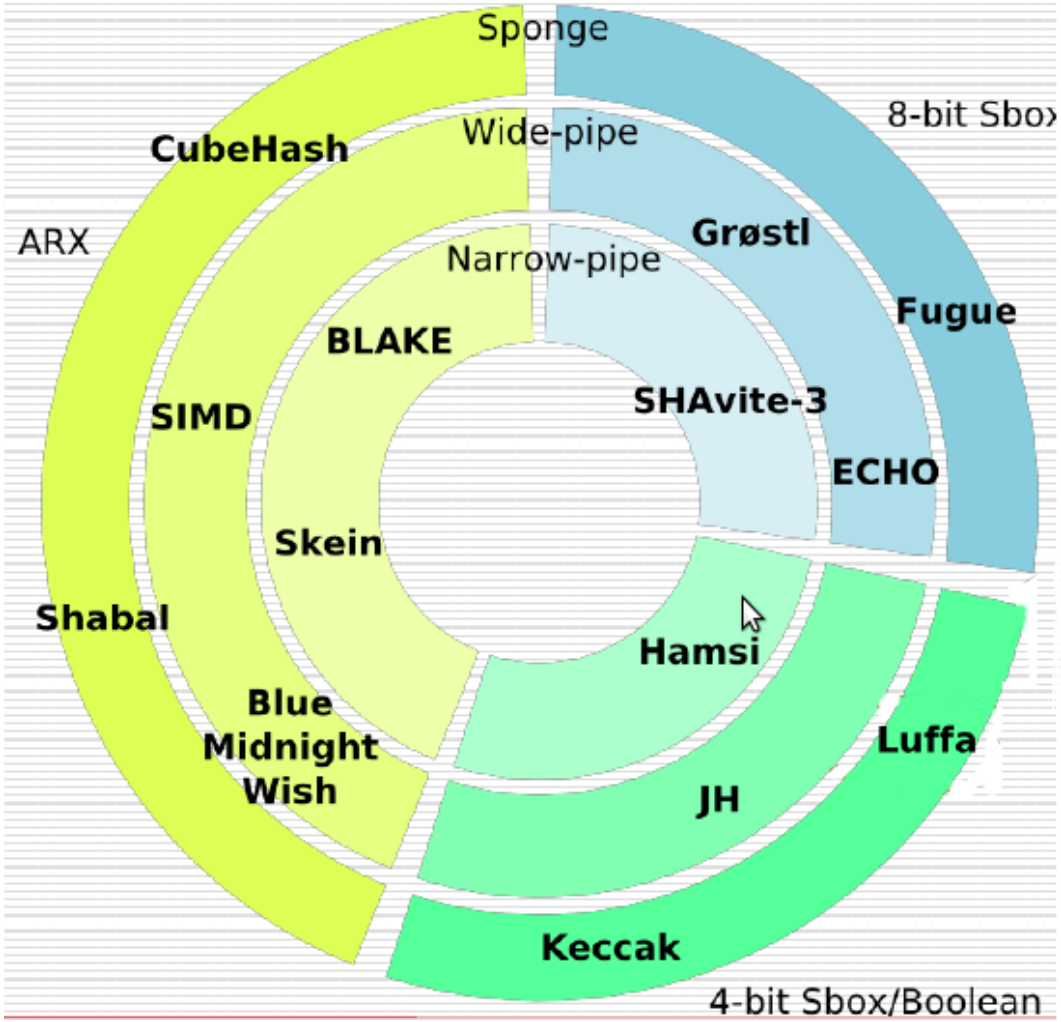
**The SHA-3 competition motivated new domain extension algorithms:**

- **HAIFA framework.** the idea is to add an iteration counter to the compression function and to fully integrate the salt input into the hash function.
- **Double-pipe.** the idea is to increase the size of the internal state of the hash function (at least double in order to get a chance to simulate a randon oracle)
- **Sponge functions.**



# New permutation and compression function designs

**The SHA-3 competition motivated new comp. function designs:**



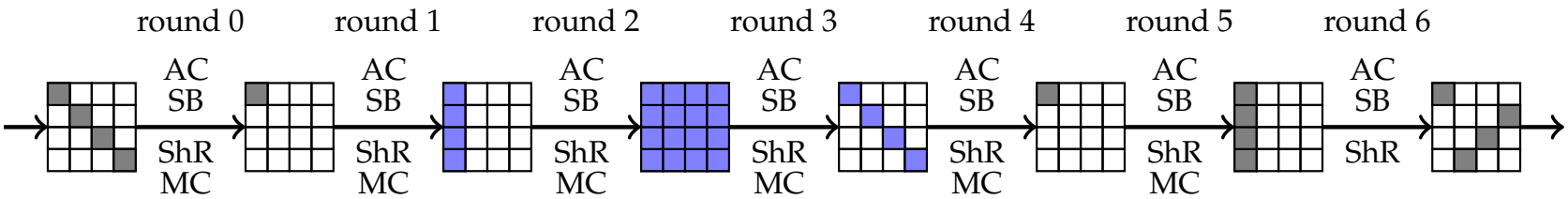

©De Cannière et al.

# New attacks

## The rebound attack (Mendel et al. 2009)

**The controlled rounds:** the part where the freedom degrees are used (usually in the middle of the path). On average, finding a solution for the controlled rounds should cost only a few operations.

**The uncontrolled rounds:** the part where all the events are verified probabilistically (left and right part of the path) because no more freedom degree is available. Determine the complexity of the overall attack.




- broke many AES-based candidates
- several variants/improvements proposed: start-from-the-middle attack, multiple inbounds attack, Super-Sbox attack, ...

# New attacks

- **rotational cryptanalysis.**
  idea: check what happens in the computation between some input message and a rotated variant of it. It works well for ARX oriented designs.

- **internal differential attack.**
  idea: for parallel branches oriented schemes, look for the difference between the internal branches (and not between two input pairs).

- **cube attack.**
  idea: try to represent an output bit with a sufficiently low degree polynomial over GF(2) of message bits.