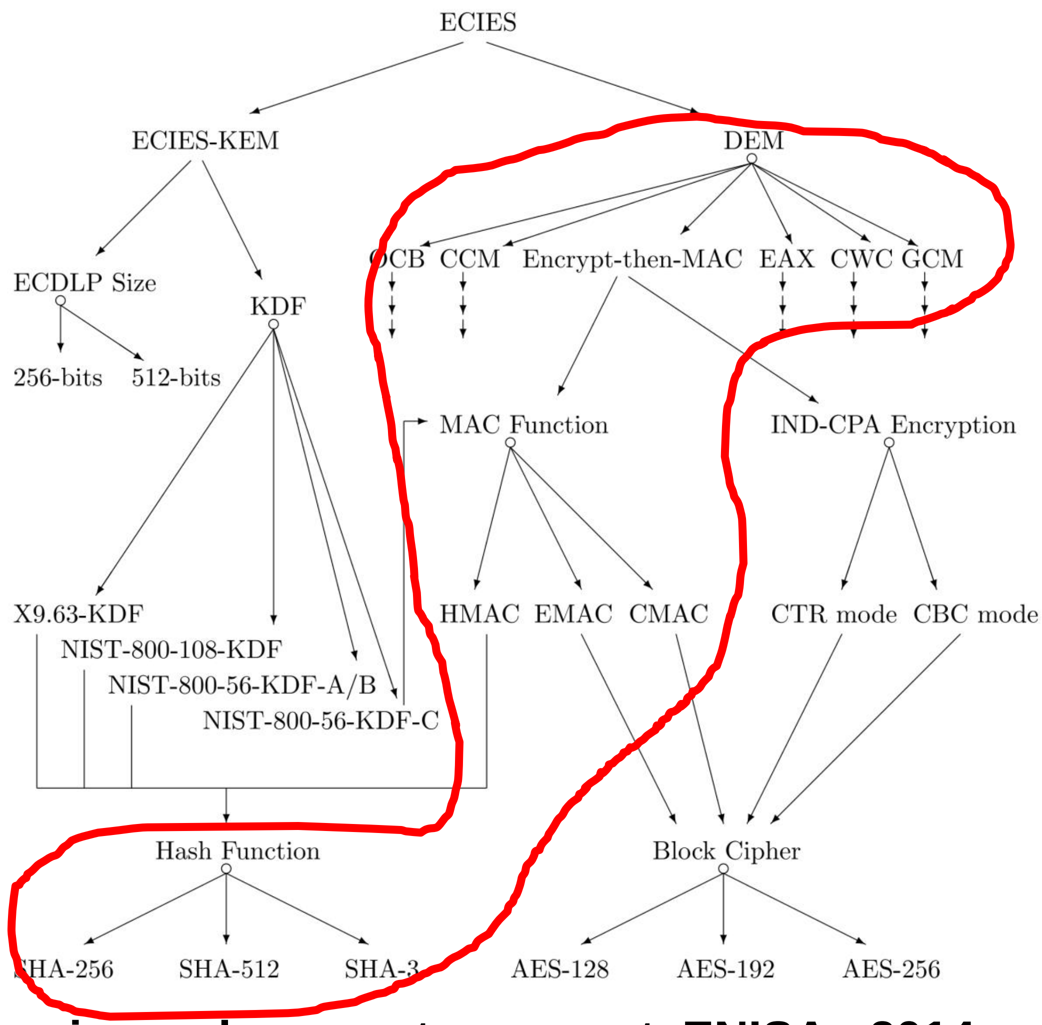
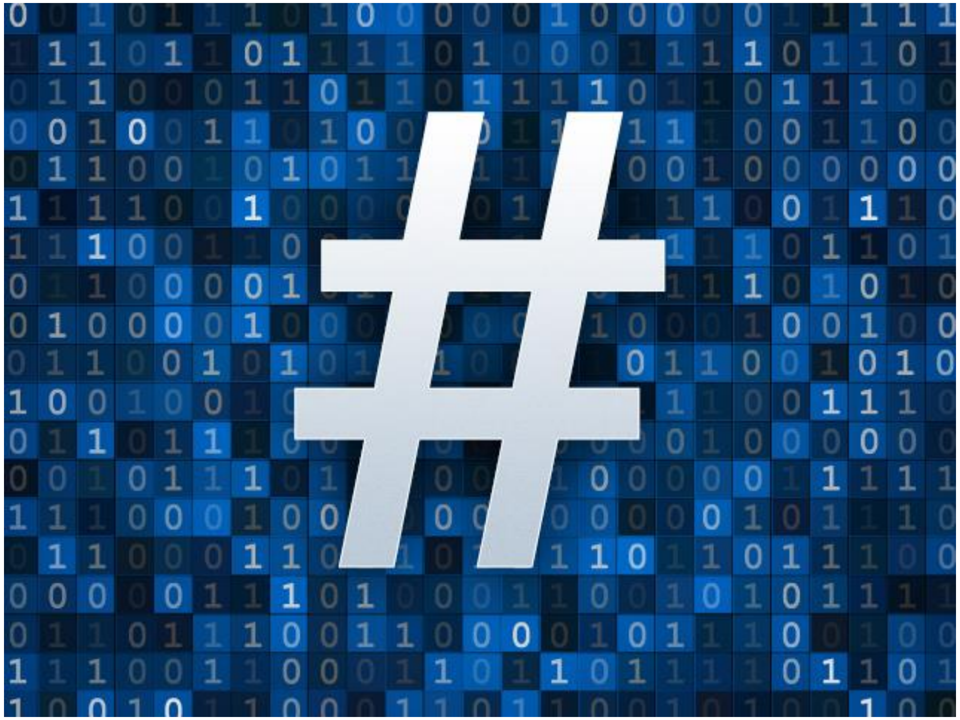
Cryptography
Lecture 3

Dr. Panagiotis Rizomiliotis

Overview



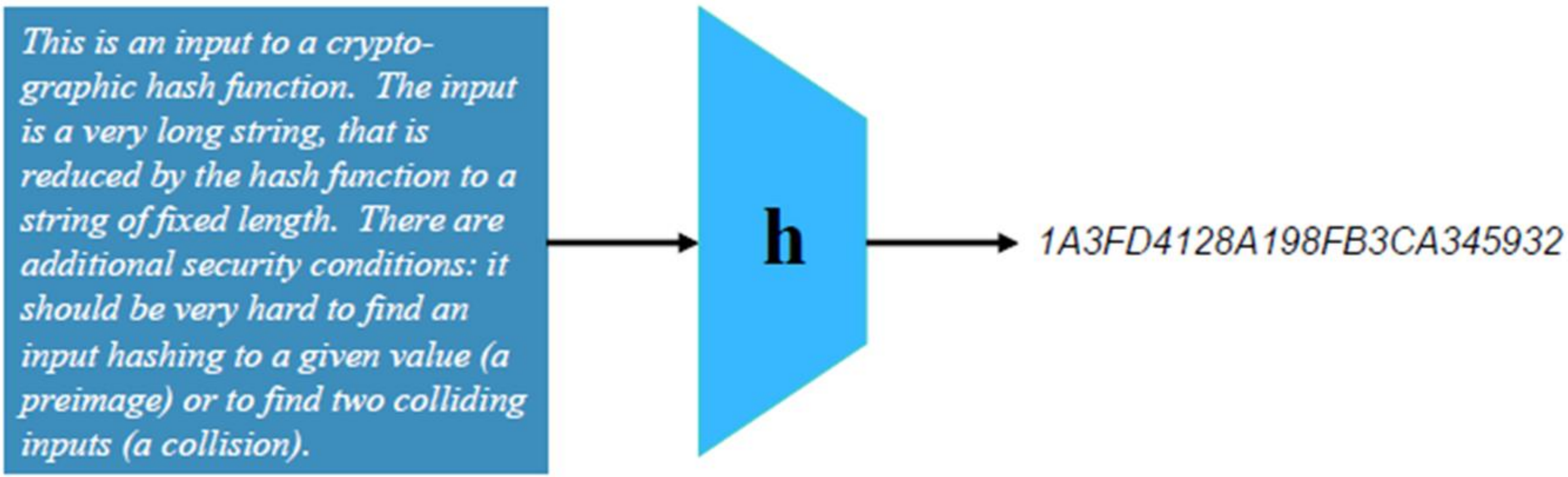
*** Algorithms, key size and parameters report. ENISA– 2014**



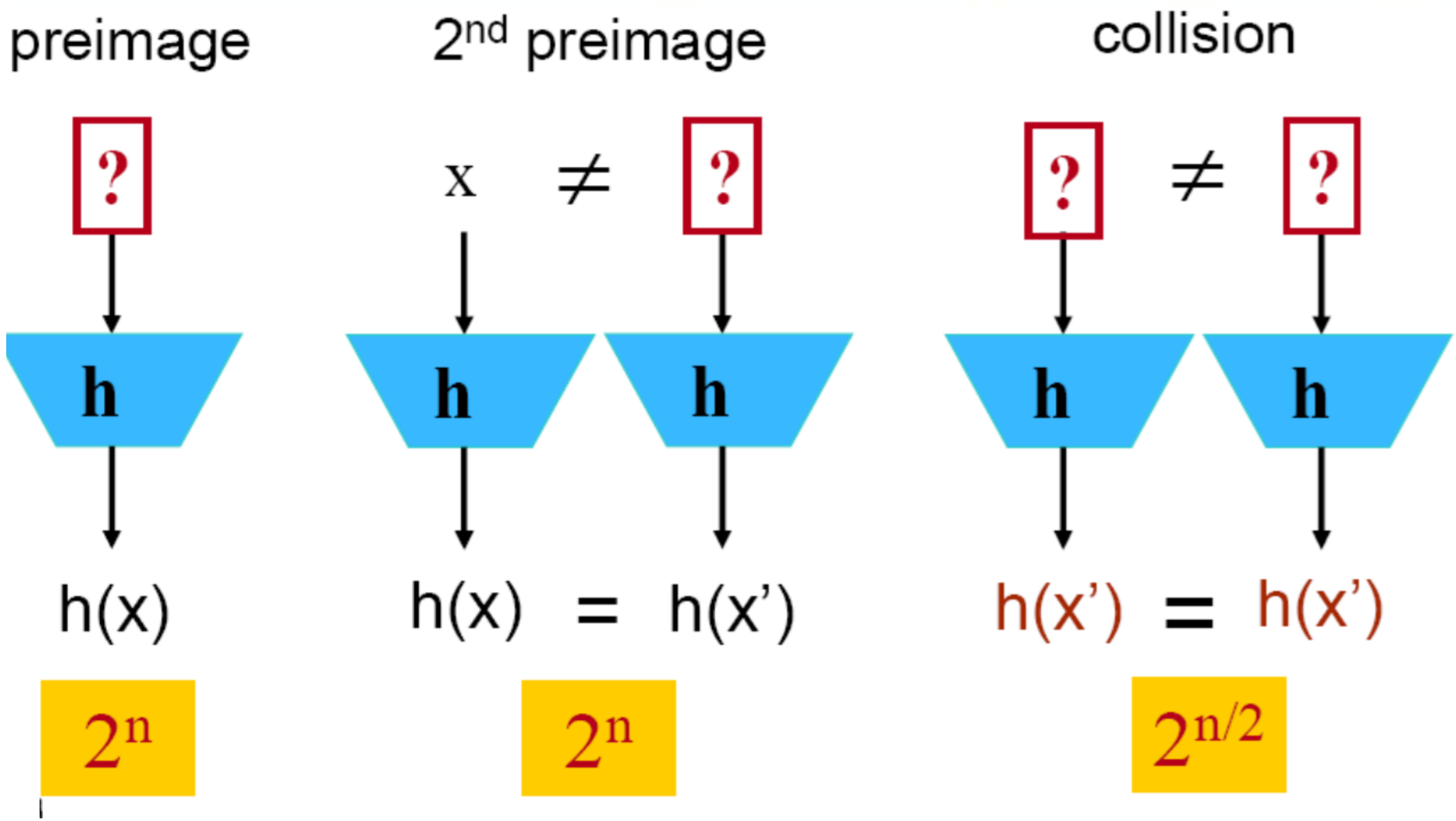
CRYPTOGRAPHIC HASH FUNCTIONS

Hash functions

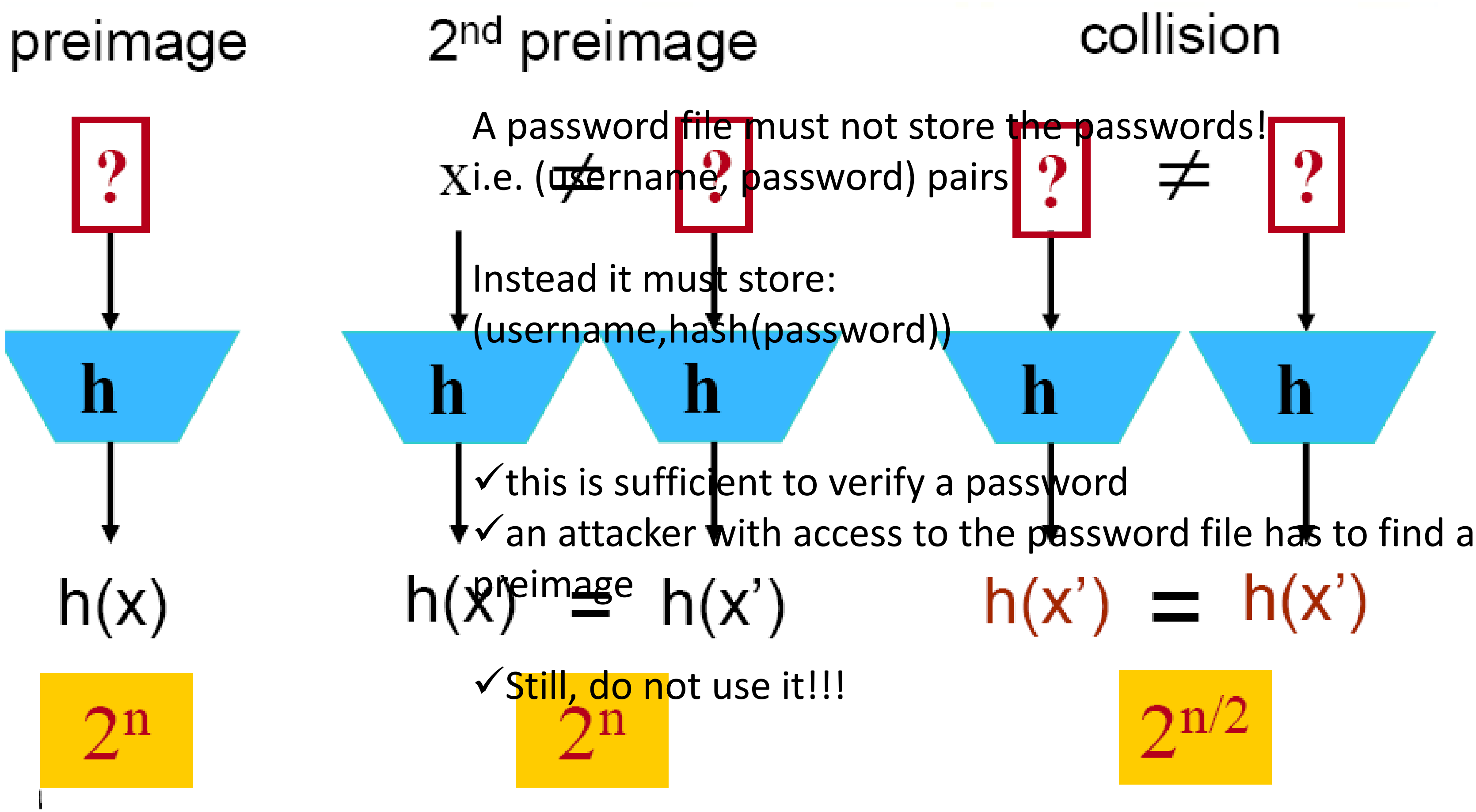
- ✓ no secret parameters
- ✓ input string x of arbitrary length ⇒ output h(x) of fixed length n (bits)
- ✓ computation "easy"
- ✓ One-way functions



Cryptographic properties



preimage



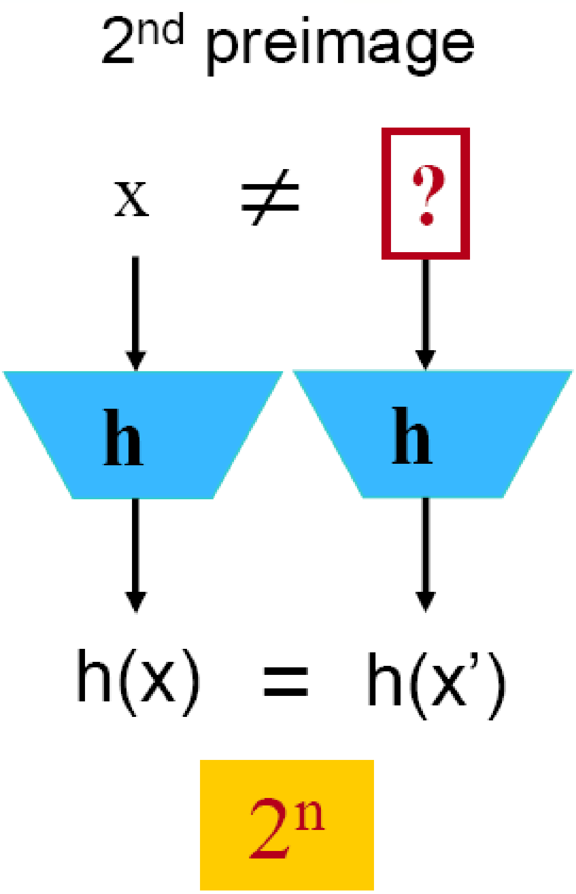
A password file must not store the passwords!
i.e. (username, password) pairs

Instead it must store:
(username,hash(password))

- ✓this is sufficient to verify a password
- ✓an attacker with access to the password file has to find a preimage

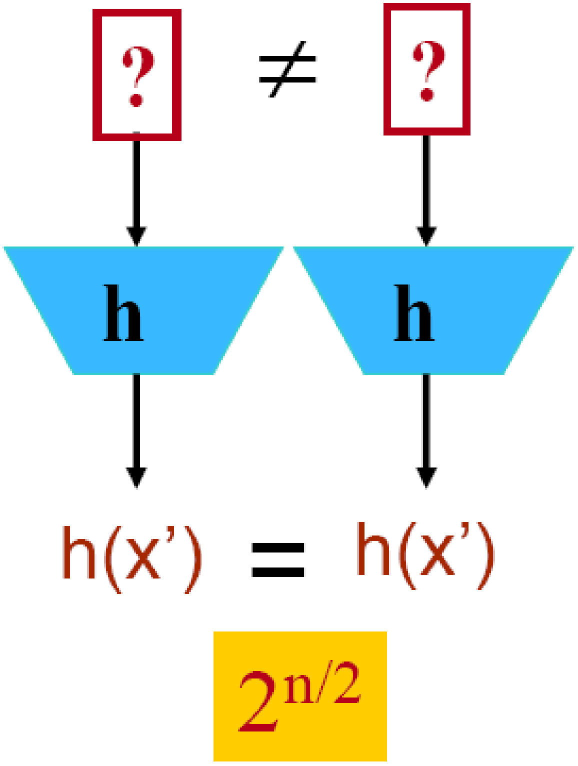
- ✓Still, do not use it!!!

Second preimage



- Can be used to protect the integrity of data x
- A secure channel is needed to send h(x) to the verifier.
- The attacker wants to modify x and remain undetected
- The attack is successful if the attacker can find a second preimage of x

collision



The hacker prepares two versions of a software Let

1. x be the correct code
2. x' contain a backdoor that gives hacker access to a machine

The hacker submits x for inspection to Bob

If Bob is satisfied, he digitally signs h(x) with his private key
The hacker distributes x';
The users verify the signature with Bob's public key
This signature works for x and for x',
since h(x) = h(x')!

Birthday paradox

- the birthday problem or birthday paradox concerns the probability that, in a set of *n* randomly chosen people, some pair of them will have the same birthday.

Example: lets assume that we have a group of 23 people.

$$\binom{23}{2} = \frac{23!}{21!2!} = 253\, pairs$$

We can show that the birthday paradox is larger than 50%!

Birthday paradox

$$p(n) = 1 - \frac{n!\binom{365}{n}}{365^n}$$

n	$p(n)$
10	11.7%
20	41.1%
23	50.7%
30	70.6%
50	97.0%
57	99.0%
100	99.99997%
200	99.9999999999999999999999999998%
300	$(100 - (6\times10^{-80}))\%$
350	$(100 - (3\times10^{-129}))\%$
365	$(100 - (1.45\times10^{-155}))\%$
367	100%

Birthday Attack

- A birthday attack is a name used to refer to a class of brute-force attacks. More precisely,

"If some function, when supplied with a random input, returns one of |k| equally-likely values, then by repeatedly evaluating the function for different inputs, we expect to obtain the same output after about $1.2|k|^{1/2}$. "

- Example: for the birthday paradox, we have |k|=365.

Brute force

- multiple target second preimage (1 out of many):
- – if one can attack 2^t simultaneous targets, the effort to find a single preimage is 2^{n-t}
- multiple target second preimage (many out of many):
  - time-memory trade-off with $\Theta(2^n)$ precomputation and storage $\Theta(2^{2n/3})$ time per (2nd) preimage: $\Theta(2^{2n/3})$ [Hellman'80]
- answer: randomize hash function with a parameter S

(salt, key, spice,…)

Brute force attacks in practice

- (2nd) preimage search
  - n = 128: 23 B$ for 1 year if one can attack **240 targets in** parallel
- parallel collision search: small memory using cycle finding algorithms (distinguished points)

  – n = 128: 1 M$ for 8 hours (or 1 year on 100K PCs)

  – n = 160: 90 M$ for 1 year

  – need 256-bit result for long term security (30 years or more)

Quantum era

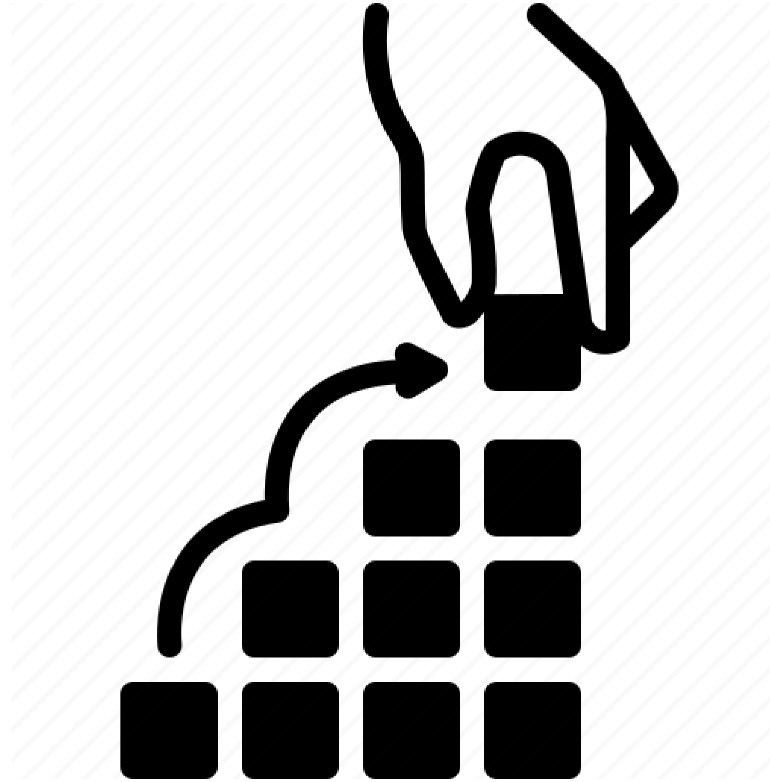
- in principle exponential parallelism
  - inverting a one-way function: 2^n reduced to $2^{n/2}$ [Grover'96]
- collision search:

– $2^{n/3}$ computation + hardware [Brassard-Hoyer-Tapp'98]

– [Bernstein'09] classical collision search requires $2^{n/4}$ computation and hardware (= standard cost of $2^{n/2}$)

Properties in practice

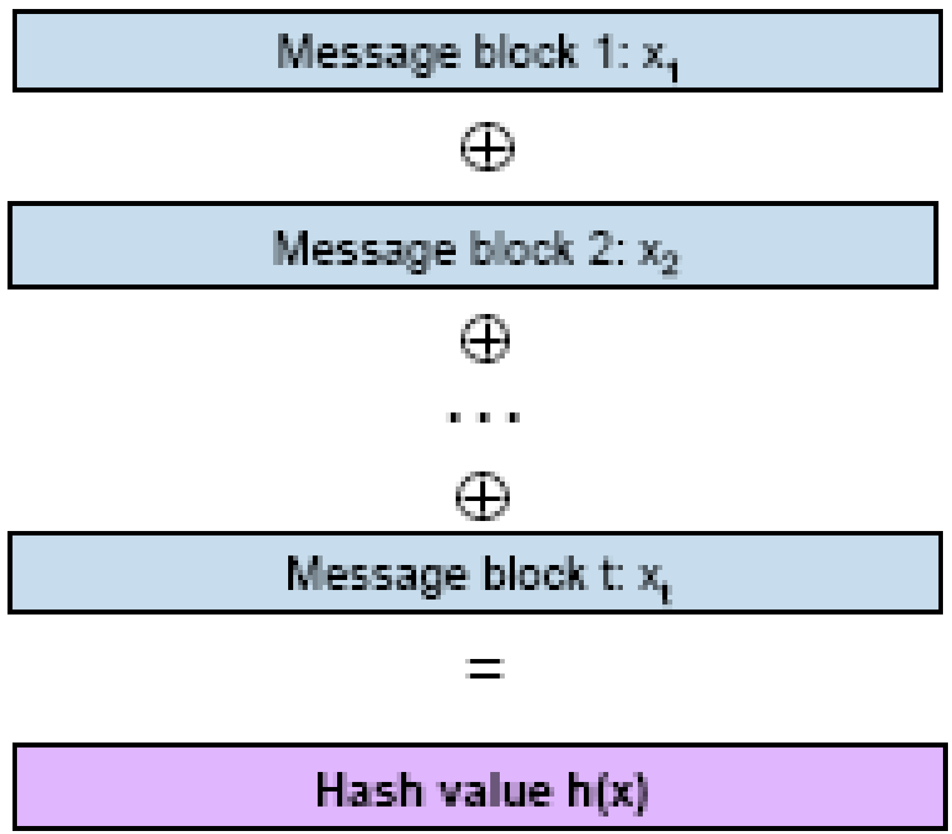
- collision resistance is not always necessary
- other properties are needed:
  - PRF: pseudo-randomness if keyed (with secret key)
  - PRO: pseudo-random oracle property (formalization of security properties when there is no key)
  - near-collision resistance
  - partial preimage resistance (most of input known)
  - multiplication freeness
- how to formalize these requirements and the relation between them?



BASIC CONSTRUCTIONS

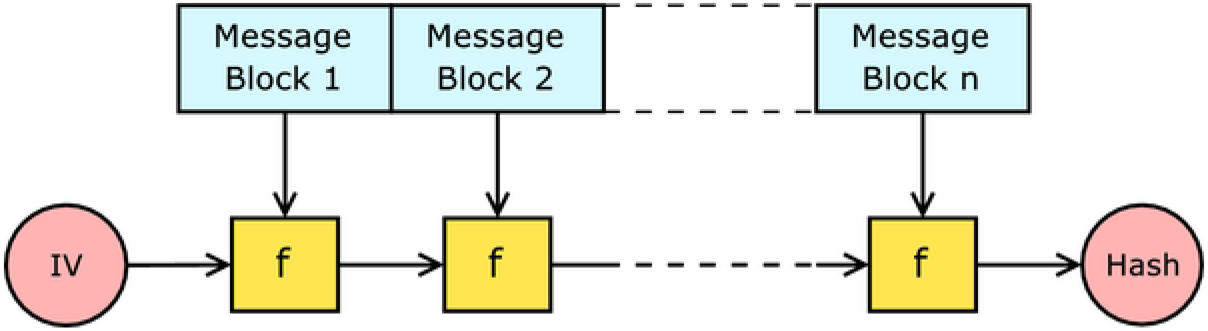
A simple approach

Divide the message into t blocks x_i of n bits each



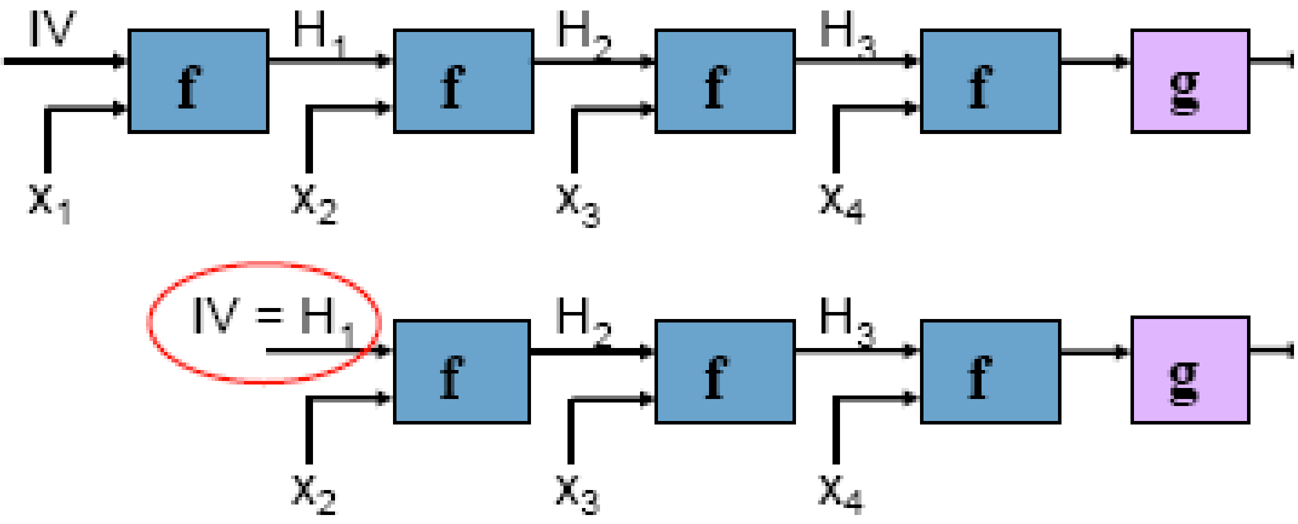
Merkle–Damgård construction

- f is a compression function
- How to choose the function
- - ad hoc
- - based on a block cipher



Iterated structure -attack

- iterating f can degrade its security
  - trivial example: 2nd preimage



Merkle-Damgard strengthening

Algorithm MD-strengthening

Before hashing a message $x = x_1 x_2 \ldots x_t$ (where x_i is a block of bitlength r appropriate for the relevant compression function) of bitlength b, append a final length-block, x_{t+1}, containing the (say) right-justified binary representation of b. (This presumes $b < 2^r$.)

Security relation between f and h

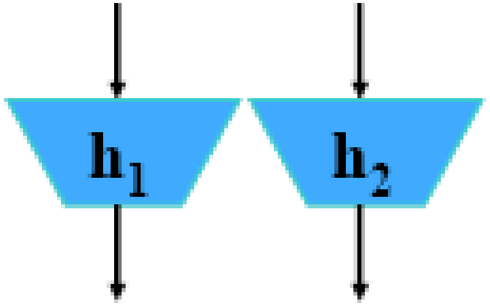
- solution: Merkle-Damgård (MD) strengthening
  - fix IV, use unambiguous padding and insert length at the end

- f is collision resistant $\Rightarrow$ h is collision resistant
  [Merkle'89-Damgård'89]
- f is ideally 2nd preimage resistant $\overset{?}{\Leftrightarrow}$ h is ideally 2nd preimage resistant [Lai-Massey'92]

- property preservation has been a heavily studied topic since 2005

How (NOT) to strengthen a hash function?[Joux'04]

- answer: concatenation
- h_1 (n1-bit result) and h_2 (n2-bit result)



$g(x) = h_1(x) \,||\, h_2(x)$

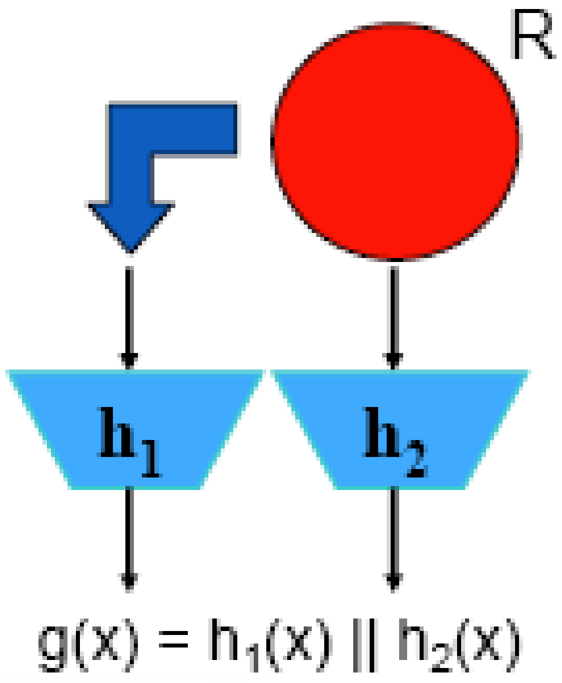
- intuition: the strength of g against collision/(2nd) preimage attacks is the product of the strength of h_1 and h_2
  - if both are "independent"
- but…. for iterated hash functions only the strongest function matters

Multi-collisions [Joux '04]

- finding multi-collisions for an iterated hash function is not much harder than finding a single collision (if the size of the internal memory is n bits)

- algorithm
  - generate R = $2^{n1/2}$-fold multi-collision for h_2
  - in R: search by brute force for h_1

- time: n1. $2^{n2/2} + 2^{n1/2}$ << $2^{(n1 + n2)/2}$



Multi-collisions [Joux '04]

consider h_1 (n1-bit result) and h_2 (n2-bit result), with $n1 \geq n2$.

concatenation of 2 iterated hash functions ($g(x)= h_1(x) \,||\, h_2(x)$) is as most as strong as the strongest of the two (even if both are independent)

- cost of collision attack against g at most

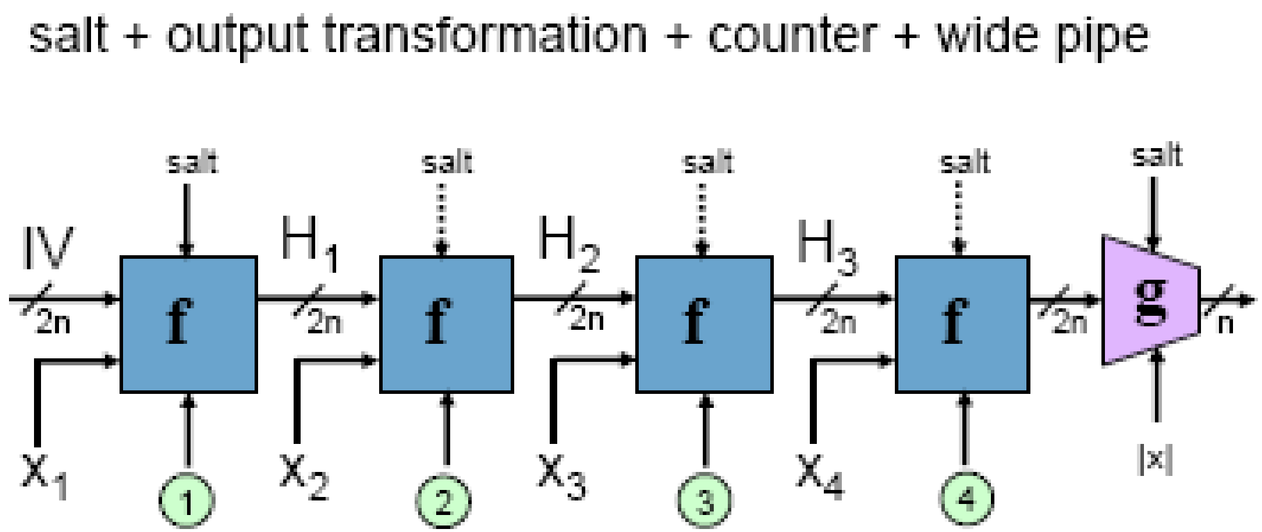
  $$n1 \,.\, 2^{n2/2} + 2^{n1/2} << 2^{(n1 + n2)/2}$$

- cost of (2nd) preimage attack against g at most

  $$n1 \,.\, 2^{n2/2} + 2^{n1} + 2^{n2} << 2^{n1 + n2}$$

- if either of the functions is weak, the attacks may work better

Improving MD iteration



Improving MD iteration

- degradation with use: salting (family of functions, randomization)

– or should a salt be part of the input?

- PRO: strong output transformation g

– also solves length extension

- long message 2nd preimage: preclude fix points

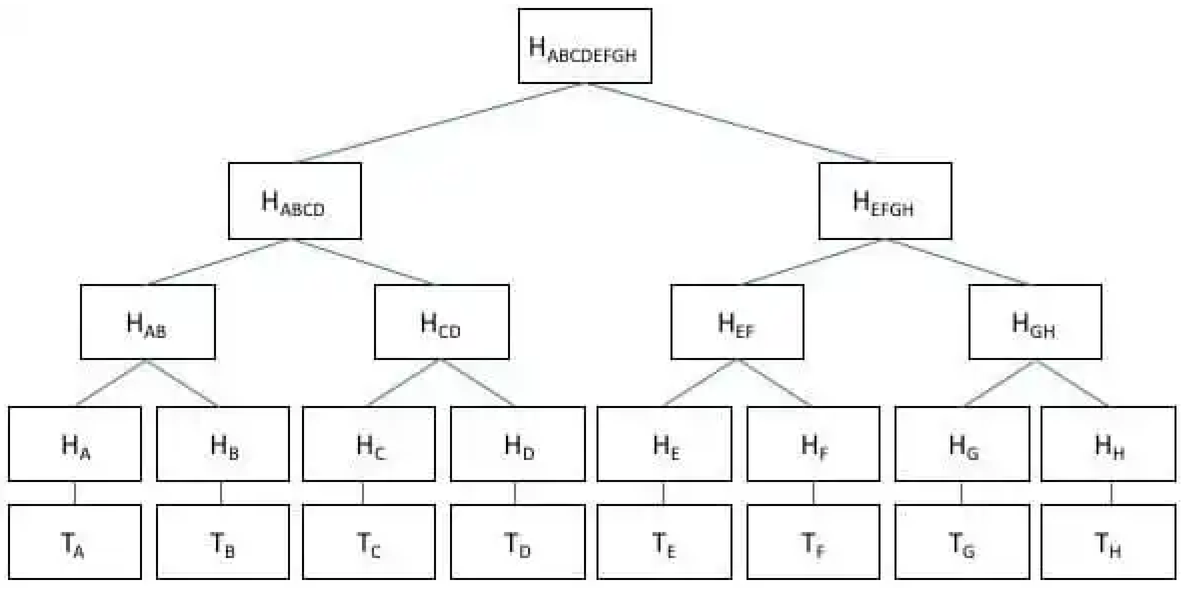
– counter f → fi [Biham-Dunkelman'07]

- multi-collisions, herding: avoid breakdown at $2^{n/2}$ with larger internal memory: known as wide pipe

– e.g., extended MD4, RIPEMD, [Lucks'05]

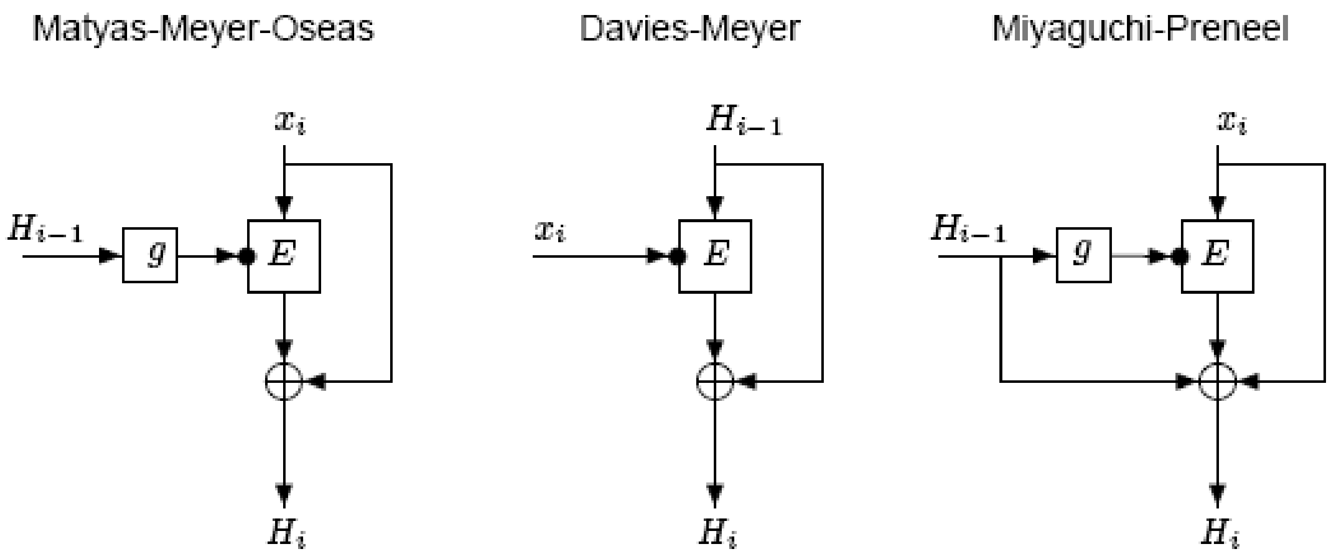
Merkle Tree

- Hash trees allow efficient and secure verification of the contents of large data structures



COMPRESSION FUNCTIONS

Block cipher based

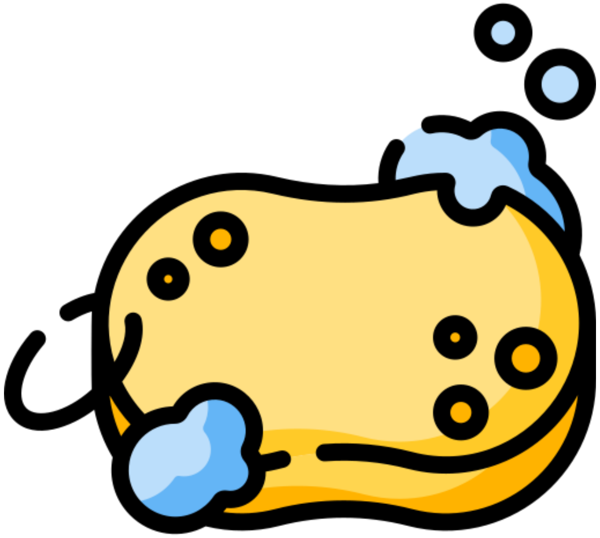


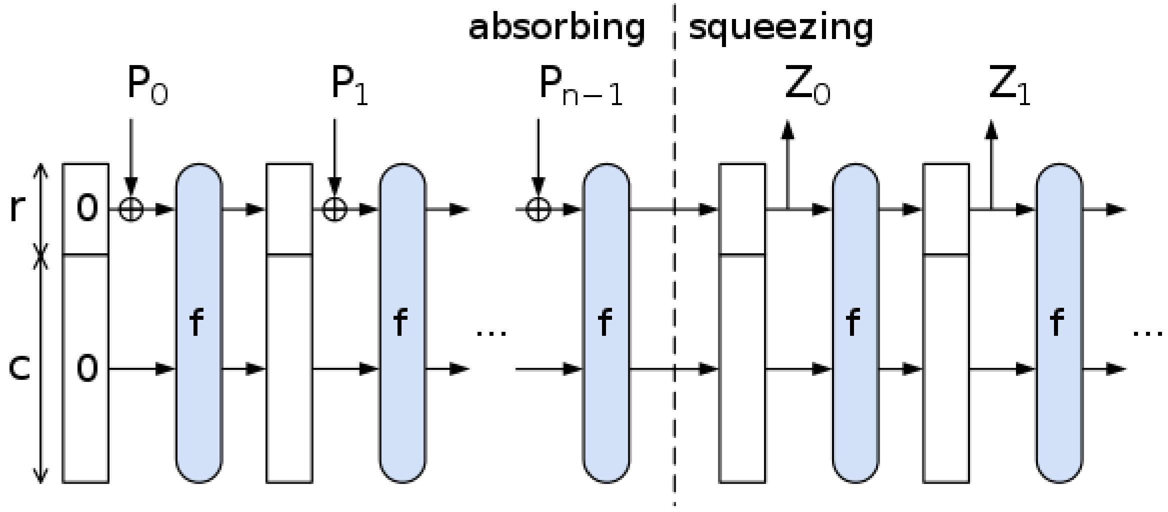
Security Analysis

- The security of the Davies–Meyer construction in the Ideal Cipher Model
- For Matyas–Meyer–Oseas construction there is second preimage attack
- For Miyaguchi–Preneel construction there is second preimage attack

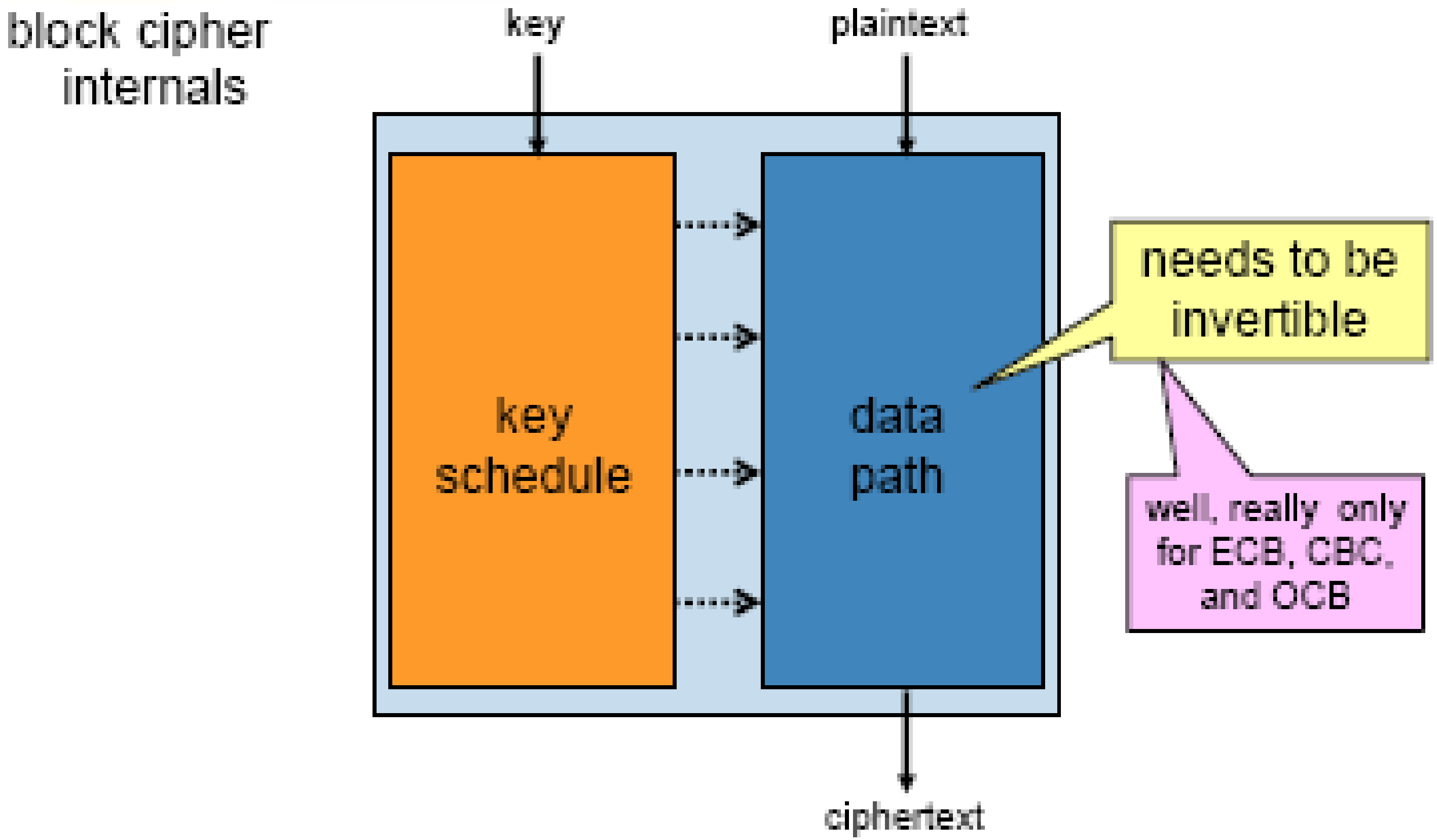
Non block cipher based

- Sponge construction!

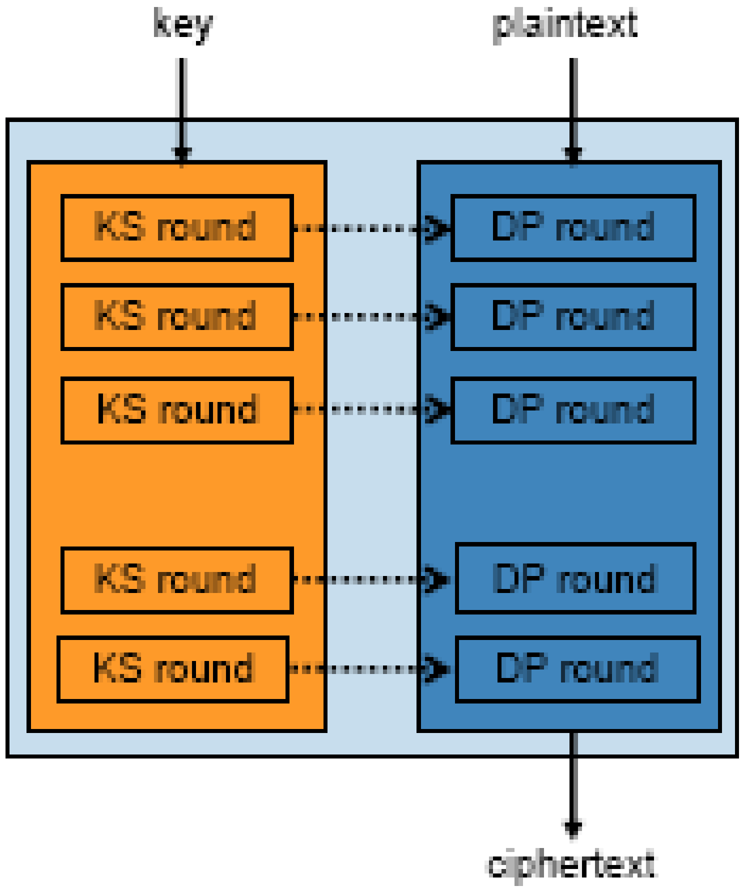




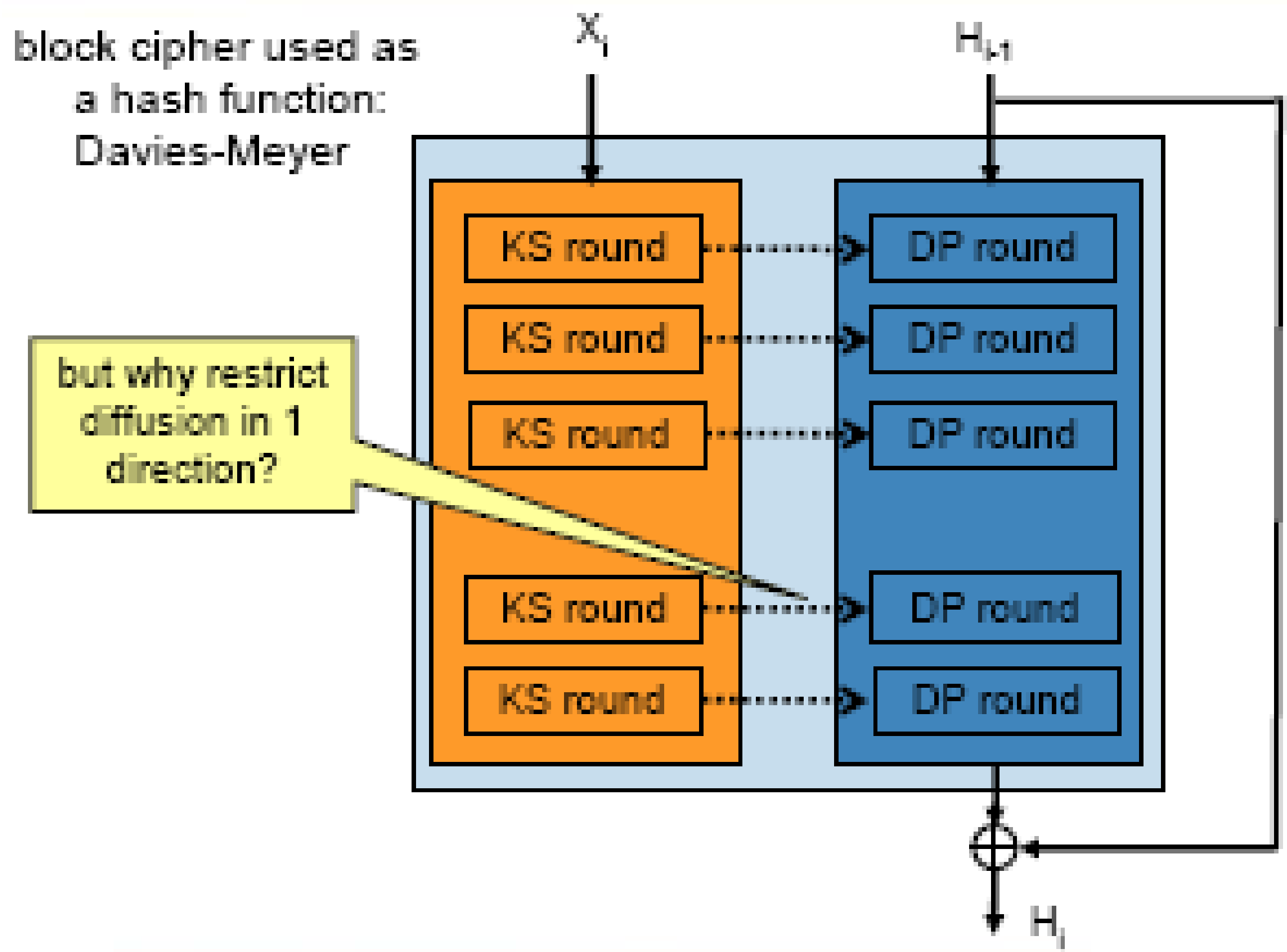
Motivation for use of a larger permutation



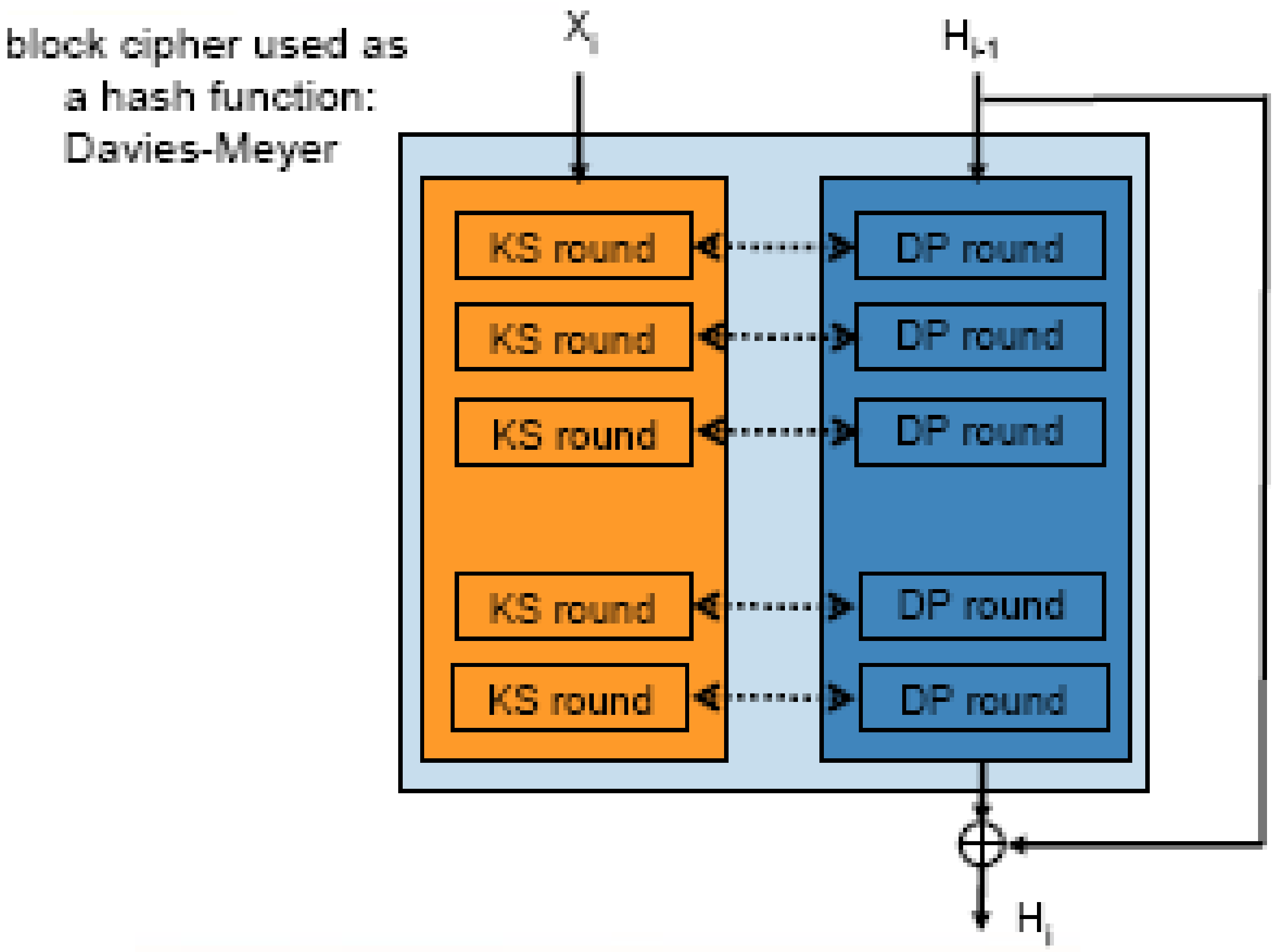
Motivation for use of a larger permutation



Motivation for use of a larger permutation



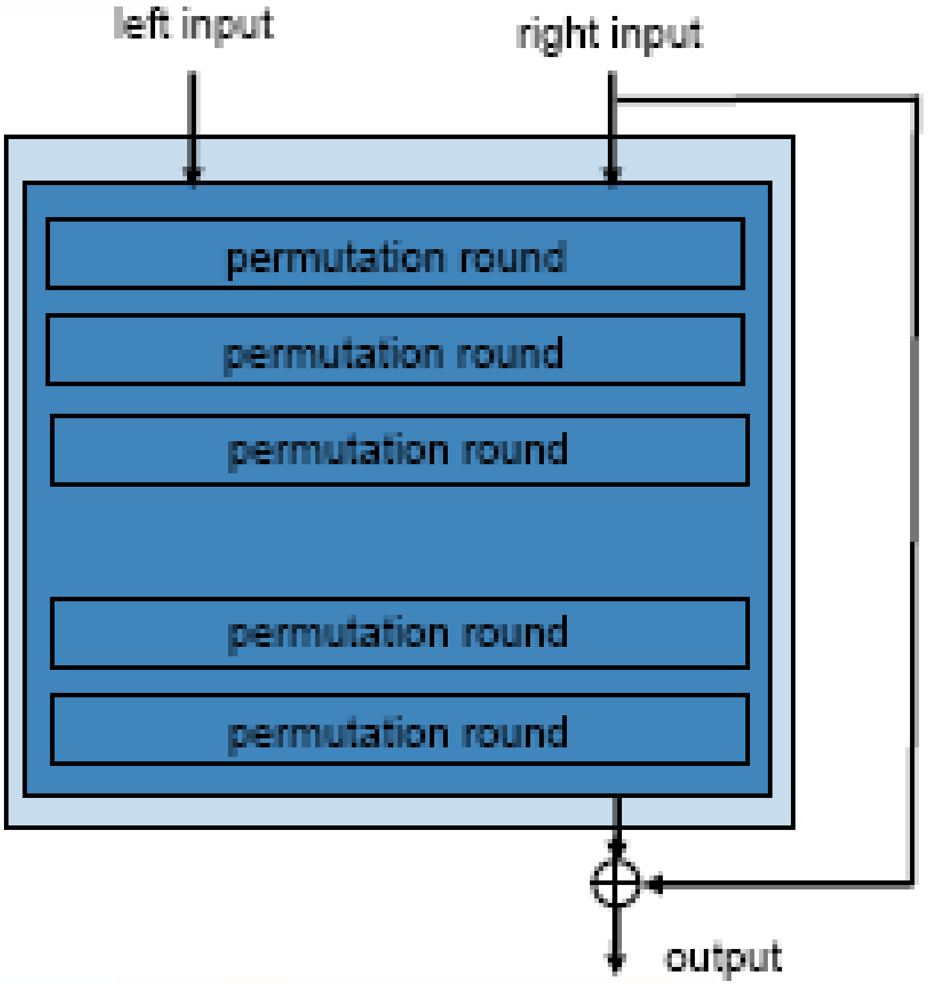
Motivation for use of a larger permutation



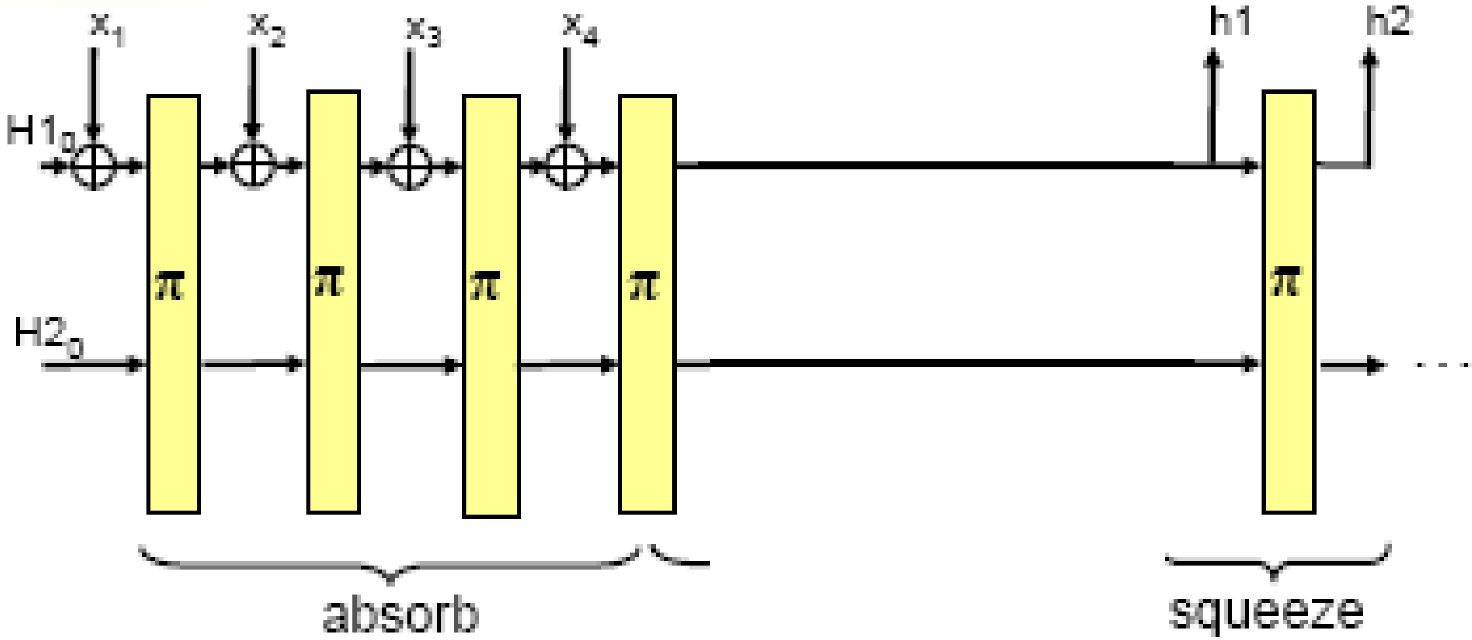
Motivation for use of a larger permutation

block cipher used as a hash function: Davies-Meyer

then one can as well have a permutation



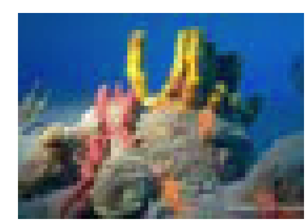
Motivation for use of a larger permutation



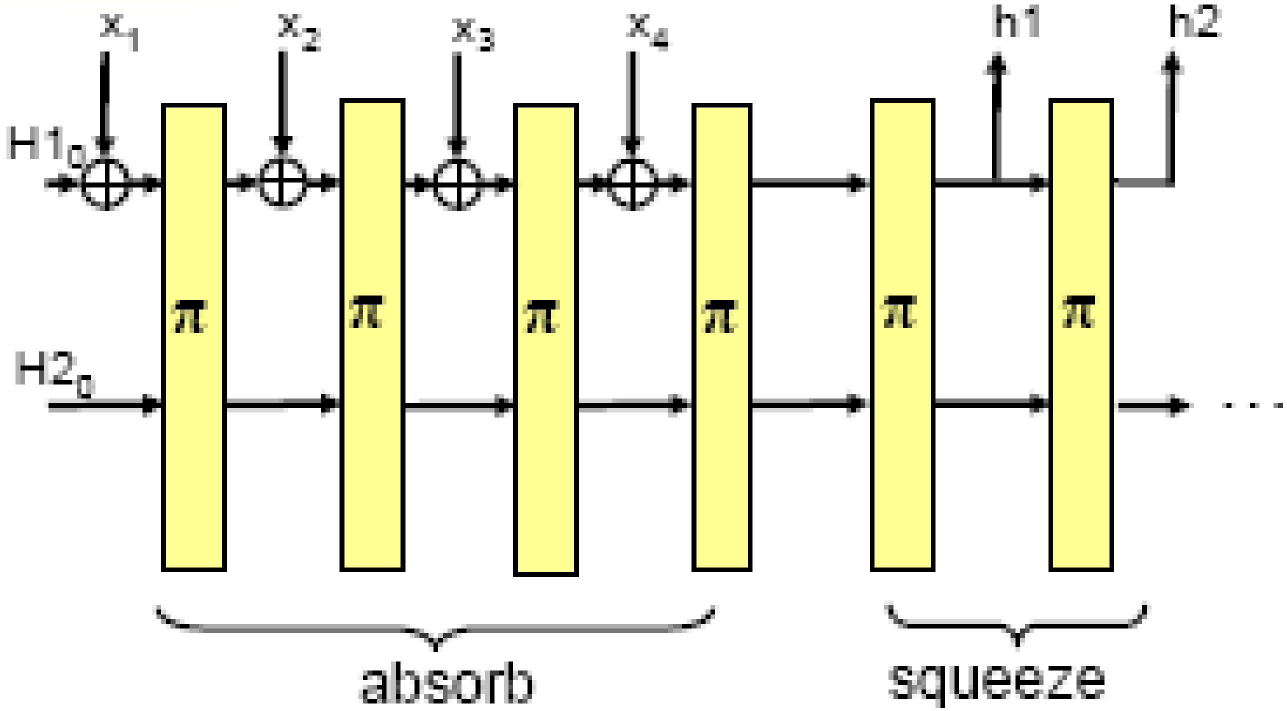
example: Keccak (no buffer)

generalization called Parazoa

JH, Cubehash, Fuge, Grindahl, Hamsi, Luffa



Motivation for use of a larger permutation



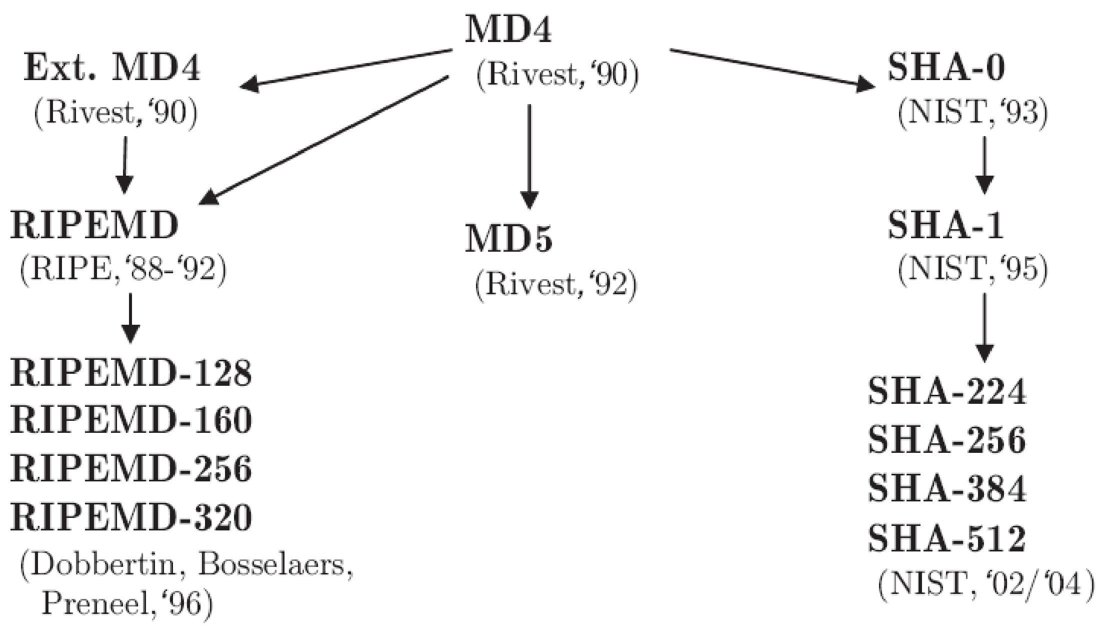
If H1 has r bits (rate) and H2 has c bits (capacity) and the permutation π is "ideal", then a sponge function has security $O(2^c)$ against (2nd) preimage attacks and $O(2^{c/2})$ against collision attacks

Iteration modes and compression functions

- security of simple modes well understood
  - powerful tools available
- analysis of slightly more complex schemes very difficult
  - which properties are meaningful?
  - which properties are preserved?
  - MD versus sponge is still open debate

CONSTRUCTIONS

MD4 family



MD4
(Rivest,'90)
Ext. MD4
(Rivest,'90)
SHA-0
(NIST,'93)
RIPEMD
(RIPE,'88-'92)
MD5
(Rivest,'92)
SHA-1
(NIST,'95)
RIPEMD-128
RIPEMD-160
RIPEMD-256
RIPEMD-320
(Dobbertin, Bosselaers,
Preneel,'96)
SHA-224
SHA-256
SHA-384
SHA-512
(NIST, '02/'04)

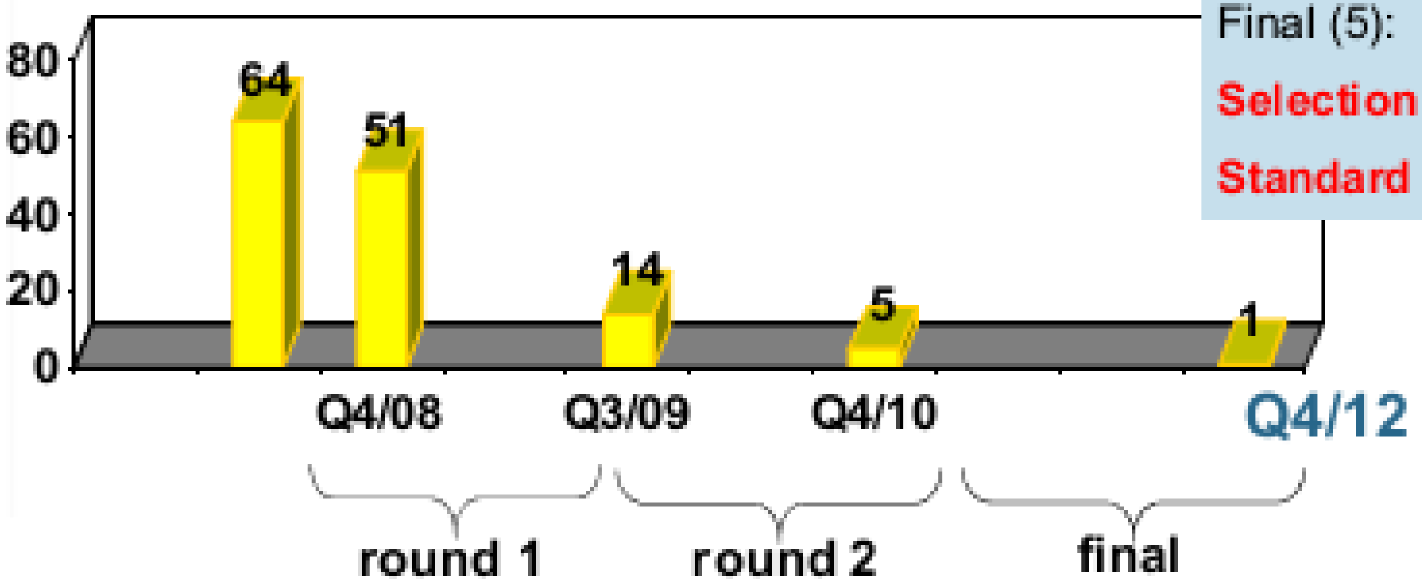
timeline

- 1990: MD4 by Ron Rivest
- 1991: MD5 by Ron Rivest (RFC 1321, 1992)
- 1992: RIPEMD by H. Dobbertin, A. Bosselaers and B. Preneel
- 1993: SHA-0 by U.S. Government (FIPS PUB 180)
- 1995: SHA-1 by U.S. Government (FIPS PUB 180-1)
- 2000: Whirlpool by V. Rijmen and P. Barreto
- 2001: SHA-2 by U.S. Government (FIPS PUB 180-2)
- 2005: First attacks against SHA-1
- 2015: SHA-3 by the Keccak team (FIPS 202)
- 2017: February 2017, CWI Amsterdam and Google announced they had performed a collision attack against SHA-1

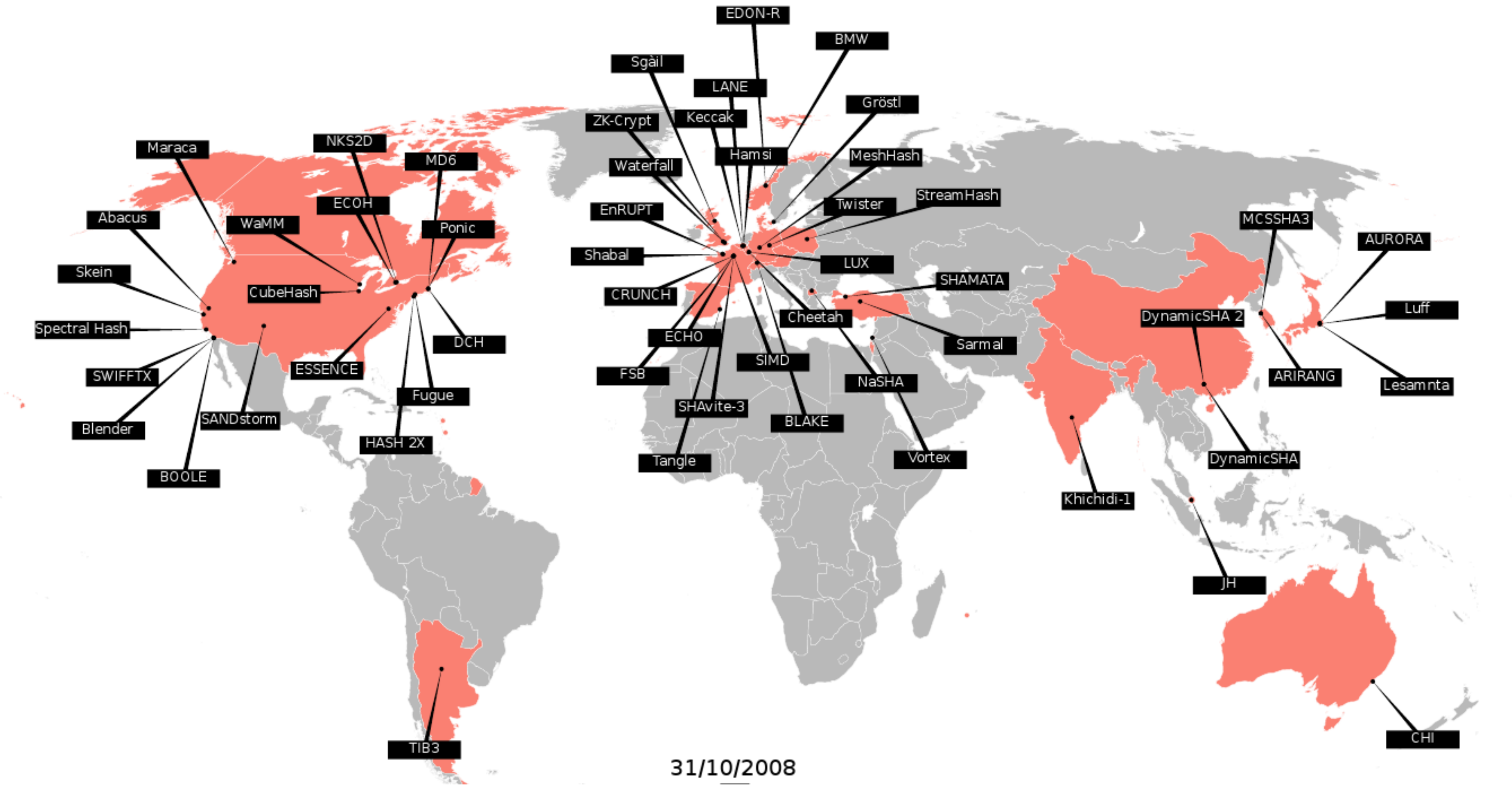
SHA-3 competition

- SHA-3: 224, 256, 384, and 512-bit message digests
- (similar to SHA-2)

Call:	02/11/07
Deadline (64):	31/10/08
Round 1 (51):	09/12/08
Round 2 (14):	24/7/09
Final (5):	10/12/10
Selection:	**02/10/12**
Standard	**05/08/15**



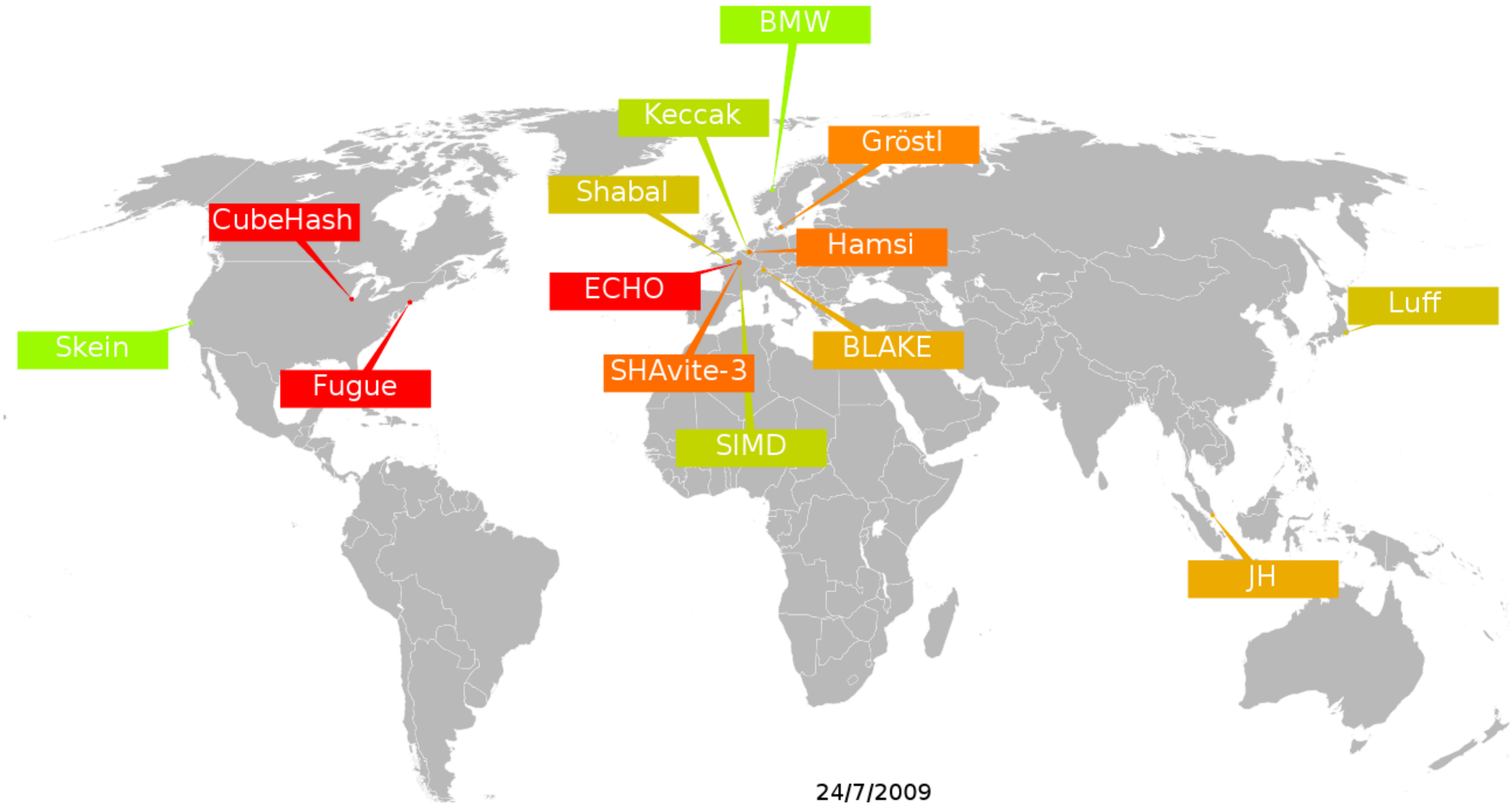
The candidates



From B. Preneel slides

Slides credits: Christophe De Cannière

Round-2 candidates



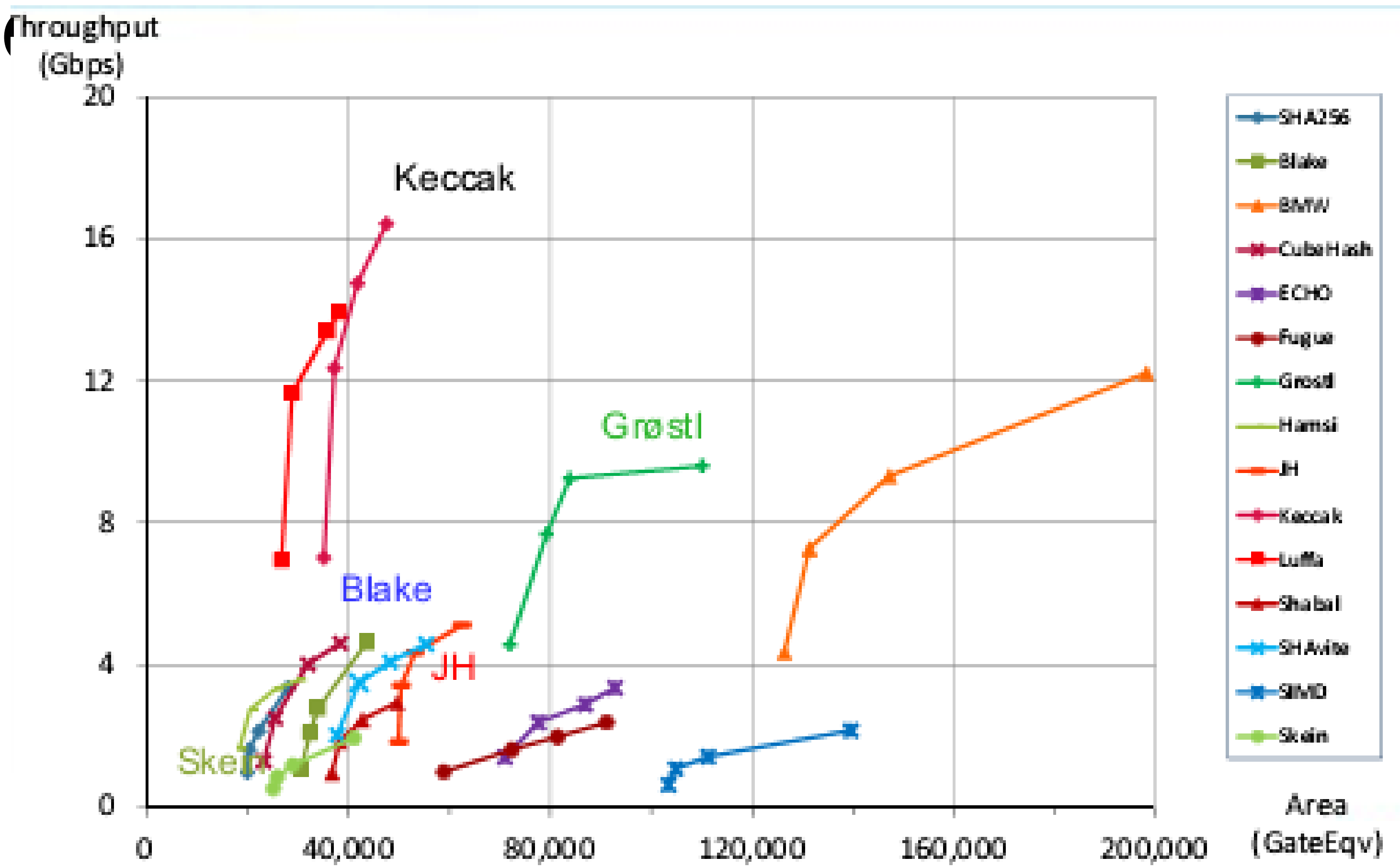
SHA-3 finalists

- ✓ BLAKE (Aumasson et al.)
- ✓ Grøstl (Knudsen et al.)
- ✓ JH (Hongjun Wu)
- ✓ Keccak (Keccak team, Daemen et al.)
- ✓ Skein (Schneier et al.)

  - Geography: 3 from Europe, 1 from Asia, 1 from America
  - Team members also AES finalist: 3

Hardware: post-place & route results ASIC 130nm

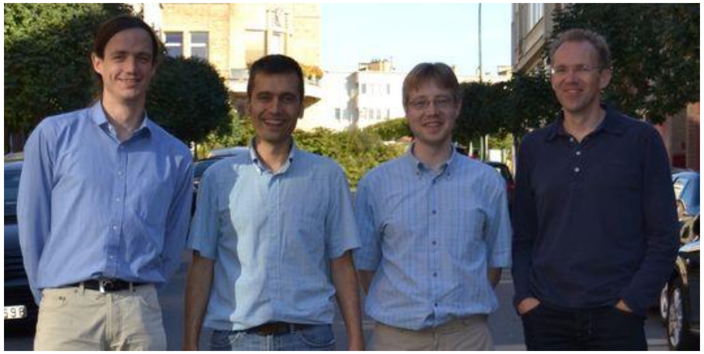
- [(



Keccak: FIPS 202 (published: 5 August 2015)

- append 2 extra bits for domain separation to allow
  - flexible output length (XOFs or eXtendable Output Functions)
  - tree structure (Sakura) allowed by additional encoding
- 6 versions
  - SHA3-224: n=224; c = 448; r = 1152 (72%)
  - SHA3-256: n=256; c = 512; r = 1088 (68%)
  - SHA3-384: n=384; c = 768; r = 832 (52%)
  - SHA3-512: n=512; c = 1024; r = 576 (36%)
  - SHAKE128: n=x; c = 256; r = 1344 (84%)
  - SHAKE256: n=x; c = 512; r = 1088 (68%)
- if result has n bits, H1 has r bits (rate), H2 has c bits (capacity) and the permutation π is "ideal":
  - collisions: min $(2^{c/2}, 2^{n/2})$
  - 2^{nd} preimage: min $(2^{c/2}, 2^{n})$
  - Preimage: min $(2^{c}, 2^{n})$

SHA3 Winner: Keccak



- ✓ Not an MD construction
- ✓ Based on a new design: sponge

- ✓ Design team: Guido Bertoni, Joan Daemen, Michaël Peeters, Gilles Van Assche
- ✓ FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and
- Extendable-Output Functions
- ✓ https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf

State of the art

Primitive	Output Length	Classification Legacy	Classification Future
SHA-2	256, 384, 512	✓	✓
SHA3	256,384,512	✓	✓
Whirlpool	512	✓	✓
SHA3	224	✓	✗
SHA-2	224	✓	✗
RIPEMD-160	160	✓	✗
SHA-1	160	✗	✗
MD-5	128	✗	✗
RIPEMD-128	128	✗	✗

Other hash functions

- BLAKE2
  – Since 2012
  – high efficiency that it offers on modern CPUs

- Whirlpool
  – Since 2000
  – designed by Vincent Rijmen and Paulo S. L. M. Barreto
  – 512 bits


Questions?