ECRYPT II

www.ecrypt.eu.org

Cryptographic Hash Functions: Theory and Practice

Bart Preneel

Katholieke Universiteit Leuven - COSIC

firstname.lastname@esat.kuleuven.be

X.509 Annex D

MDC-2

MD2, MD4, MD5

SHA-1

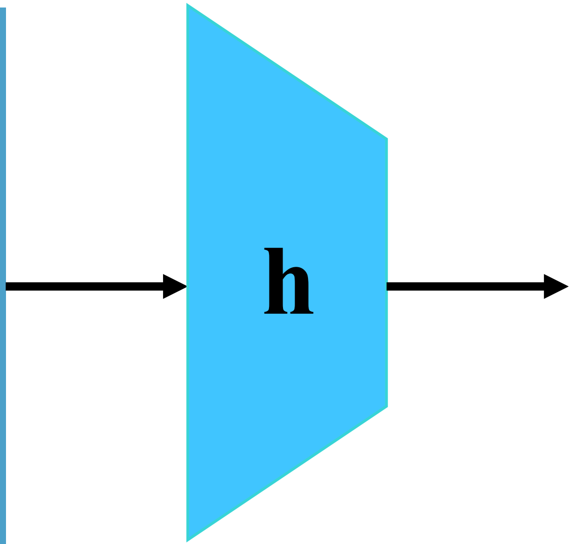
RIPEMD-160

SHA-256

SHA-512

SHA-3

This is an input to a crypto-graphic hash function. The input is a very long string, that is reduced by the hash function to a string of fixed length. There are additional security conditions: it should be very hard to find an input hashing to a given value (a preimage) or to find two colliding inputs (a collision).



1A3FD4128A198FB3CA345932

Applications

- short unique identifier to a string
  - digital signatures
  - data authentication
- one-way function of a string
  - protection of passwords
  - micro-payments
- confirmation of knowledge/commitment

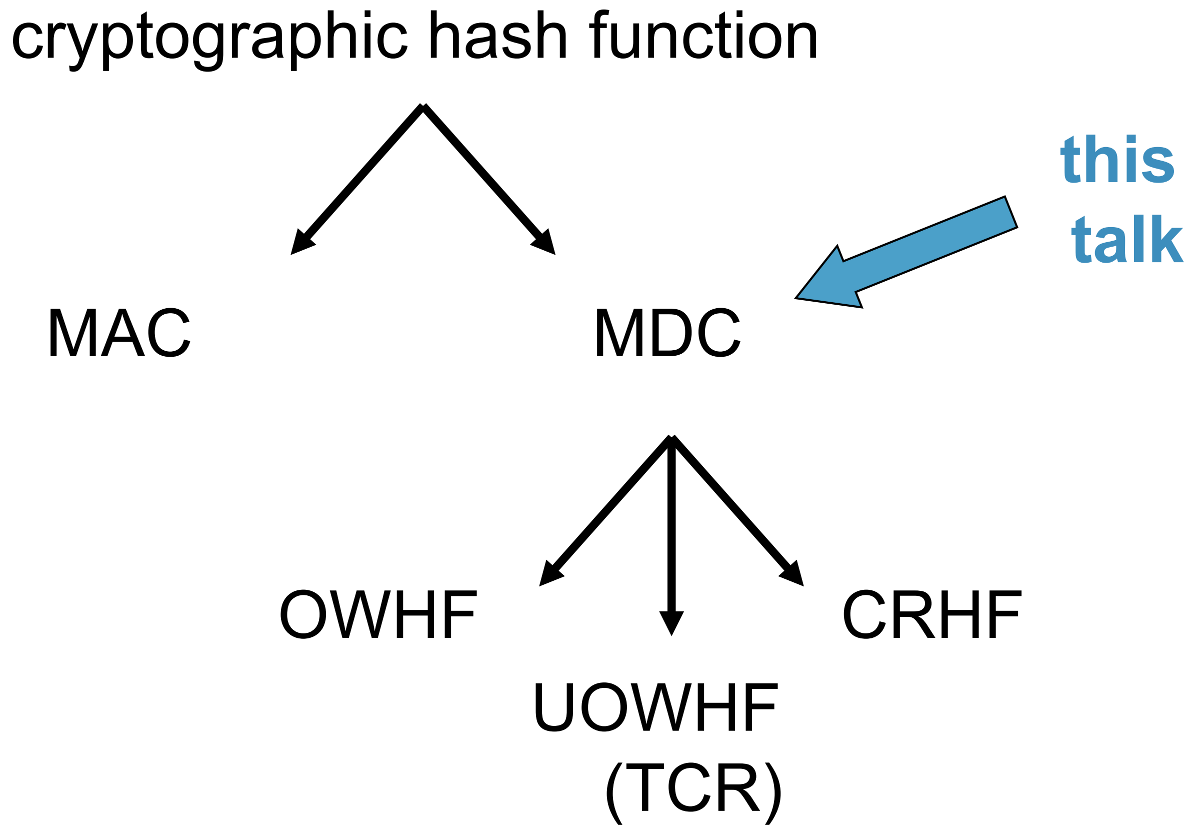
- pseudo-random string generation/key derivation
- entropy extraction
- construction of MAC algorithms, stream ciphers, block ciphers,…

2005: 800 uses of MD5 in Microsoft Windows

Agenda

- Definitions
- Iterations (modes)
- Compression functions
- SHA-{0,1,2}
- SHA-3 bits and bytes

Hash function flavours

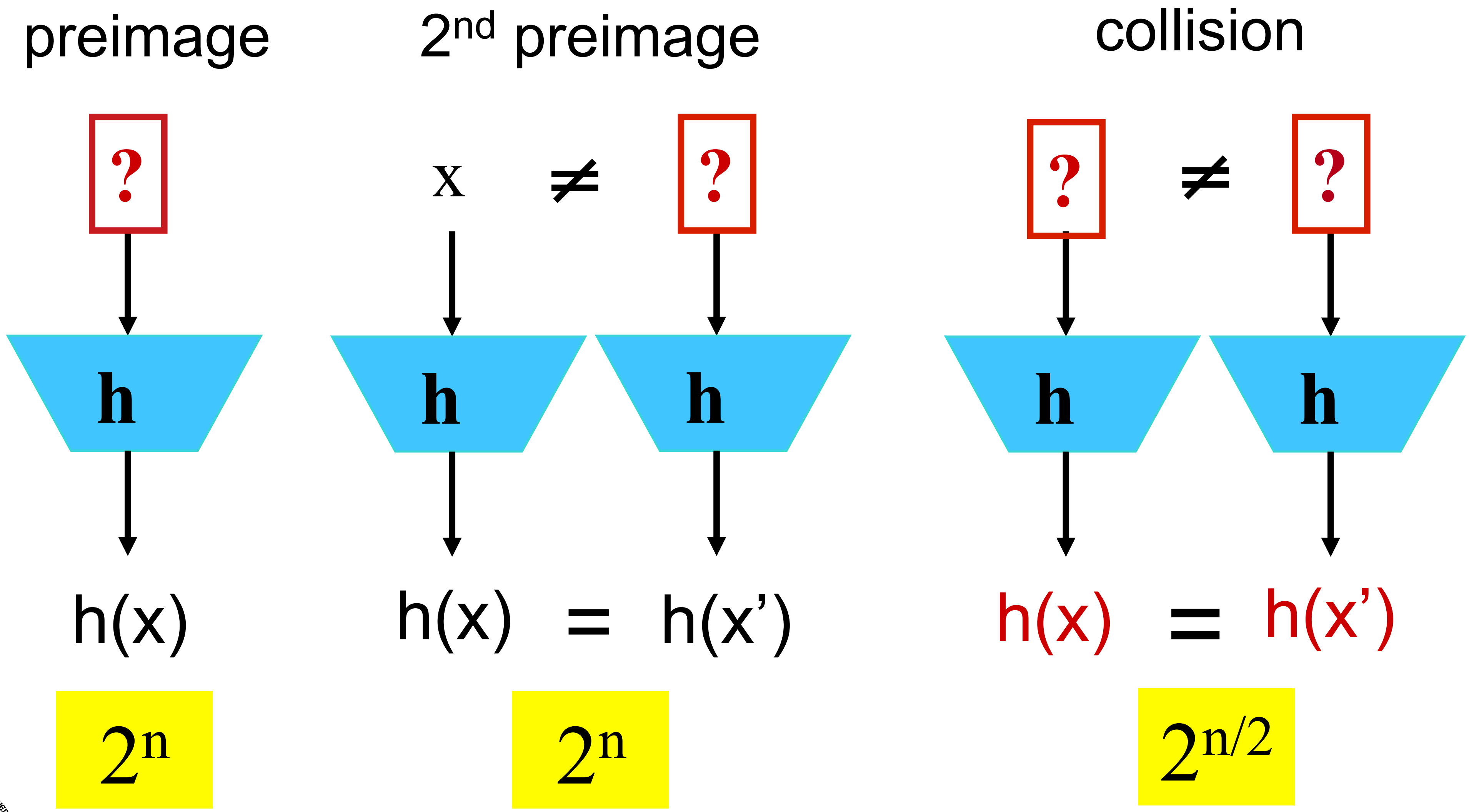


Informal definitions

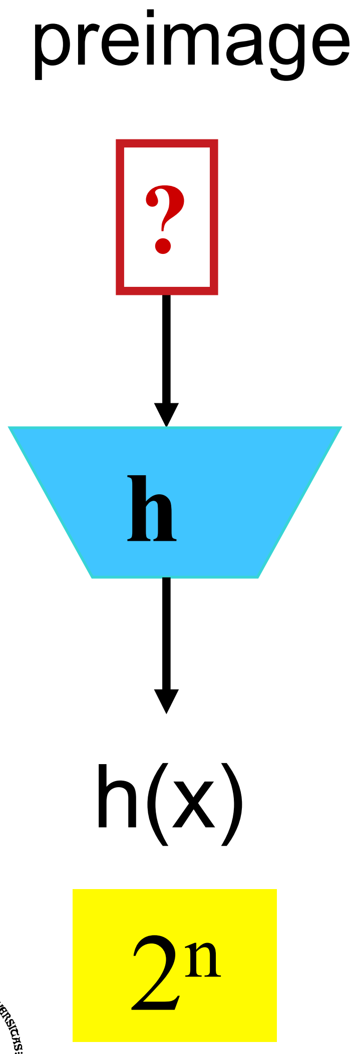
- no secret parameters
- input string *x* of arbitrary length ⇒ output h(x) of fixed bitlength *n*
- computation “easy”

- One Way Hash Function (OWHF)
  - preimage resistance
  - 2^{nd} preimage resistance
- Collision Resistant Hash Function (CRHF): OWHF +
  - collision resistant

Security requirements (n-bit result)



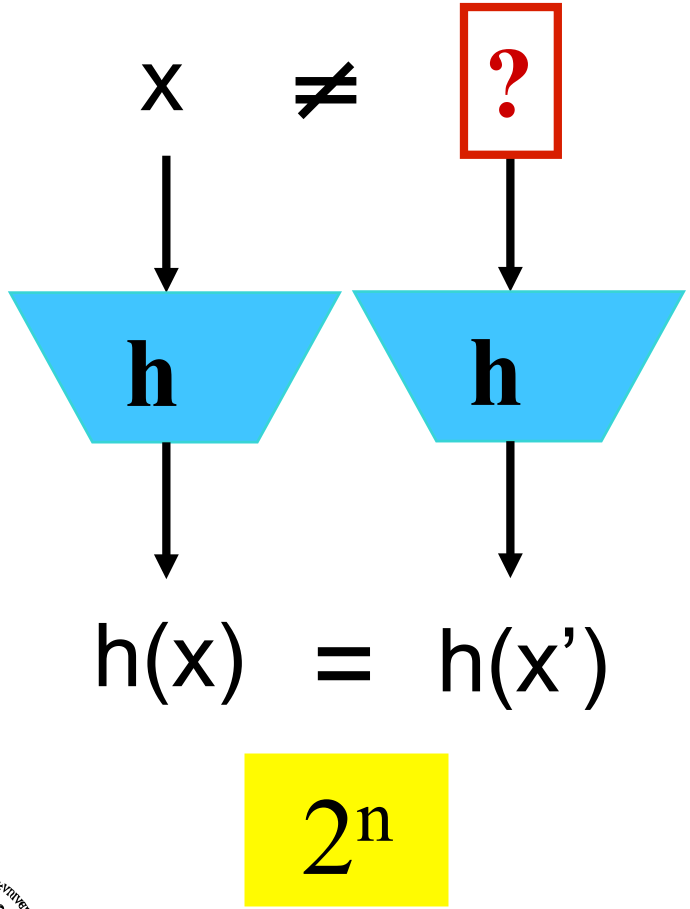
Preimage resistance

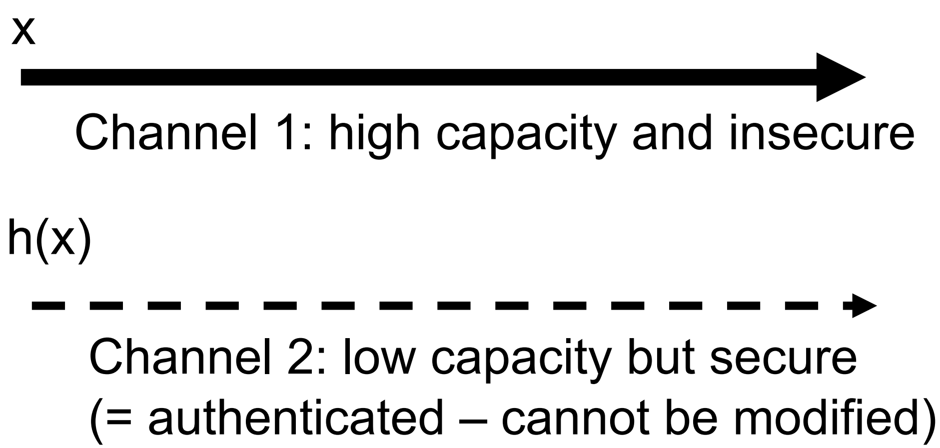


- in a password file, one does not store
  - (username, password)
- but
  - (username,hash(password))
- this is sufficient to verify a password
- an attacker with access to the password file has to find a preimage

Second preimage resistance

2nd preimage

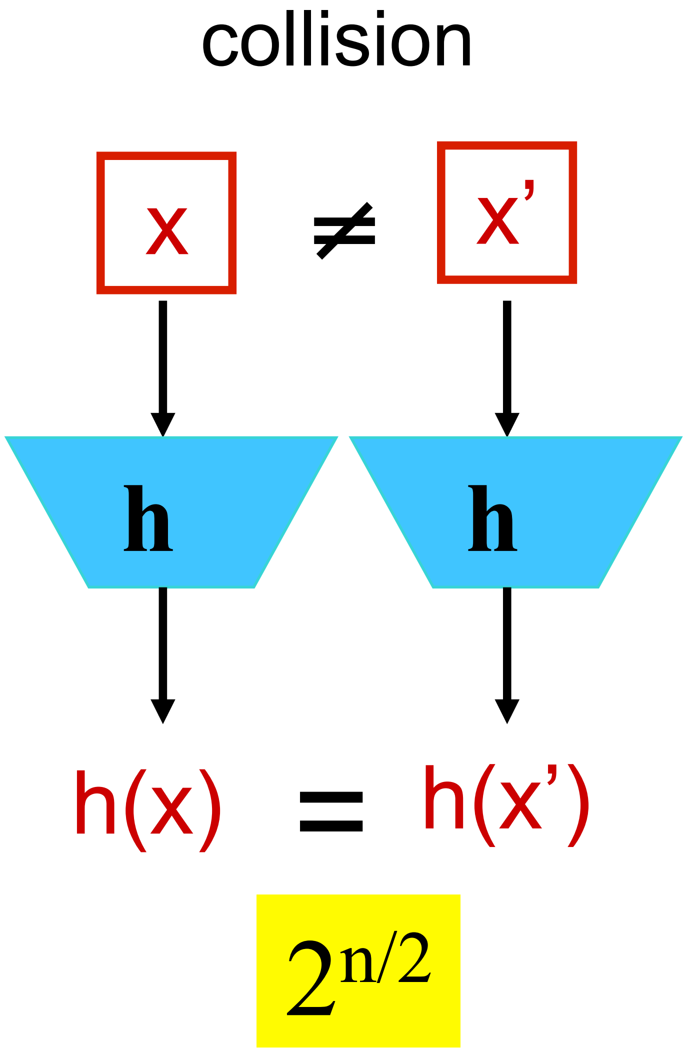




- an attacker can modify x but not h(x)
- he can only fool the recipient if he finds a second preimage of x

Collision resistance (1/2)

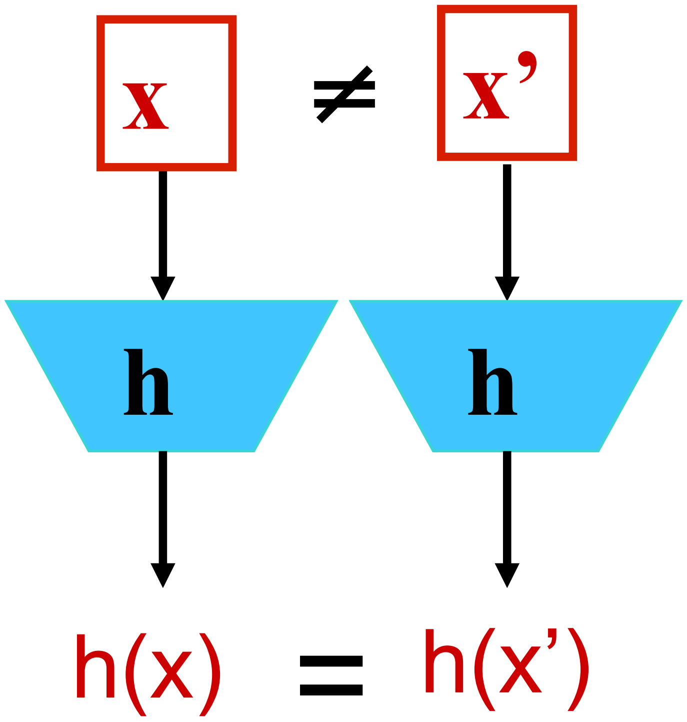
- hacker Alice prepares two versions of a software driver for the O/S company Bob
  - x is correct code
  - x’ contains a backdoor that gives Alice access to the machine
- Alice submits x for inspection to Bob
- if Bob is satisfied, he digitally signs h (x) with his private key
- Alice now distributes x’ to users of the O/S; these users verify the signature with Bob’s public key
- this signature works for x and for x’, since h(x) = h(x’)!



Collision resistance (2/2)

- in many cryptographic protocols, Alice wants to commit to a value x without revealing it
- Alice picks a secret random string r and sends y = h(x || r) to Bob
- in a later phase of the protocol, Alice reveals x and r to Bob and he checks that y is correct
- if Alice can find a collision, that is (x,r) and (x',r') with x' ≠ x she can cheat
- if Bob can find a preimage, he can learn x and cheat

collision



$2^{n/2}$

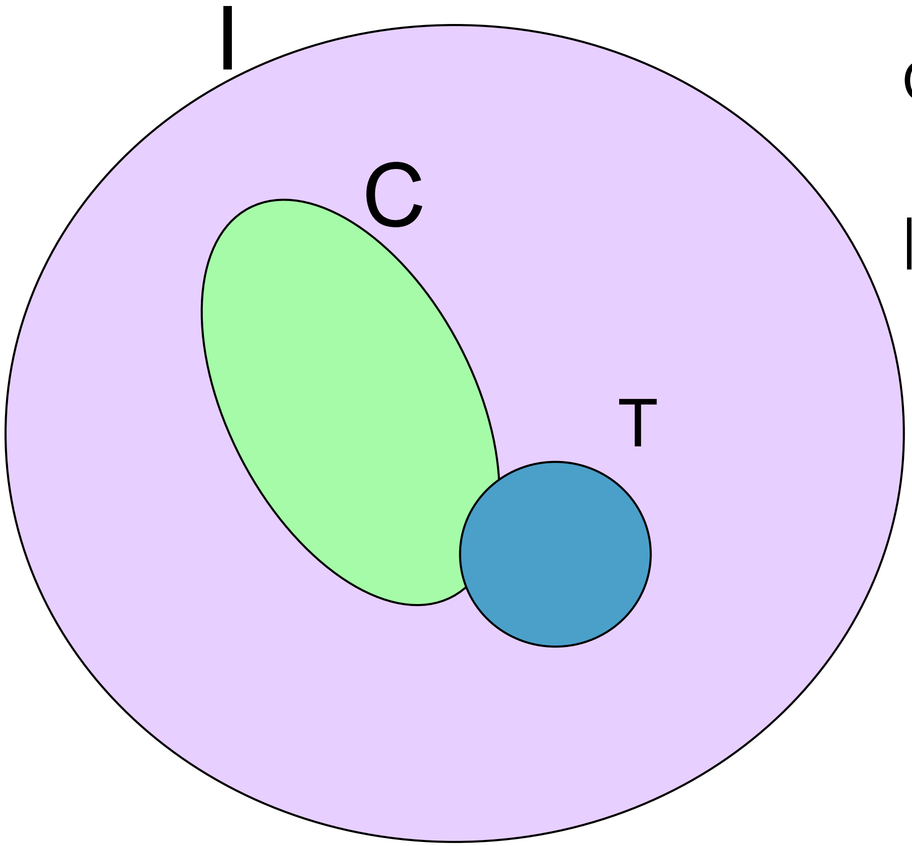
Brute force (2^{nd}) preimage

- **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 (2^{nd}) preimage: $\Theta(2^{2n/3})$ [Hellman'80]
- answer: randomize hash function with a parameter S (salt, key, spice,…)

The birthday paradox

- given a set with S elements
- choose r elements at random (with replacements) with $r \ll S$
- the probability p that there are at least 2 equal elements (a collision) $\cong 1 - \exp(-r(r-1)/2S)$
- more precisely, it can be shown that
  - $p \geq 1 - \exp(-r(r-1)/2S)$
  - if $r < \sqrt{2S}$ then $p \geq 0.6\, r(r-1)/2S$

How to find collisions?



I = space of pairs of messages; size $\approx (2^{2^{64}})^2$

C = space of all input messages that collide under h

$|C| \approx 2^{-n} | I |$

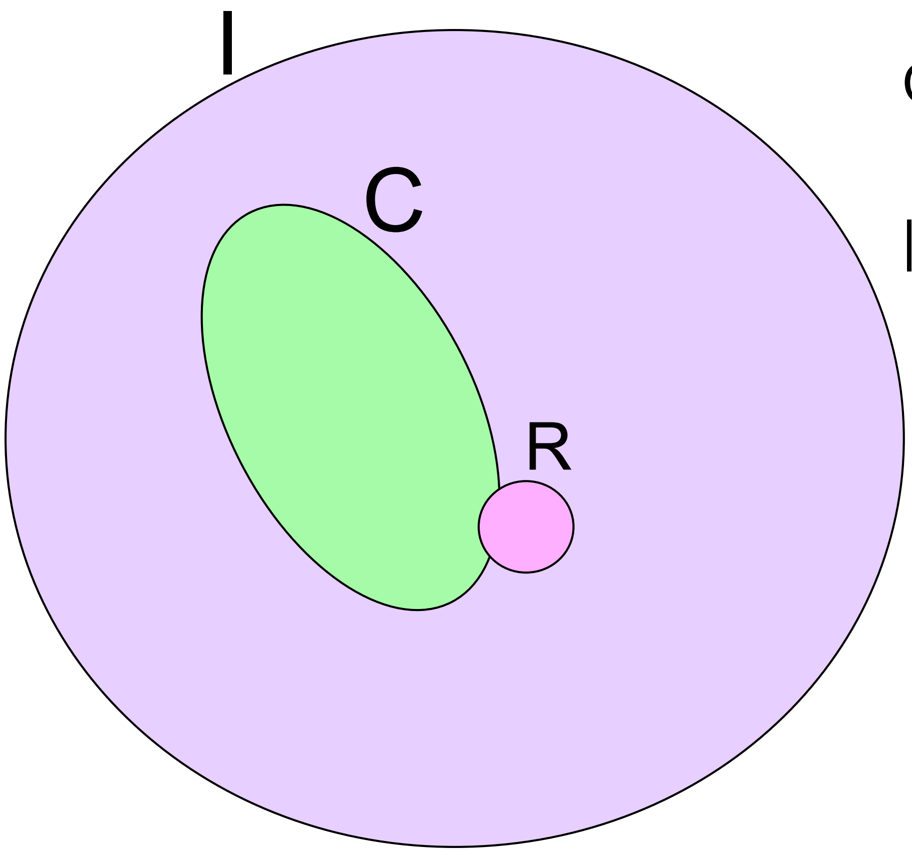
Collision search algorithm 1

Pick 2^n random message pairs (x,x')

For each pair, Prob(h(x)=h(x')=2^{-n})

You expect to find a collision, that is, a non-empty intersection with C

How to find collisions?



I = space of pairs of messages; size $\approx (2^{2^{64}})^2$

C = space of all input messages that collide under h

$|C| \approx 2^{-n} | I |$

Collision search algorithm 2

Pick a set R of $2^{n/2}$ random messages

Find a collision

You expect to find a collision, that is, a non-empty intersection with C as there are about $2^n/2$ distinct pairs in R

Collision resistance

- hard to achieve in practice
  - many attacks
  - requires double output length $2^{n/2}$ versus 2^n
- hard to achieve in theory
  - [Simon'98] one cannot derive collision resistance from "general" preimage resistance (there exists no black box reduction)
- hard to formalize: requires
  - family of functions: key, parameter, salt, spice,…
  - "human ignorance" trick [Stinson'06], [Rogaway'06]

16

Relation between properties

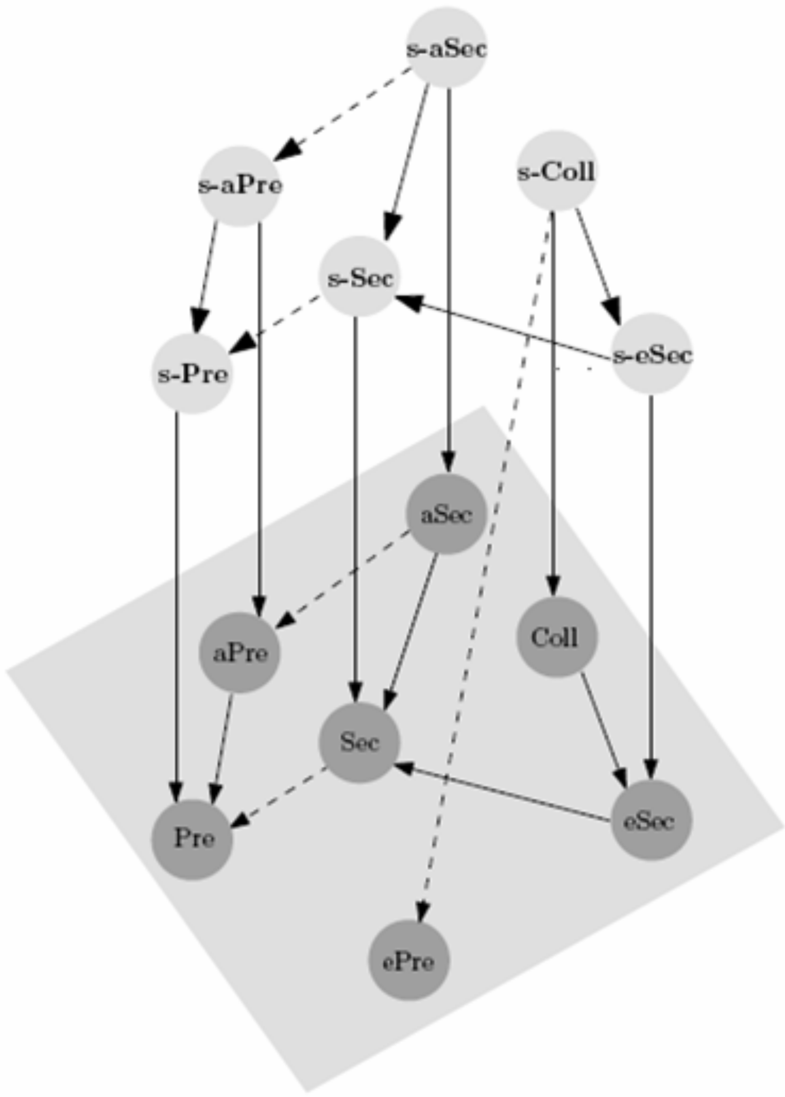
[Rogaway-Shrimpton’04]

[Stinson’06]

[Reyhanitabar-Susilo-Mu’10]

[Andreeva-Stam’10]

Even if Coll ⇒ xSEC/Pre: bound always $2^{n/2} << 2^n$

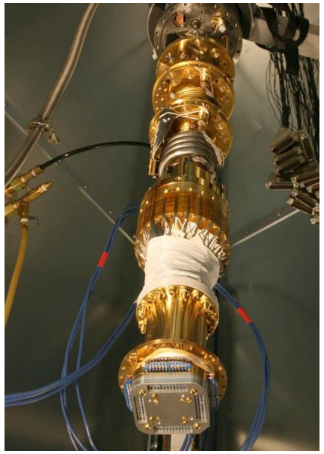


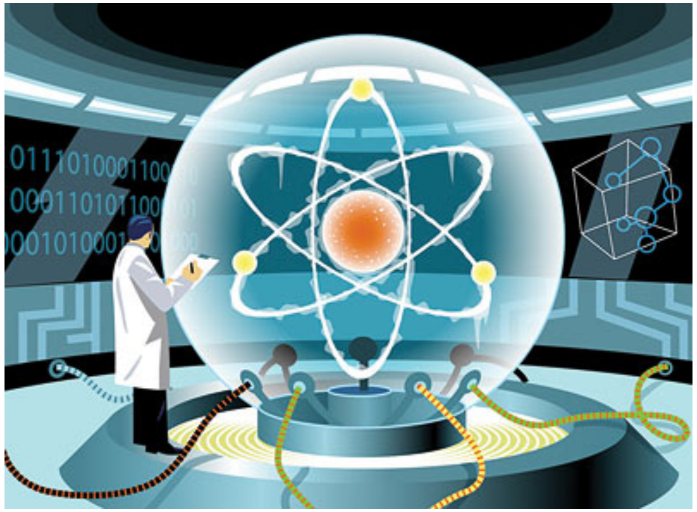
Brute force attacks in practice

- (2^{nd}) preimage search
  - n = 128: 23 B$ for 1 year if one can attack $\mathbf{2^{40}}$ 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 computers

- 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
  - near-collision resistance
  - partial preimage resistance (most of input known)
  - multiplication freeness
- how to formalize these requirements and the relation between them?

Iteration
(mode of compression function)

How not to construct a hash function

- Divide the message into t blocks x_i of n bits each

Message block 1: x_1

$\oplus$

Message block 2: x_2

$\oplus$

$\cdots$

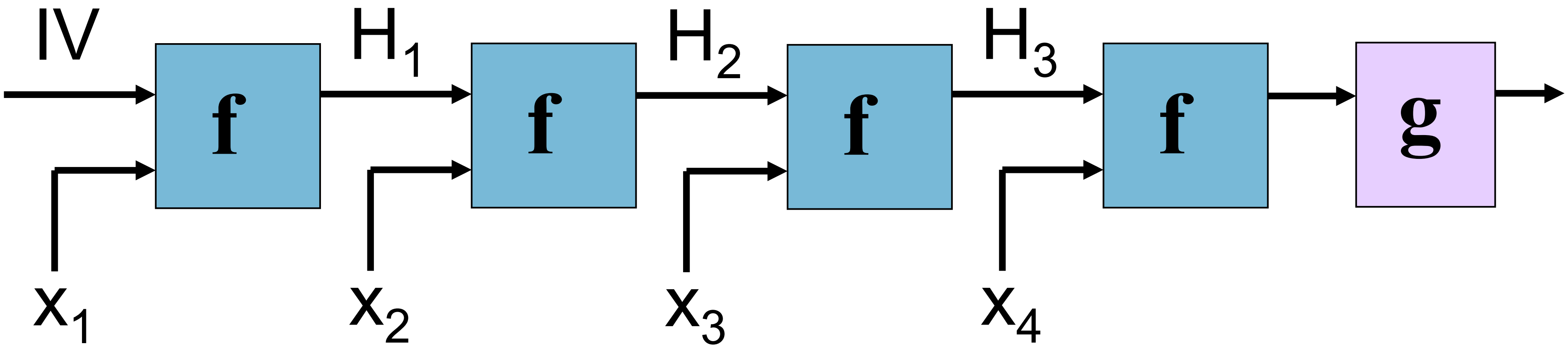
$\oplus$

Message block t: x_t

$=$

Hash value h(x)

Hash function: iterated structure

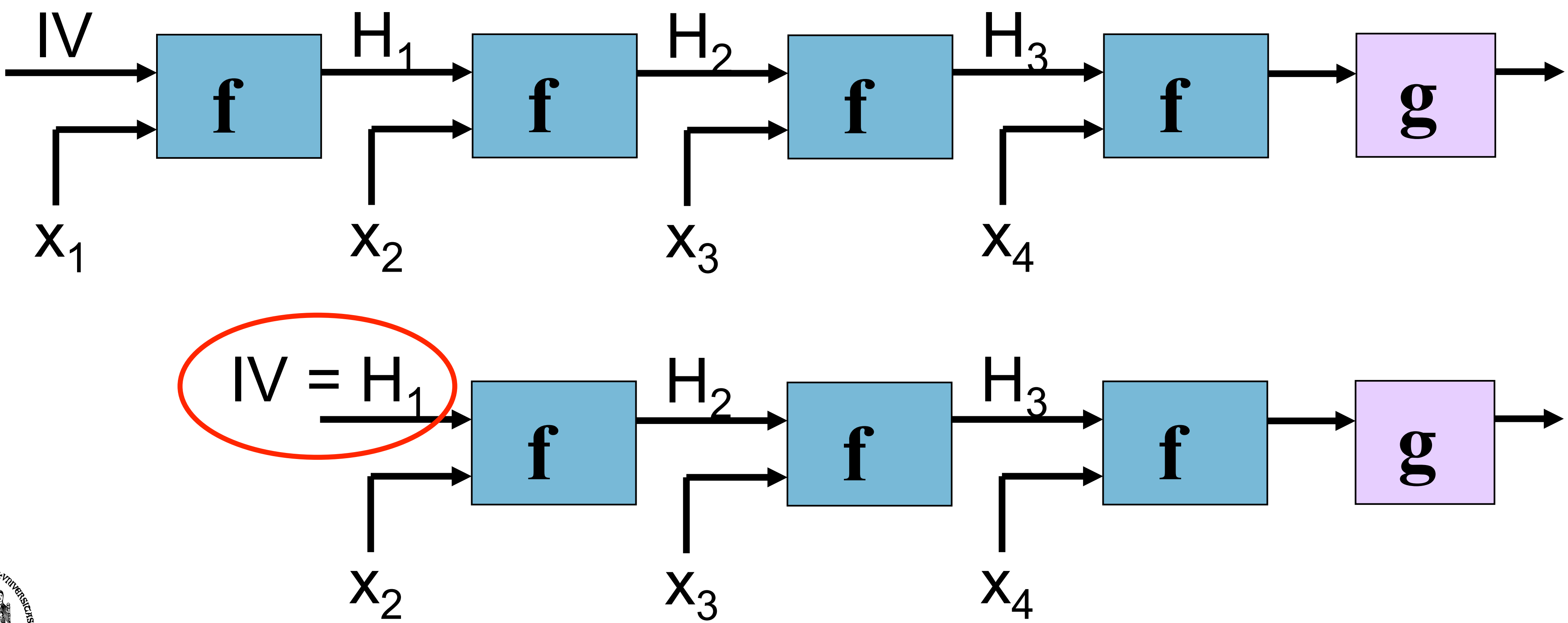


Split messages into blocks of fixed length and hash them block by block with a compression function f

Efficient and elegant
But …

Security relation between f and h

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



Security relation between f and h (2)

- 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

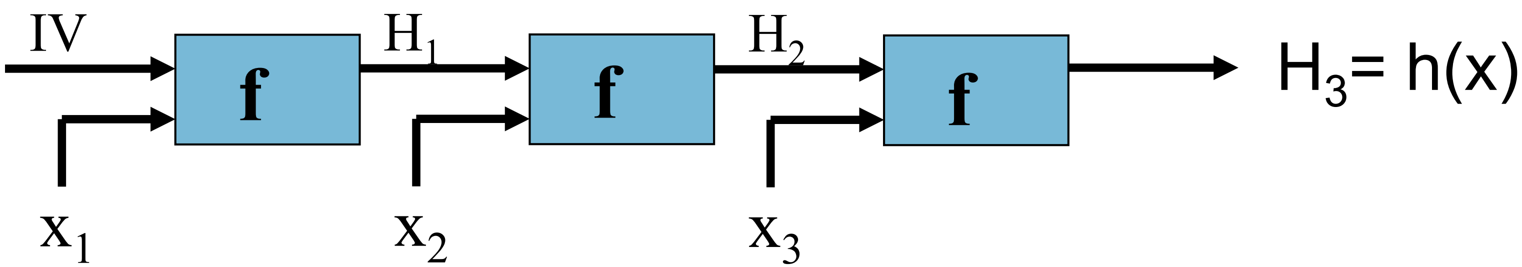
  [Merkle'89-Damgård'89]

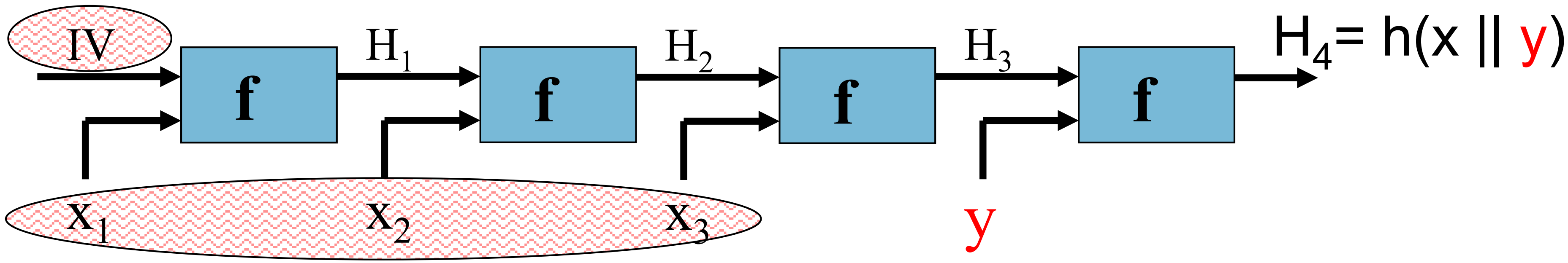
- f is ideally 2nd preimage resistant $\overset{?}{\Leftrightarrow}$ h is ideally 2nd preimage resistant [Lai-Massey'92]

- few hash functions have a strong compression function
- very few hash functions treat x_i and H_{i-1} in the same way

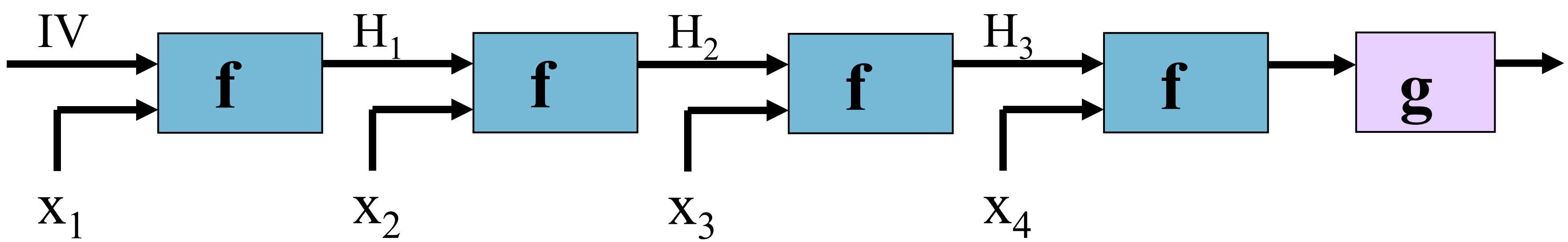
Security relation between f and h (3)

length extension: if one knows h(x), easy to compute h(x || y) without knowing x or IV





solution: output transformation



Property preservation

[Andreeva-Mennink-P'10] for overview

Sec/Pre preservation seems to be problematic

Is Pre preservation meaningful?

	Coll	Sec	Pre	Pro
Suffix- & Prefix-free MD				
Envelope MD				
BCM				?
Haifa				
RMX				

More on property preservation/domain extension

- PRO preservation $\Rightarrow$ Col, Sec and Pre for ideal compression function
  - but for narrow pipe bounds for Sec and Pre are at most $2^{n/2}$ rather than 2^n
- [...]

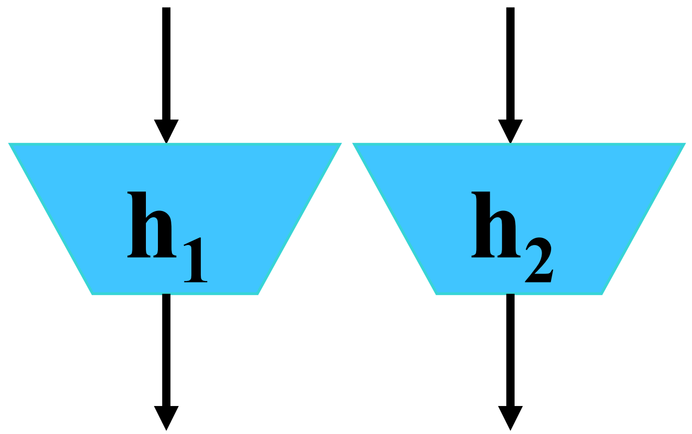
Attacks on MD-type iterations

- **multi-collision attack and impact on concatenation** [Joux'04]
- **long message 2^{nd} preimage attack**
  [Dean-Felten-Hu'99], [Kelsey-Schneier'05]
  – Sec security degrades lineary with number 2^t of message blocks hashed: $2^{n-t+1} + t\ 2^{n/2+1}$
  – appending the length does not help here!
- **herding attack** [Kelsey-Kohno'06]
  – reduces security of commitment using a hash function from 2^n
  – on-line 2^{n-t} + precomputation $2.2^{(n+t)/2}$ + storage 2^t

How (NOT) to strengthen a hash function?

[Joux'04]

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



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

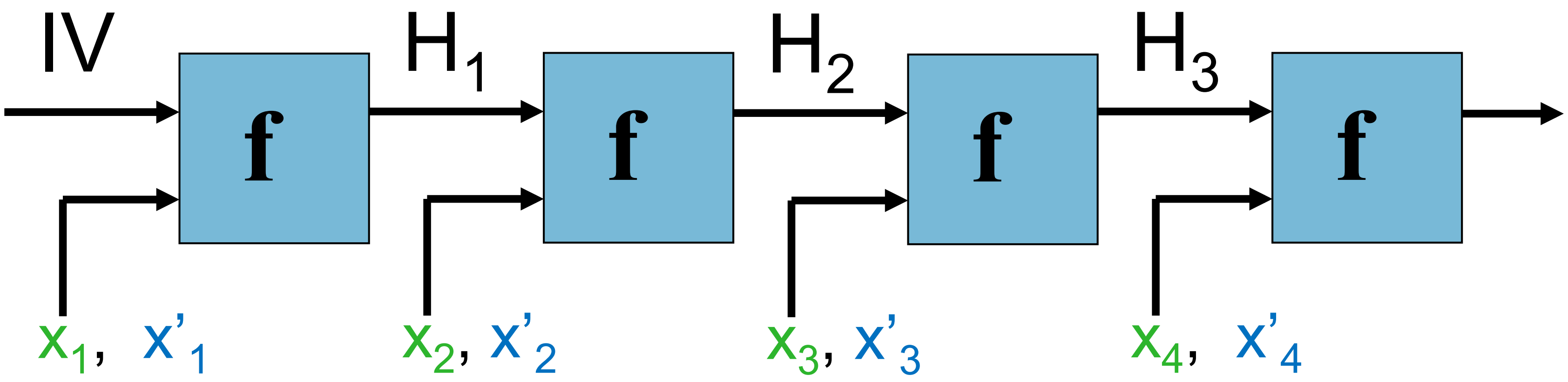
- intuition: the strength of g against collision/(2^{nd}) preimage attacks is the product of the strength of h_1 and h_2
  - if both are "independent"
- but….

Multiple collisions $\neq$ multi-collision

Assume "ideal" hash function h with n-bit result

- $\Theta(2^{n/2})$ evaluations of h (or steps): 1 collision
  - $h(x)=h(x')$
- $\Theta(r.\ 2^{n/2})$ steps: r^2 collisions
  - $h(x_1)=h(x_1')$; $h(x_2)=h(x_2')$; … ; $h(x_{r^2})=h(x_{r^2}')$
- $\Theta(2^{2n/3})$ steps: a 3-collision
  - $h(x)= h(x')=h(x'')$
- $\Theta(2^{n(t-1)/t})$ steps: a t-fold collision (multi-collision)
  - $h(x_1)= h(x_2)= \ldots =h(x_t)$

Multi-collisions on iterated hash function (2)



- for IV: collision for block 1: x_1, x'_1
- for H_1: collision for block 2: x_2, x'_2
- for H_2: collision for block 3: x_3, x'_3
- for H_3: collision for block 4: x_4, x'_4

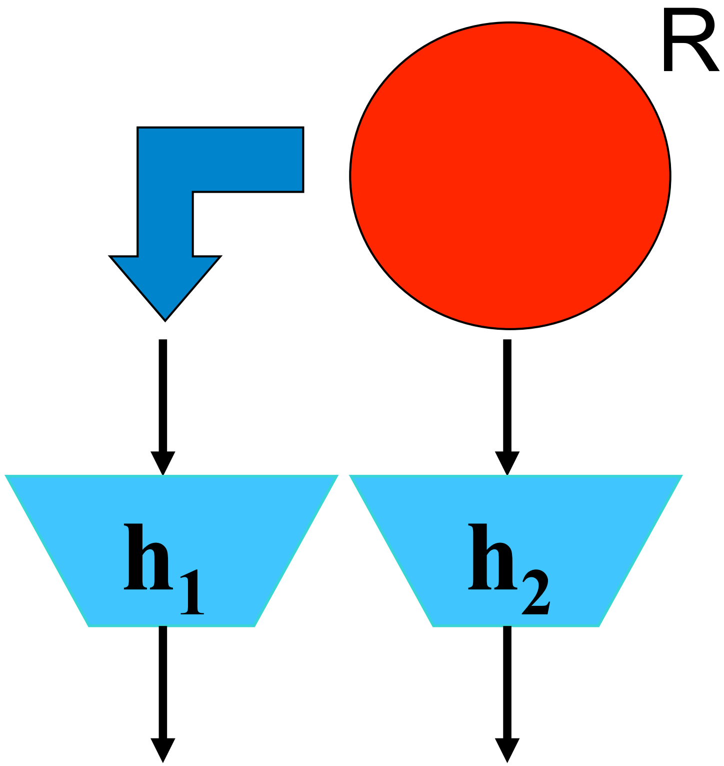
- now $h(x_1||x_2||x_3||x_4) = h(x'_1||x_2||x_3||x_4) = h(x'_1||x'_2||x_3||x_4) = \ldots$ $= h(x'_1||x'_2||x'_3||x'_4)$ **a 16-fold collision (time: 4 collisions)**

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



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

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

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

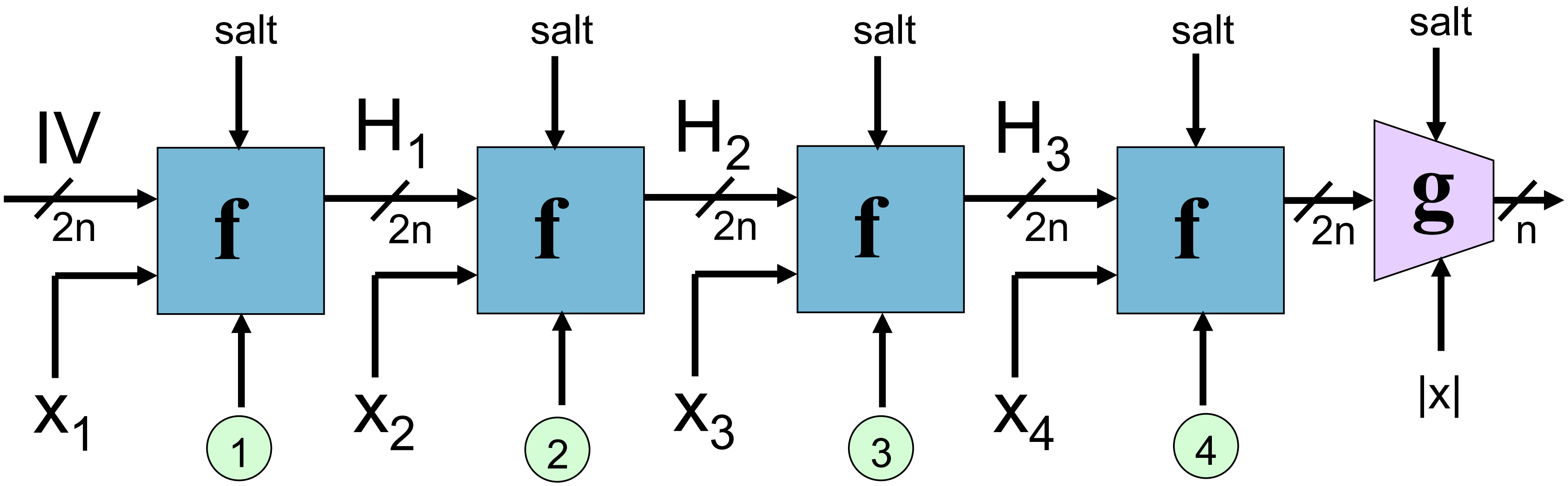
- cost of (2nd) preimage attack against g at most

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

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

Summary

salt + output transformation + counter + wide pipe



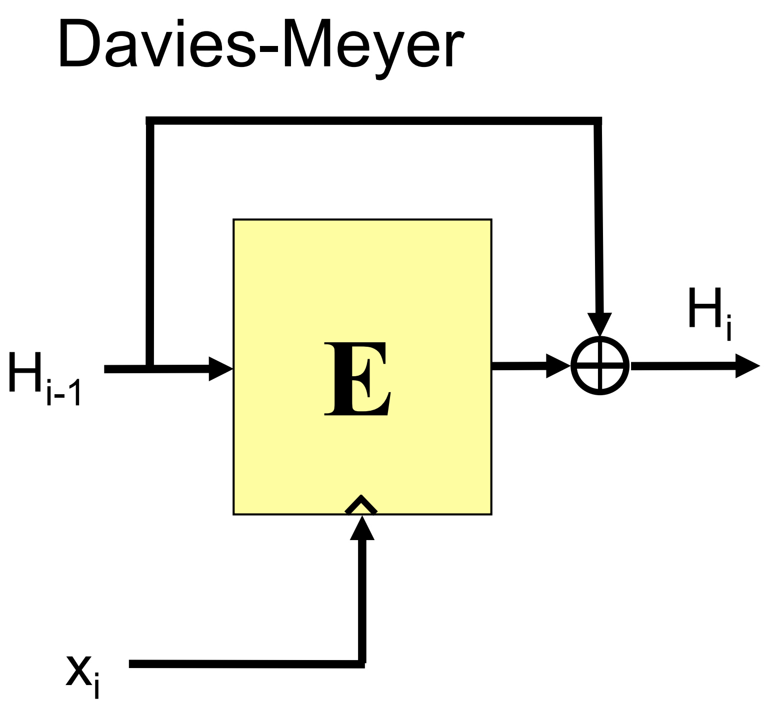
security reductions well understood
many more results on property preservation
impact of theory limited

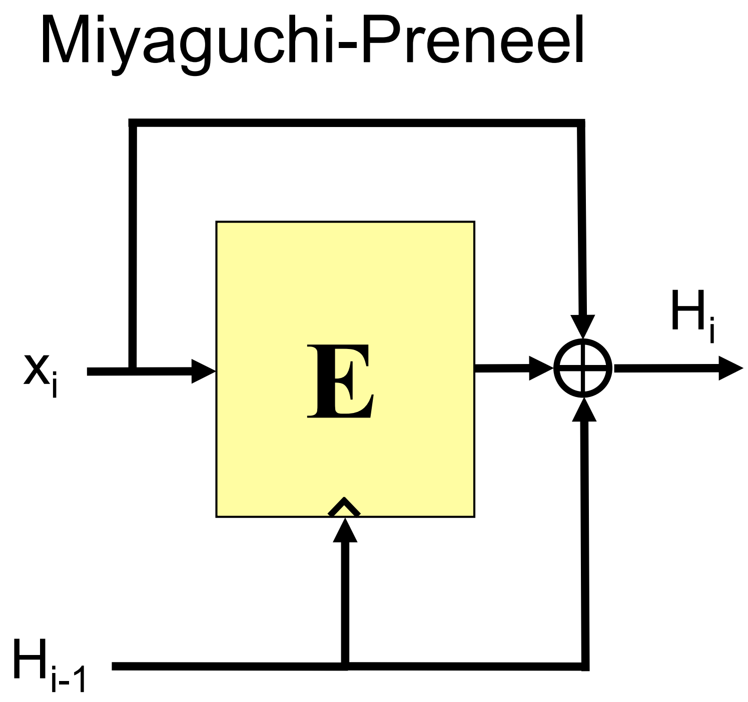
Improving MD iteration

- 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 $f \rightarrow f_i$ [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]

Compression functions

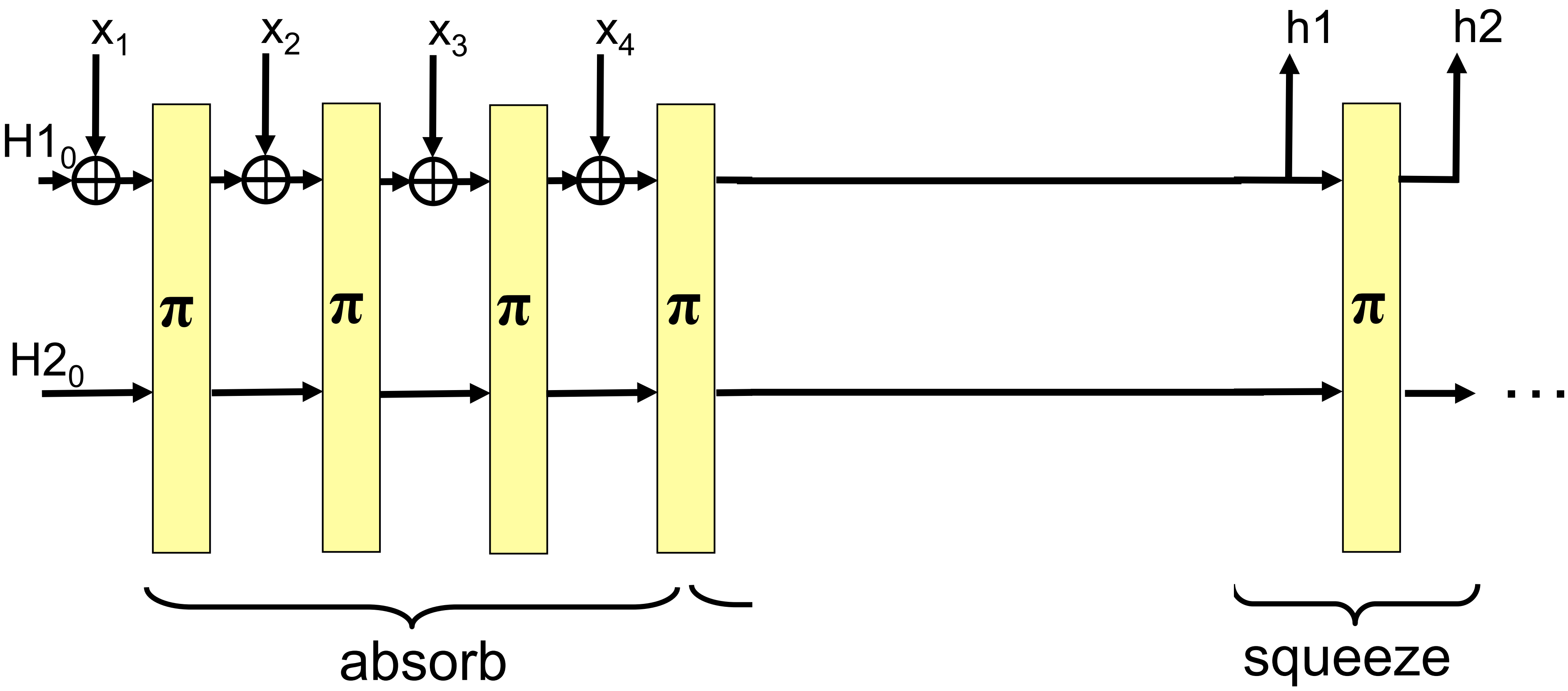
Block cipher (E_K) based





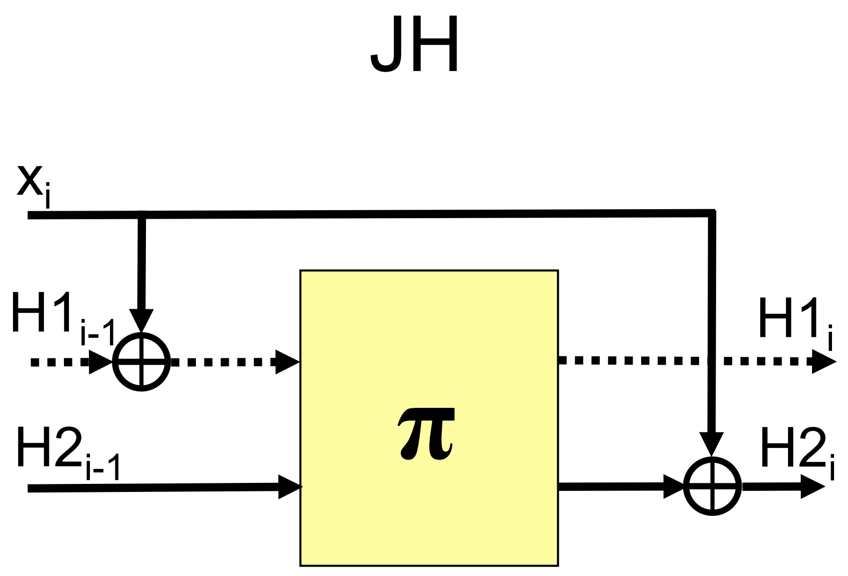
- output length = block length
- 12 secure compression functions (in ideal cipher model)
- requires 1 key schedule per encryption
- analysis [Black-Rogaway-Shrimpton'02], [Duo-Li'06], [Stam'09],…

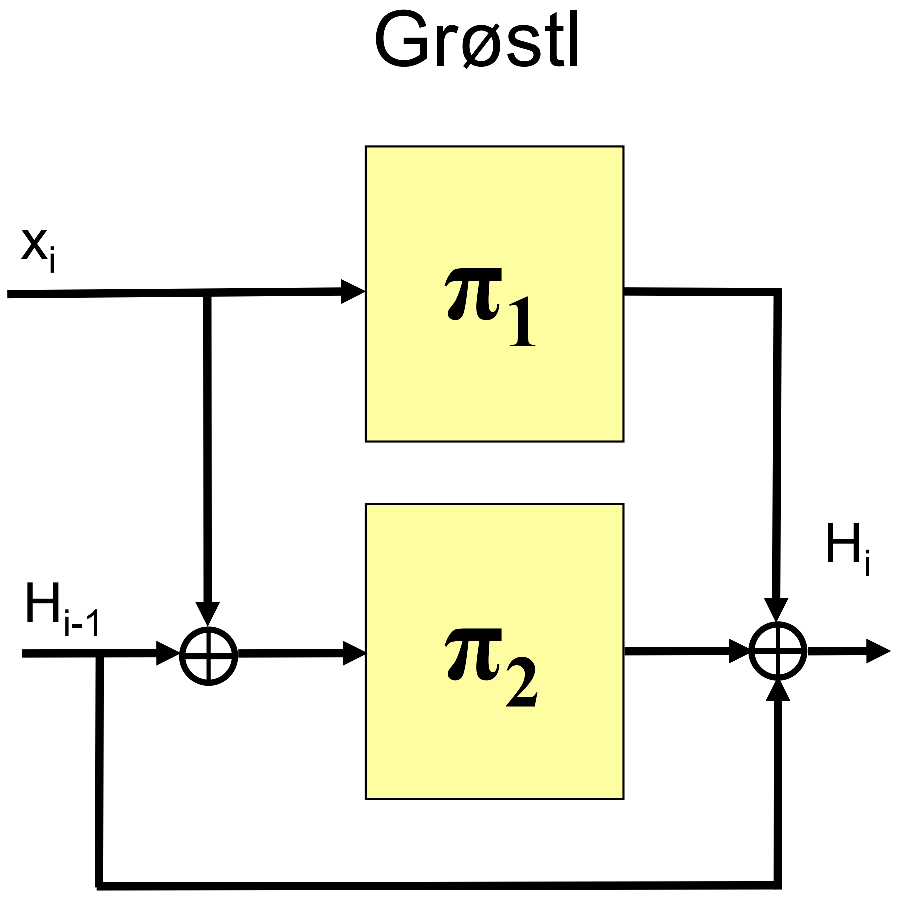
Permutation (π) based: sponge



Examples: Panama, RadioGatun, Grindahl,
Keccak (no buffer)

small permutation





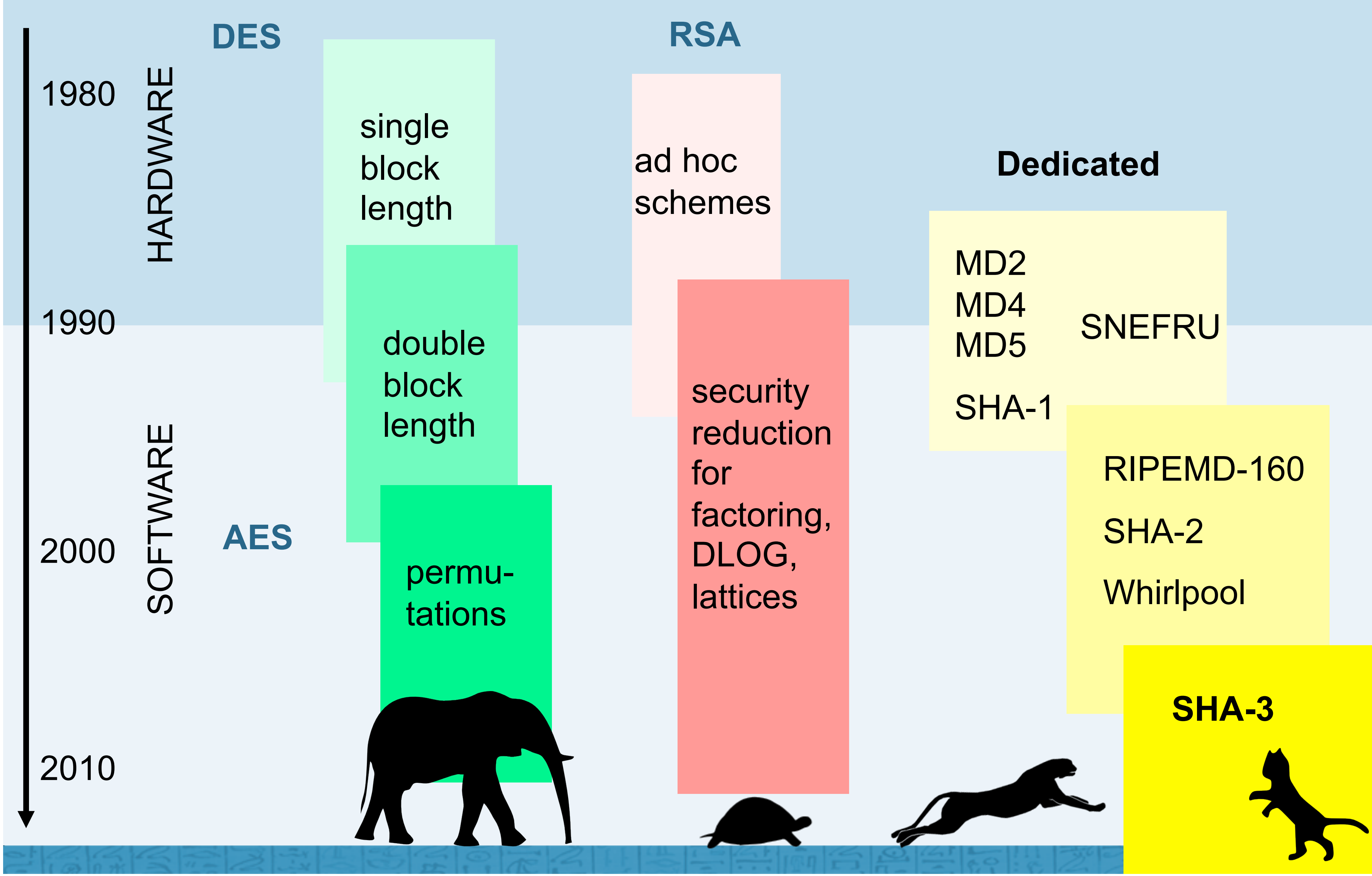
Iteration modes and compression functions

- security of simple modes well understood
- powerful tools available

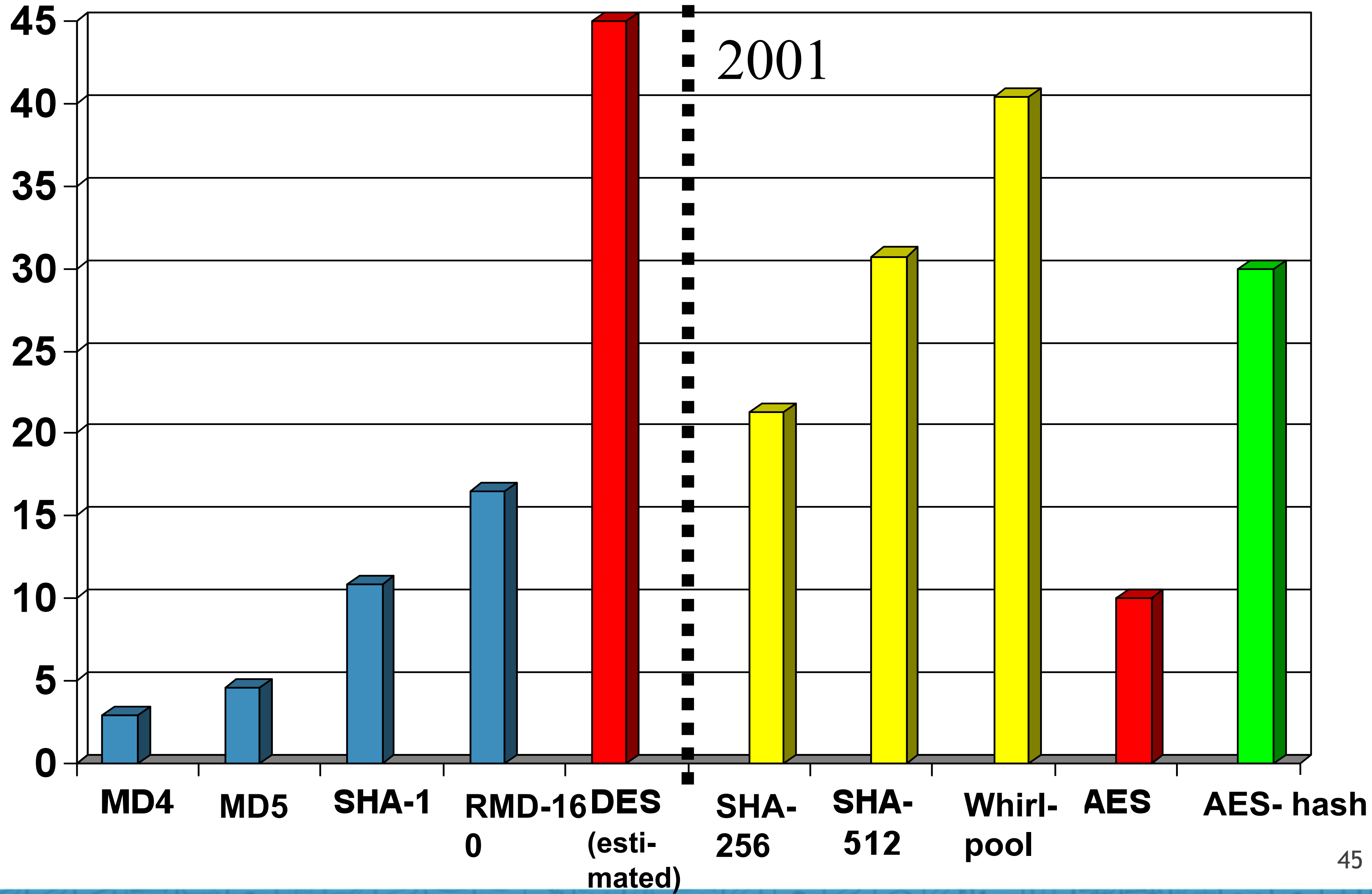
- analysis of slightly more complex schemes very difficult
- which properties are meaningful?
- which properties are preserved?
- MD versus sponge is still open debate

SHA-{0,1,2}

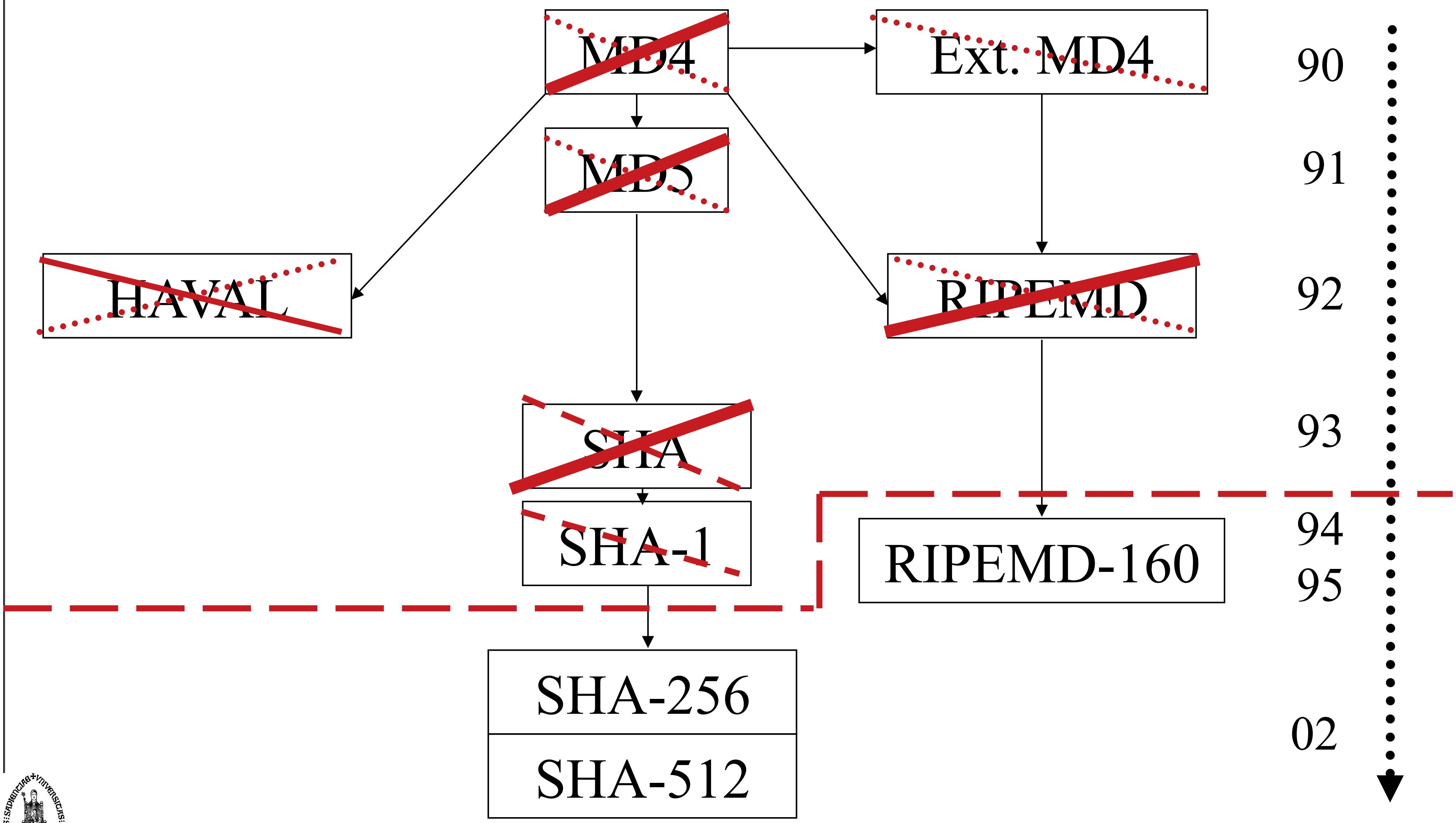
Hash function history 101



Performance of hash functions [Bernstein-Lange]
(cycles/byte) AMD Intel Pentium D 2992 MHz (f64)

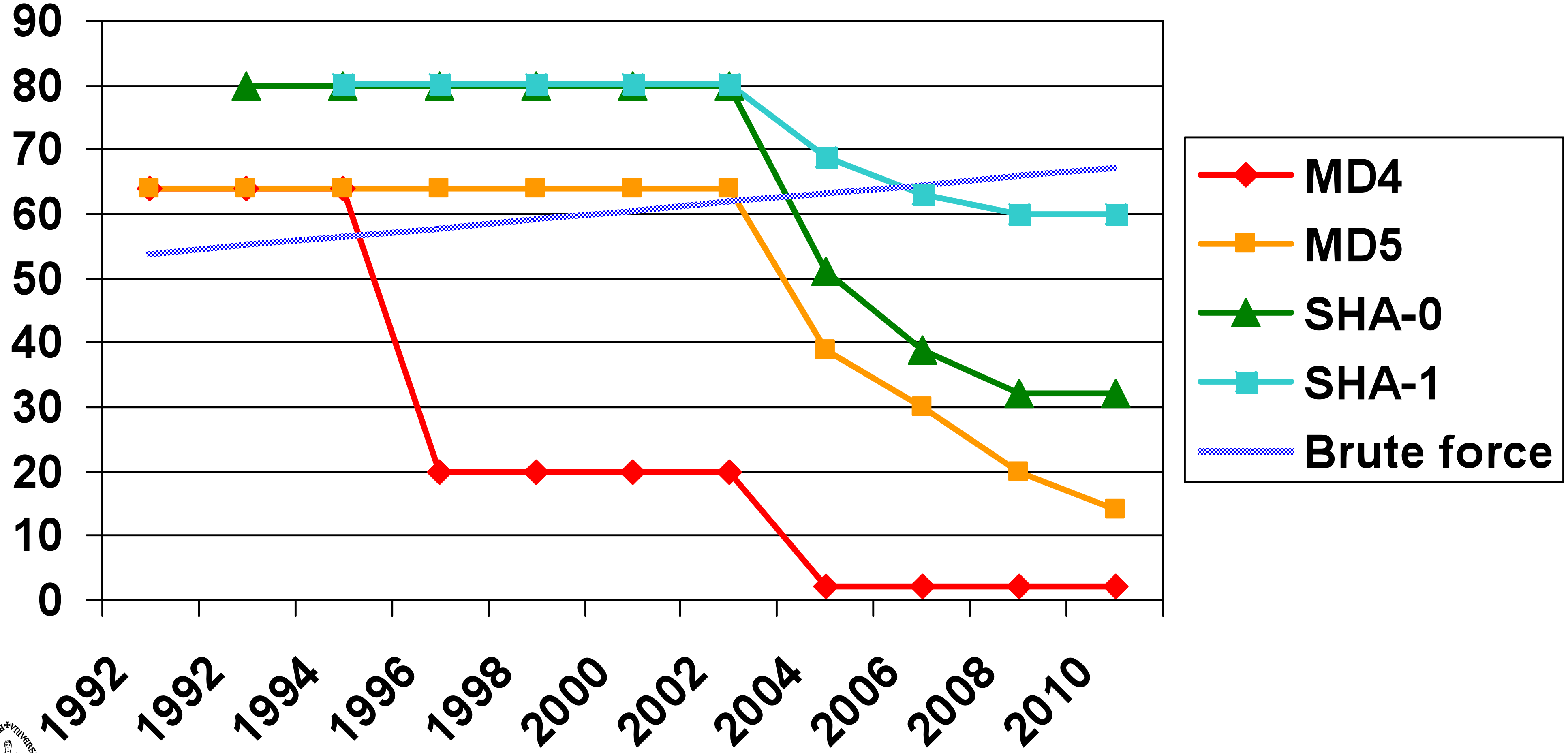


MDx-type hash function history

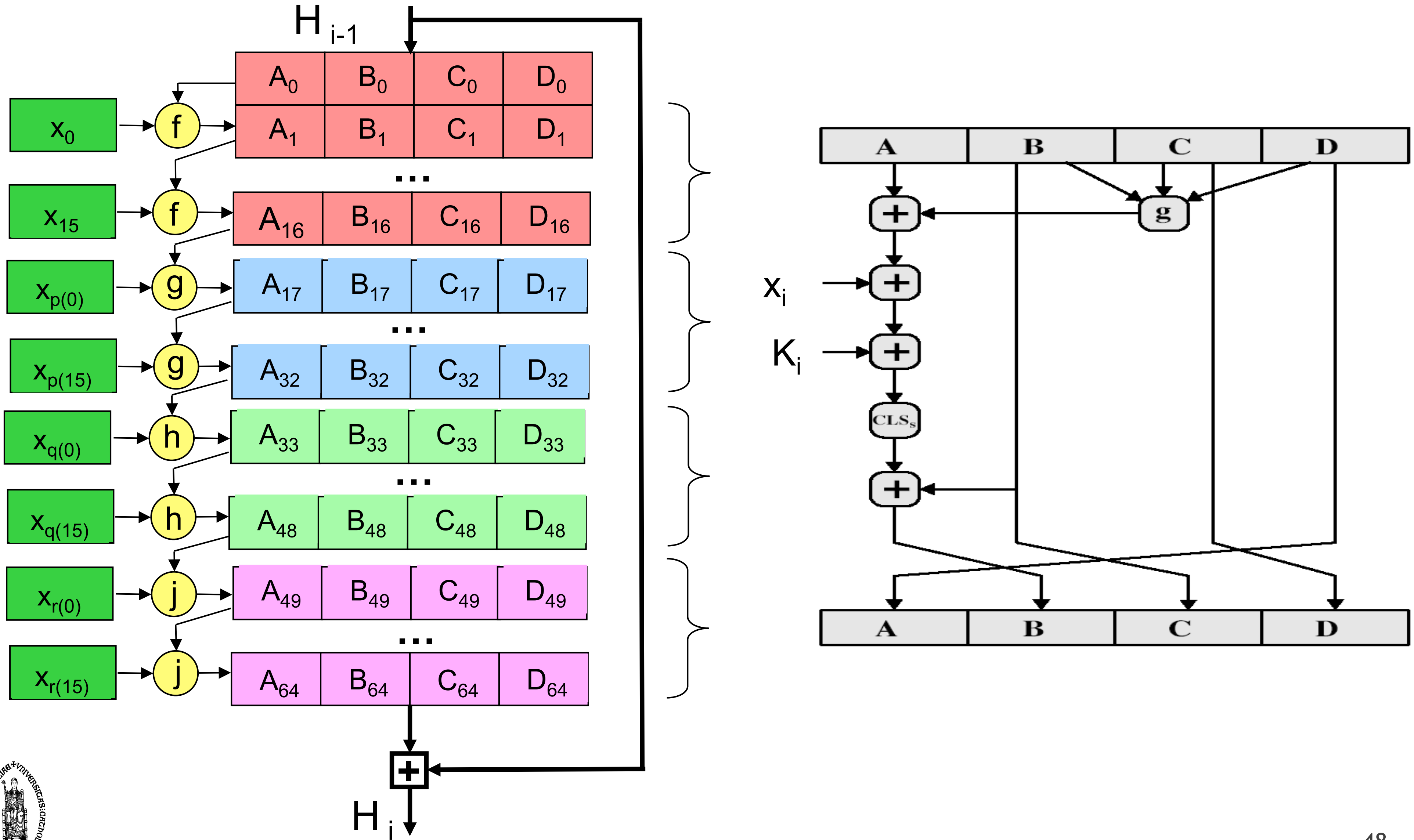


The complexity of collision attacks

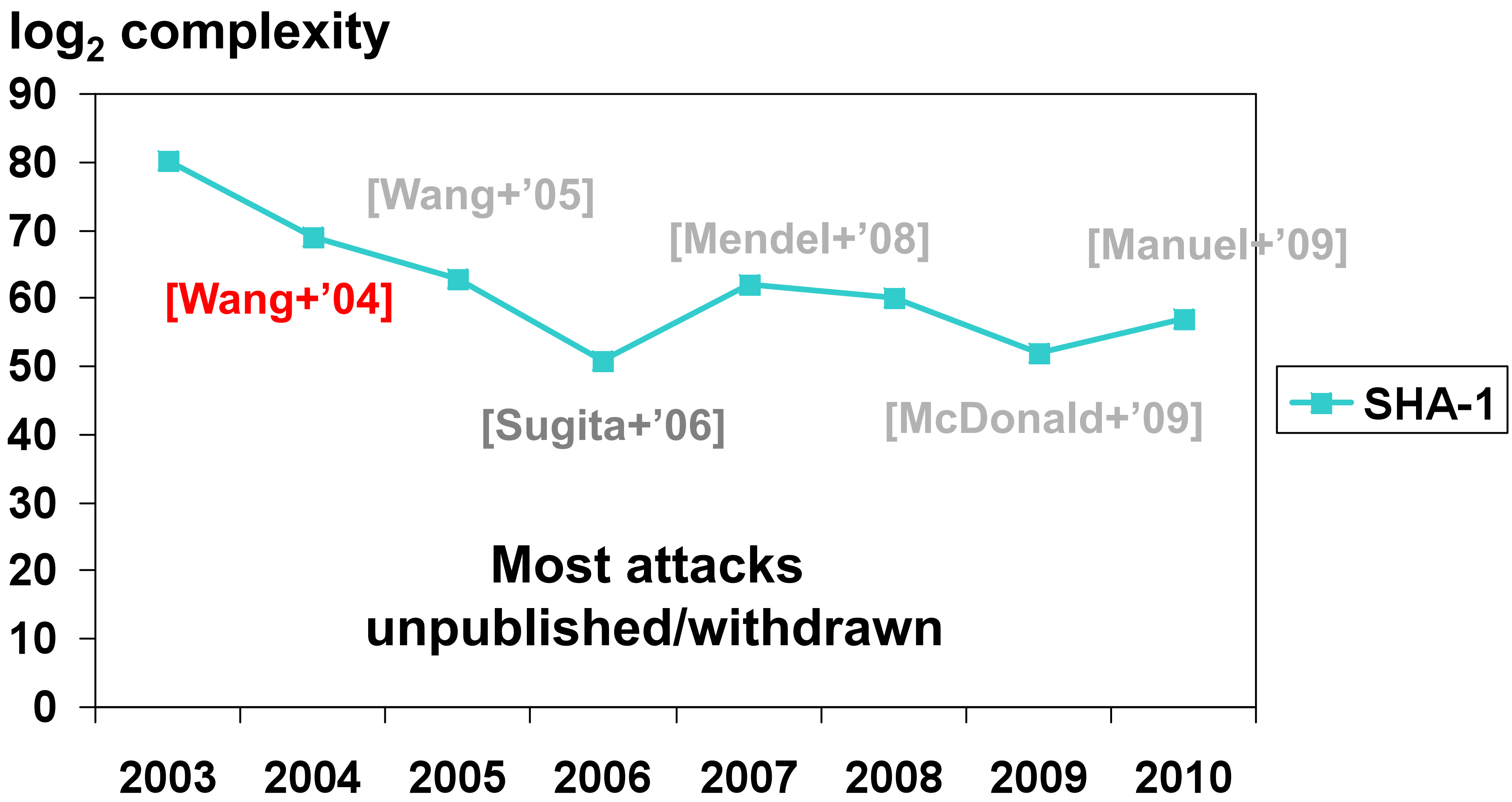
brute force: 1 million PCs (1 year) or US$ 100,000 hardware (4 days)



MD5 [Rivest'91]
4 rounds of 16 steps

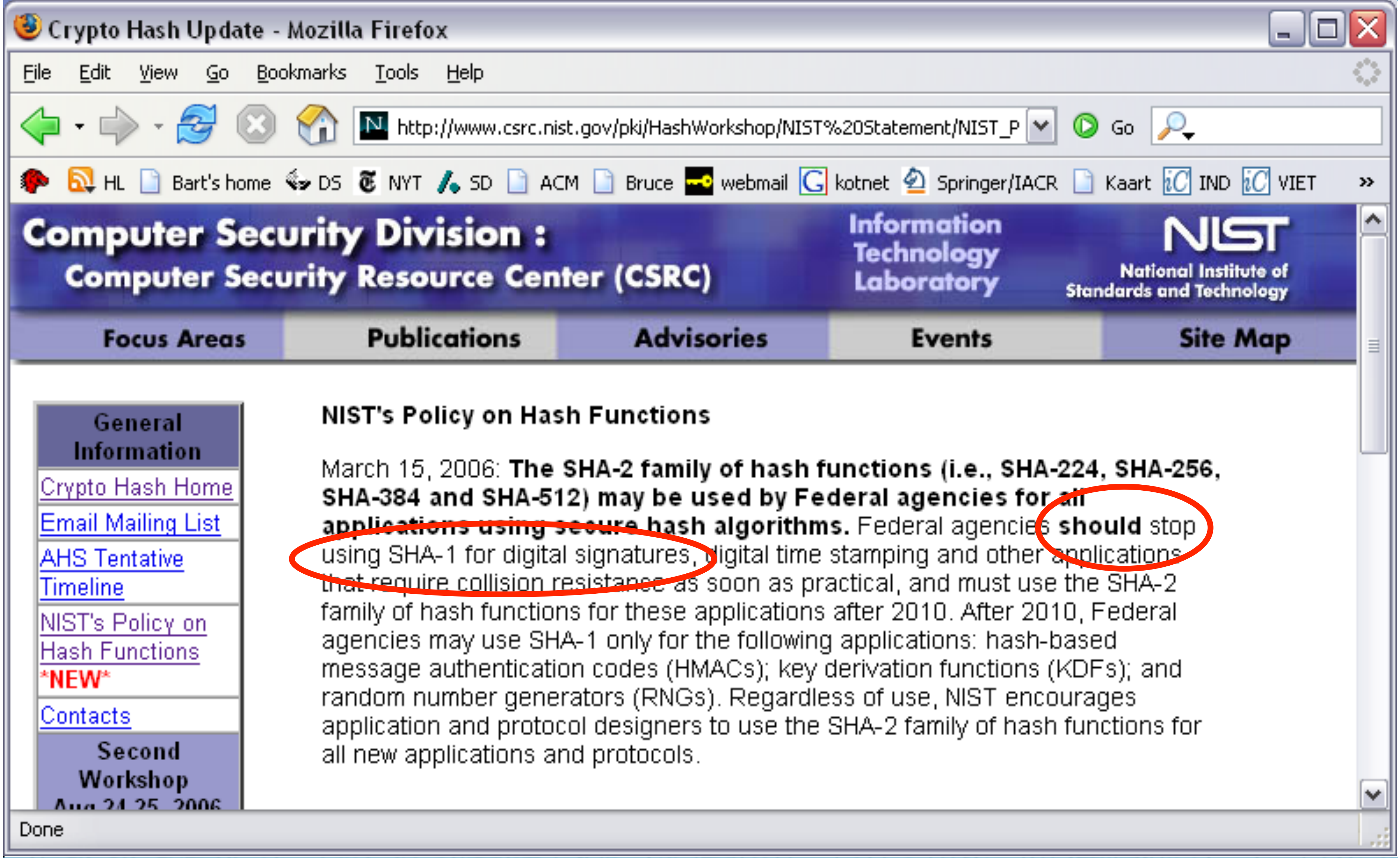


SHA-1



prediction: collision for SHA-1 in the next 12-18 months

NIST and SHA-1



Crypto Hash Update - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

http://www.csrc.nist.gov/pki/HashWorkshop/NIST%20Statement/NIST_P

HL Bart's home DS NYT SD ACM Bruce webmail kotnet Springer/IACR Kaart IND VIET

Computer Security Division : Computer Security Resource Center (CSRC)

Information Technology Laboratory

NIST National Institute of Standards and Technology

Focus Areas | Publications | Advisories | Events | Site Map

General Information

Crypto Hash Home

Email Mailing List

AHS Tentative Timeline

NIST's Policy on Hash Functions *NEW*

Contacts

Second Workshop Aug 24-25, 2006

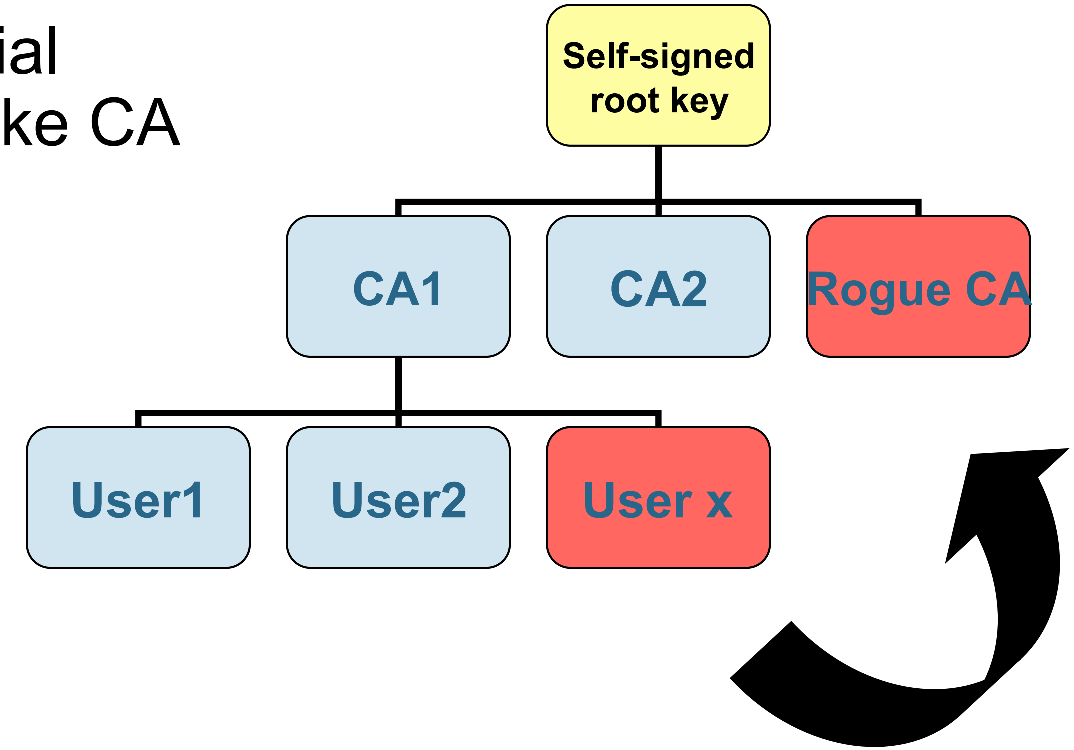
NIST's Policy on Hash Functions

March 15, 2006: **The SHA-2 family of hash functions (i.e., SHA-224, SHA-256, SHA-384 and SHA-512) may be used by Federal agencies for all applications using secure hash algorithms.** Federal agencies **should** stop using SHA-1 for digital signatures, digital time stamping and other applications that require collision resistance as soon as practical, and must use the SHA-2 family of hash functions for these applications after 2010. After 2010, Federal agencies may use SHA-1 only for the following applications: hash-based message authentication codes (HMACs); key derivation functions (KDFs); and random number generators (RNGs). Regardless of use, NIST encourages application and protocol designers to use the SHA-2 family of hash functions for all new applications and protocols.

Done

Rogue CA attack
[Sotirov-Stevens-Appelbaum-Lenstra-Molnar-Osvik-de Weger ’08]

- request user cert; by special collision this results in a fake CA cert (need to predict serial number + validity period)



impact: **rogue CA** that can issue certs that are trusted by all browsers

- 6 CAs have issued certificates signed with MD5 in 2008:
  - Rapid SSL, Free SSL (free trial certificates offered by RapidSSL), TC TrustCenter AG, RSA Data Security, Verisign.co.jp

Upgrades

- RIPEMD-160 is good replacement for SHA-1
- upgrading algorithms is always hard
- TLS uses MD5 || SHA-1 to protect algorithm negotiation (up to v1.1)
- **upgrading negotiation algorithm is even harder: need to upgrade TLS 1.1 to TLS 1.2**

SHA-2 [NIST'02]

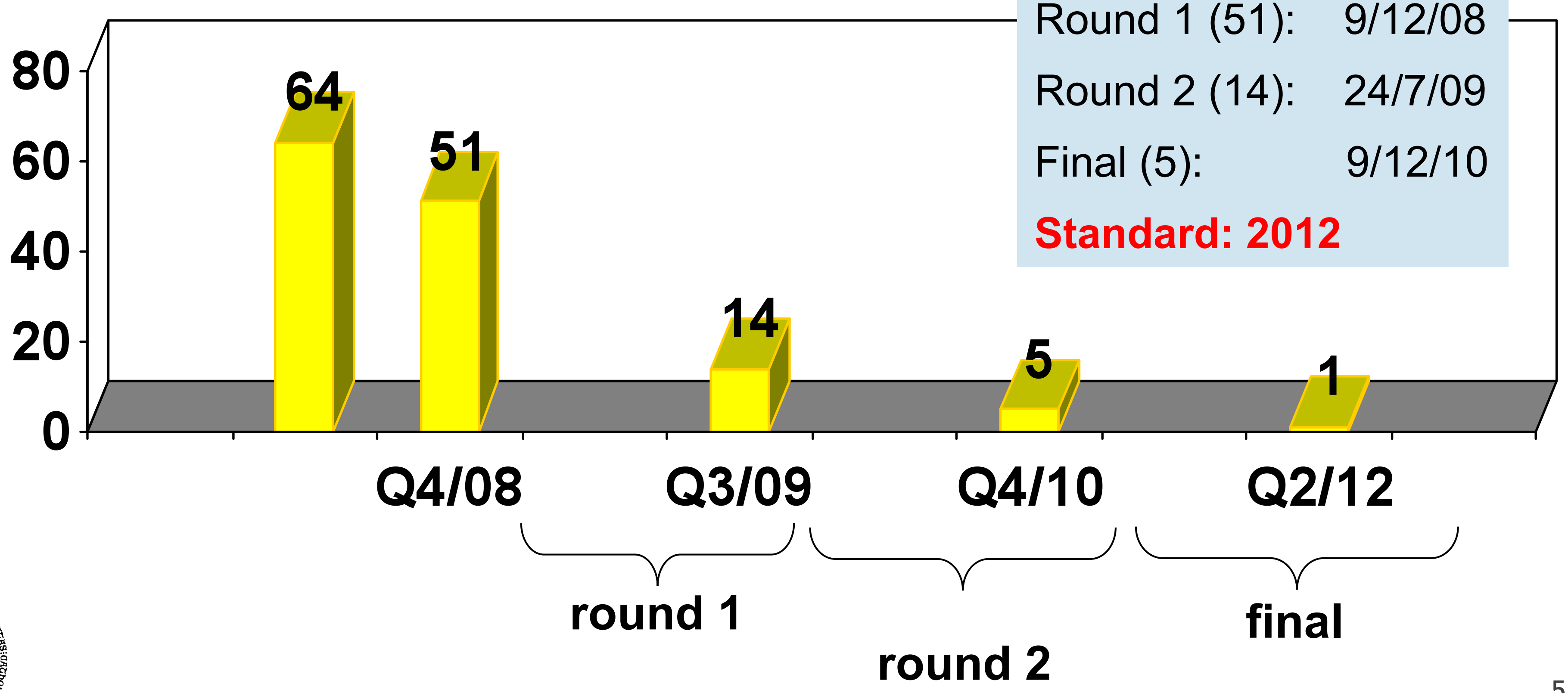
- SHA-224, SHA-256, SHA-384, SHA-512
  - non-linear message expansion
  - more complex operations
  - 64/80 steps
  - SHA-384 and SHA-512: 64-bit architectures
- SHA-256 collisions: 24/64 steps [Sanadhya-Sarkar'08]
- SHA-256 preimages: 43/64 steps [Aoki+'09]
- implementations today faster than anticipated
- adoption
  - industry may migrate to SHA-2 by 2011 or may wait for SHA-3
  - very slow for TLS/IPsec (no pressing need)

SHA-3
(bits and bytes)

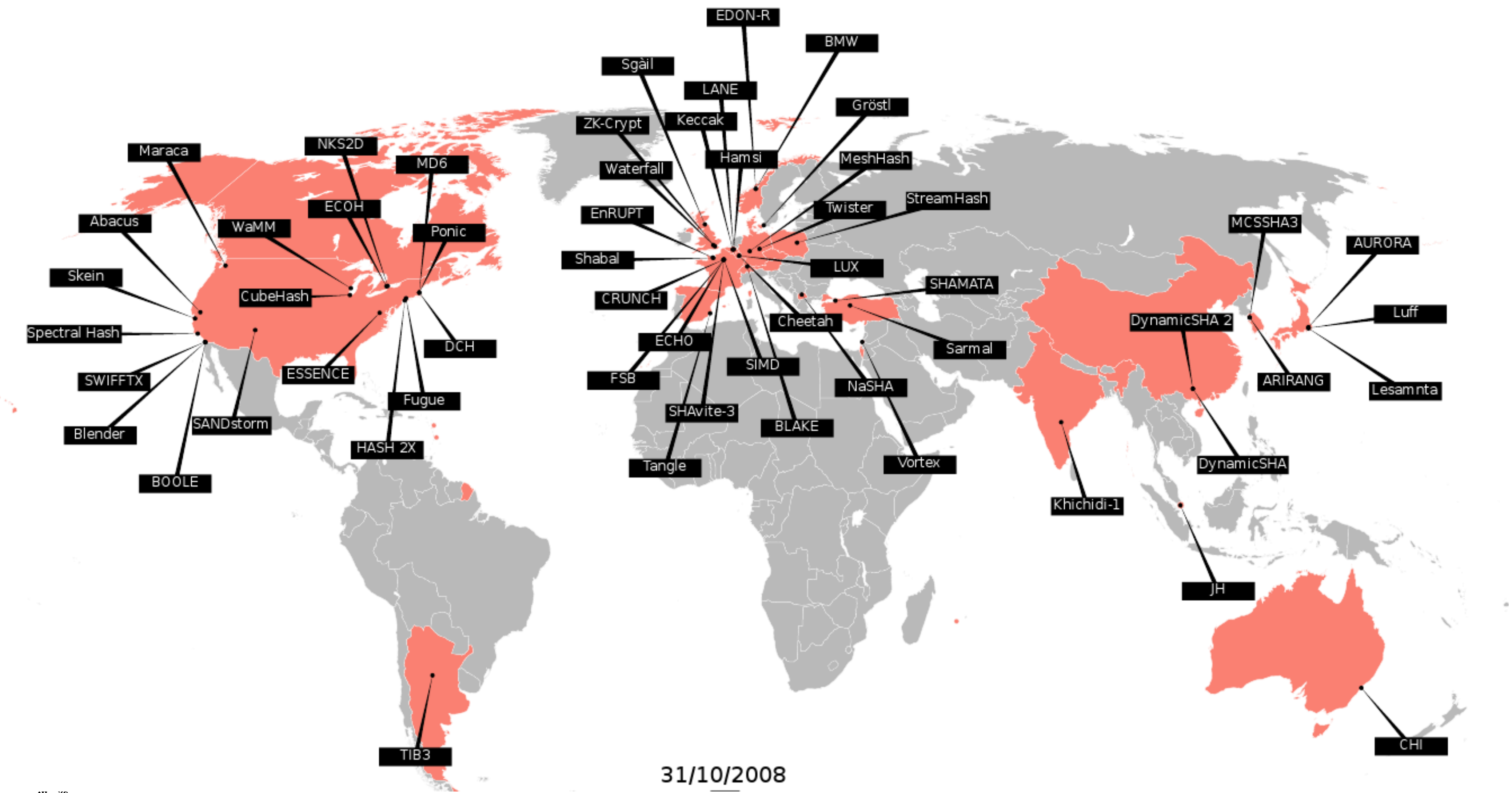
NIST AHS competition (SHA-3)

- SHA-3 must support 224, 256, 384, and 512-bit message digests, and must support a maximum message length of at least 2^{64} bits

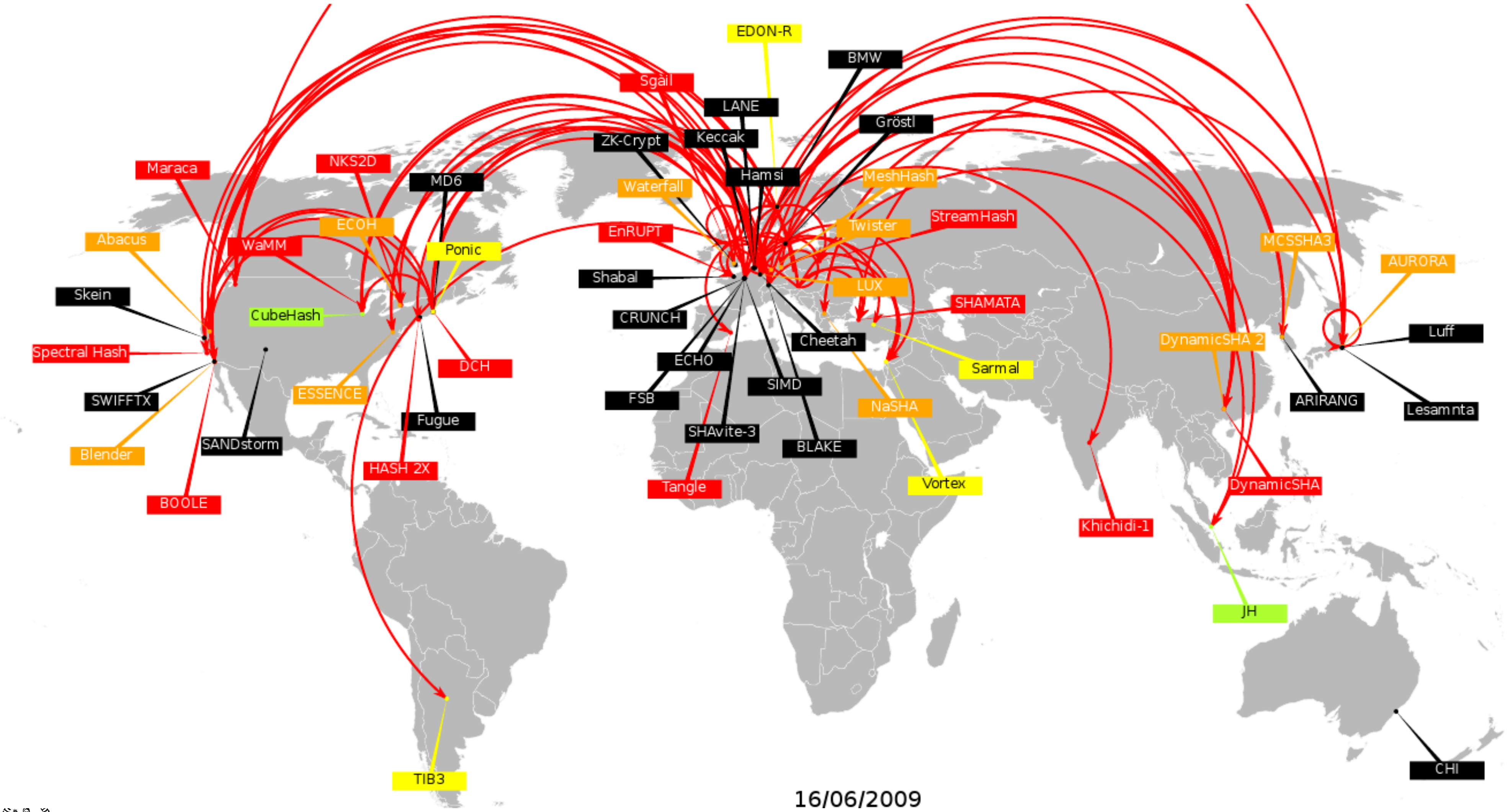
Call: 02/11/07
Deadline (64): 31/10/08
Round 1 (51): 9/12/08
Round 2 (14): 24/7/09
Final (5): 9/12/10
Standard: 2012



The candidates

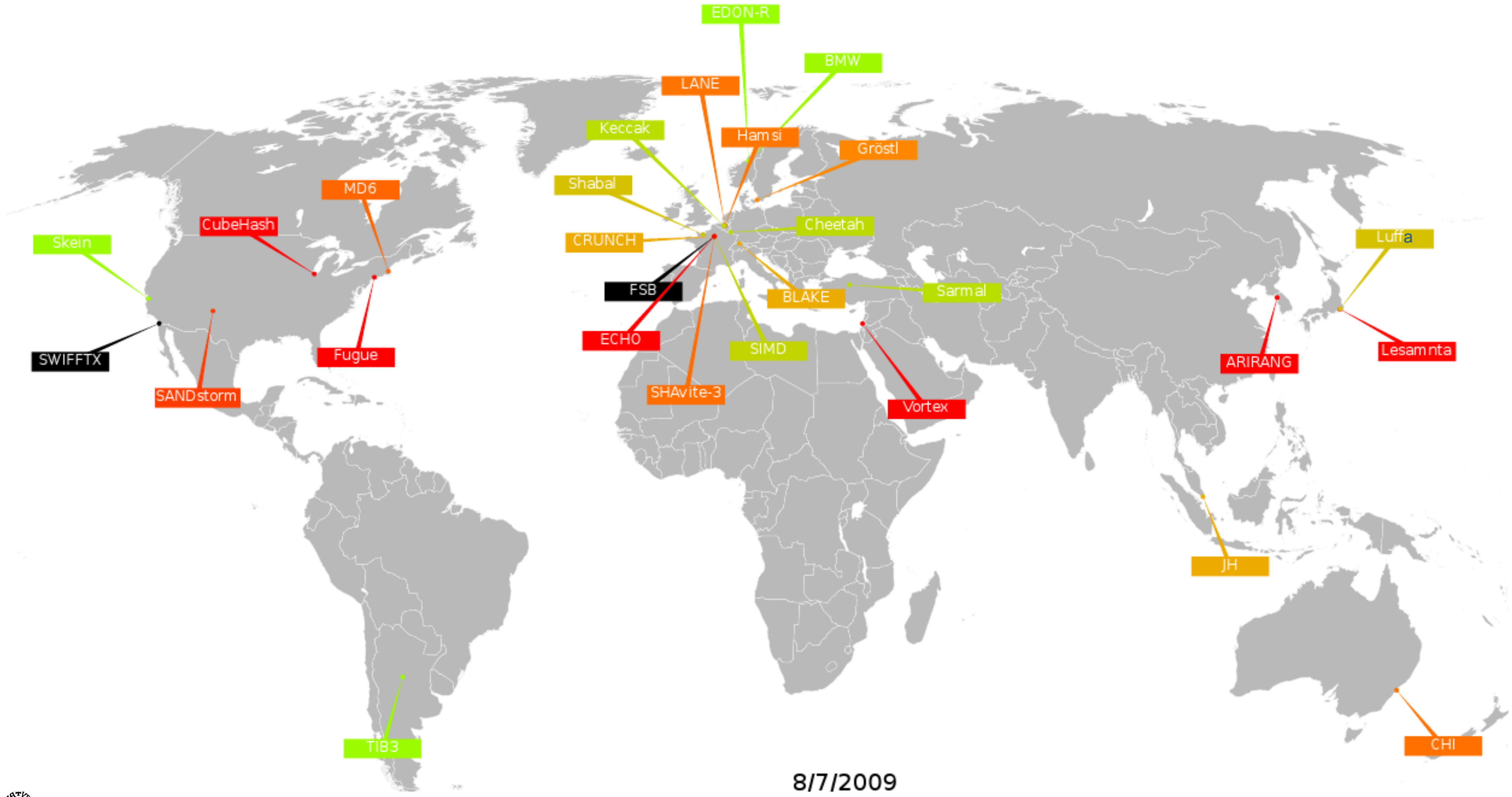


Preliminary cryptanalysis



Slide credit: Christophe De Cannière

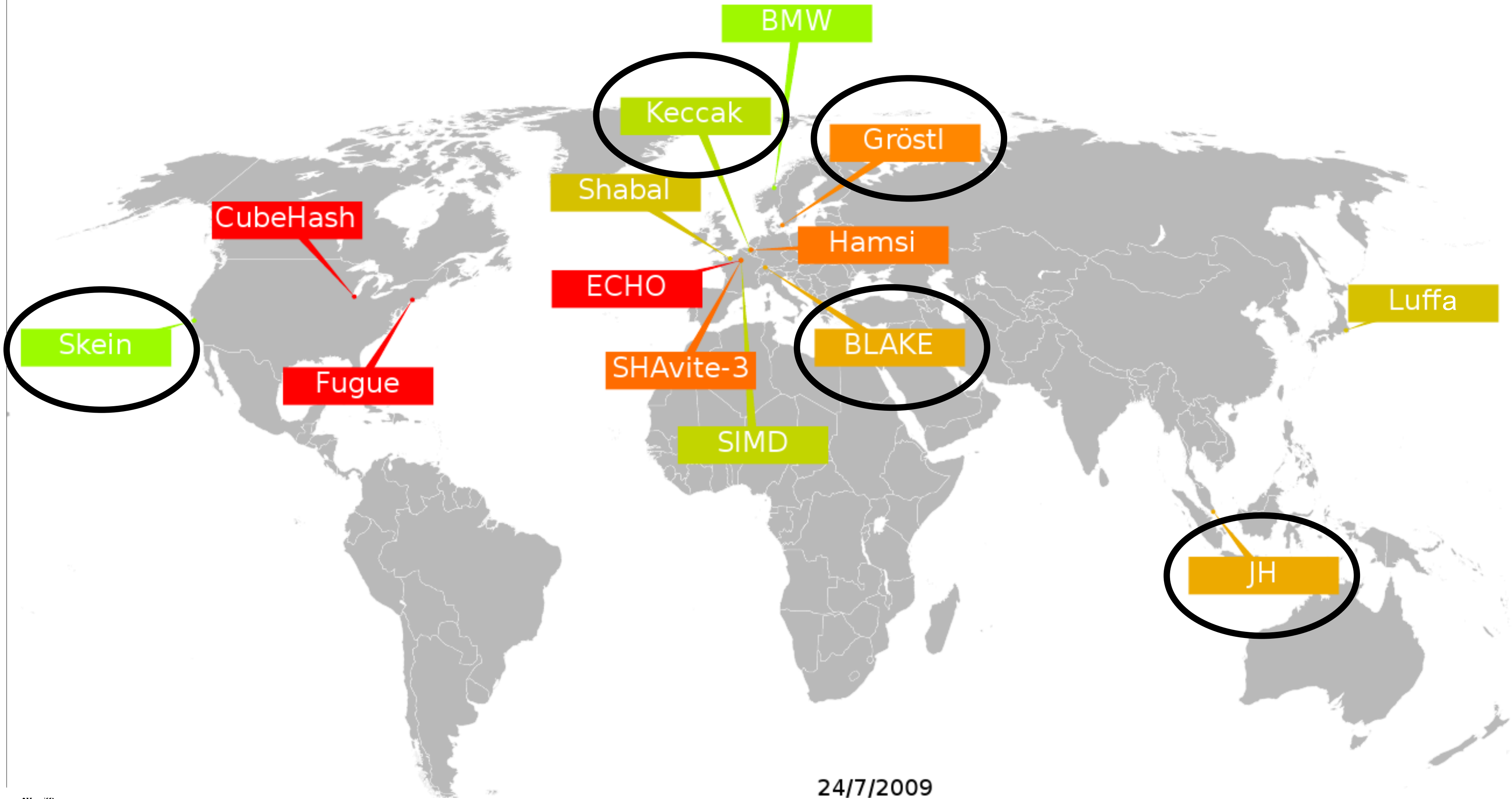
End of Round 1 candidates



8/7/2009

Slide credit: Christophe De Cannière

Round 2 candidates

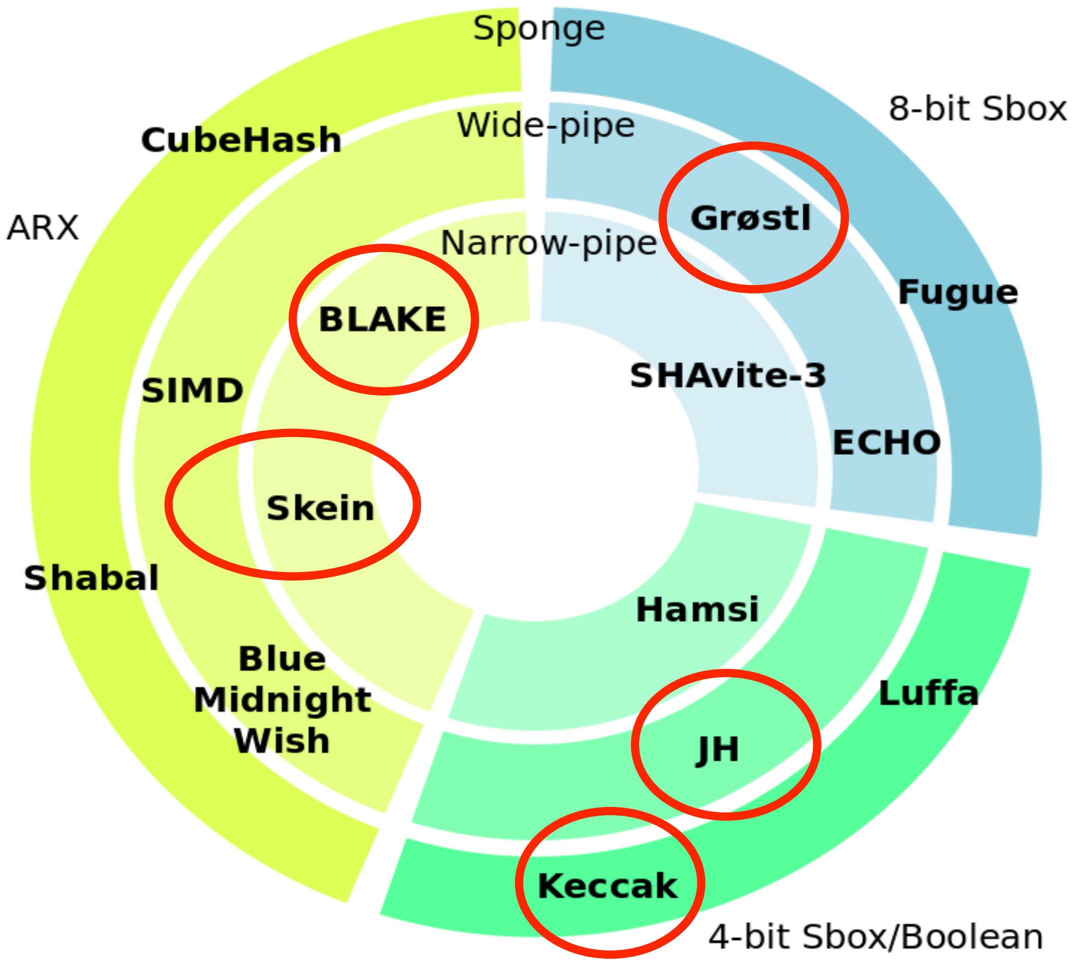


Slide credit: Christophe De Cannière

Compression function/iteration

	Block cipher	Permutation	MD/HAIFA
Blake			HAIFA
Grøstl		2-permutation	MD
JH			JH-specific
Keccak		Sponge	
Skein	MMO		MD*/Tree (UBI)

Properties: bits and bytes
[Watanabe'10]



Security reductions
[Andreeva-Mennink-P'10]

	type	sf	pf	$\mathbf{Adv}_f^{\mathrm{pre}}$	$\mathbf{Adv}_f^{\mathrm{sec}}$	$\mathbf{Adv}_f^{\mathrm{col}}$
BLAKE	HAIFA	✓	✓			
BMW	chop-(MD+FT)	✓	✗			
CubeHash	chop-(MD+FT)	✗	✗			
ECHO	chop-HAIFA	✓	✓			
Fugue	chop-(MD+FT)	✓	✗			
Grøstl	chop-(MD+FT)	✓	✗			
Hamsi	MD+FT	✓	✗			
JH	chop-MD	✓	✗			
Keccak	chop-MD	✗	✗			
Luffa	chop-(MD+FT)	✗	✗			
Shabal	chop-MD	✓	✓			
SHAvite-3	HAIFA	✓	✓			
SIMD	chop-(MD+FT)	✓	✗			
Skein	MD	✓	✓			

Table 1. A schematic summary of all results. The *first* column describes the hash function construction, and the *second* and *third* column show which hash functions have a suffix-free (sf) or prefix-free (pf) padding. A *green* box indicates the existence of a non-trivial upper bound, a *red* box means that an efficient adversary is known for the security notion, and a *yellow* box indicates that no result is known, but recent literature gives some confidence in the existence of a non-trivial bound.

Security: SHA-3 Zoo

http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo

The SHA-3 Zoo - The ECRYPT Hash Function Website - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo

Most Visited Start Bart P DS KLk KLog Kaart wmail MiF IFS DA IC BSCW P FSa mob ECII Win Secr

The SHA-3 Zoo - The ECRYPT Hash Fu...

Log in / create account

article edit history

ECRYPT II

navigation

- The eHash Main Page
- Hash Function Zoo
- SHA-3 Zoo
- Recent changes
- Random page
- Help

search

Go Search

toolbox

- What links here
- Related changes
- Upload file
- Special pages
- Printable version
- Permanent link

The SHA-3 Zoo

The SHA-3 Zoo (work in progress) is a collection of cryptographic hash functions (in alphabetical order) submitted to the SHA-3 contest (see also here). It aims to provide an overview of design and cryptanalysis of all submissions. A list of all SHA-3 submitters is also available. For a software performance related overview, see eBASH. At a separate page, we also collect hardware implementation results of the candidates. Another categorization of the SHA-3 submissions can be found here.

The idea of the SHA-3 Zoo is to give a good overview of cryptanalytic results. We try to avoid additional judgement whether a submission is broken. The answer to this question is left to NIST. However, we categorize the cryptanalytic results by their impact from very theoretic to practical attacks. A detailed description is given in Cryptanalysis Categories.

At this time, 56 out of 64 submissions to the SHA-3 competition are publicly known and available. 51 submissions have advanced to Round 1 and 14 submissions have made it into Round 2.

The following table should give a first impression on the remaining SHA-3 candidates. It shows only the best known attack, more detailed results are collected at the individual hash function pages. A description of the main table is given here.

Recent updates of the SHA-3 Zoo

New: Round 2 tweaks for all candidates

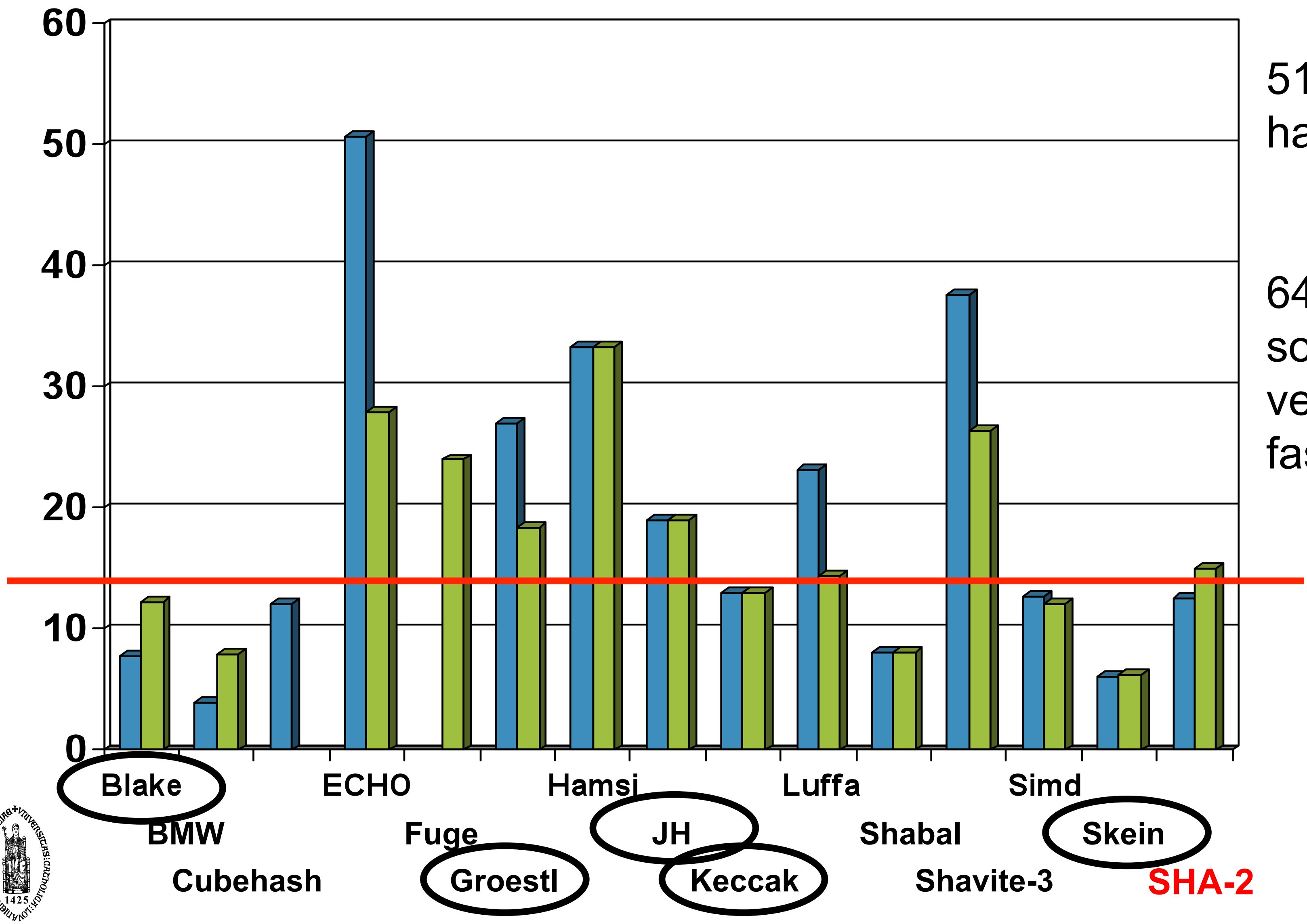
Hash Name	Principal Submitter	Best Attack on Main NIST Requirements	Best Attack on other Hash Requirements
BLAKE	Jean-Philippe Aumasson		
Blue Midnight Wish	Svein Johan Knapskog		
CubeHash	Daniel J. Bernstein	preimage	
ECHO	Henri Gilbert		
Fugue	Charanjit S. Jutla		
Grøstl	Lars R. Knudsen		
Hamsi	Özgül Küçük		

Done

Software performance

[Bernstein-Lange10] http://bench.cr.yp.to/ebash.html

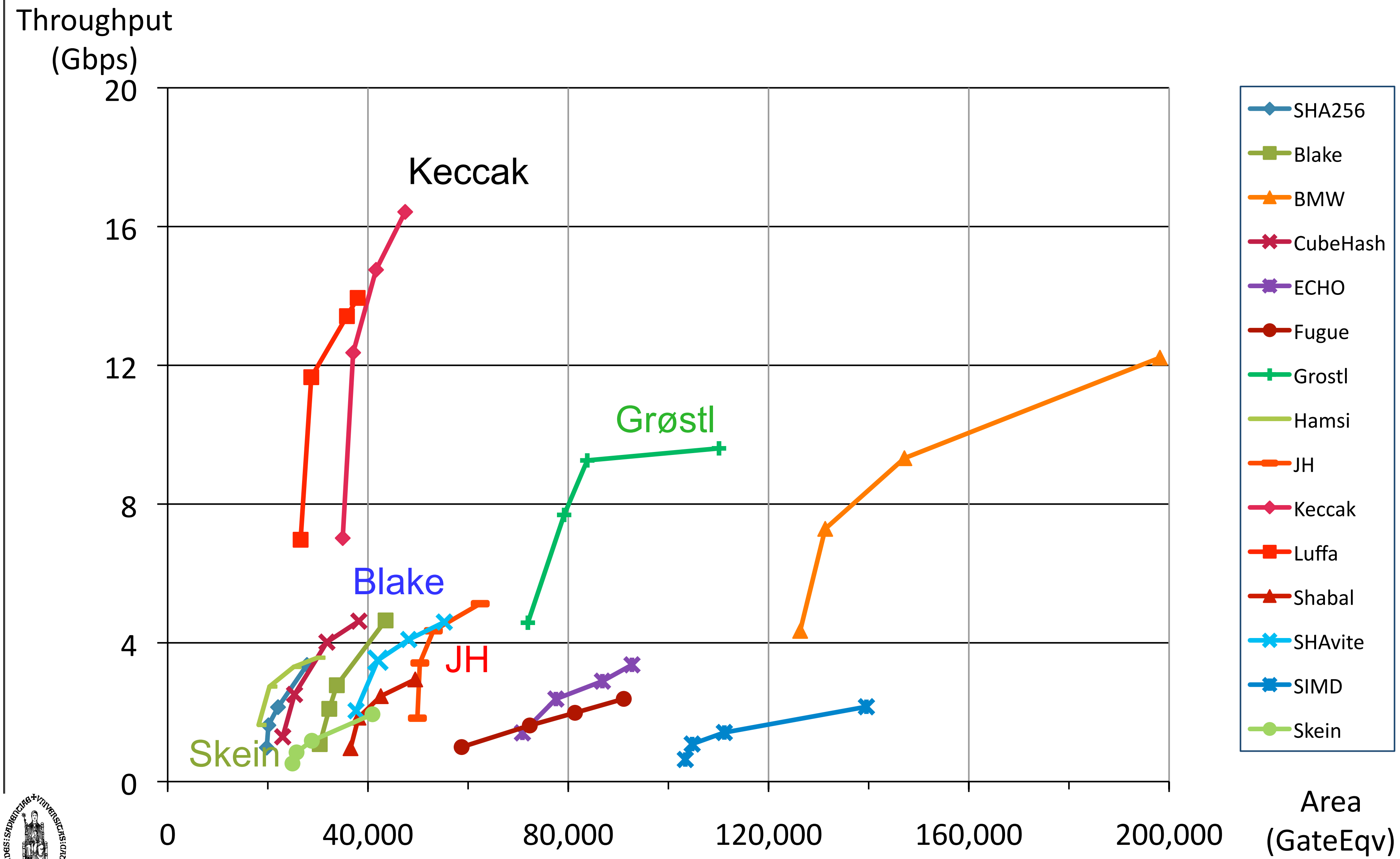
cycles/byte on 3.2 GHz, AMD Phenom II X6 1090T (100fa0)



512/256-bit hash

64-bit machine so 512-bit version is often faster

Hardware: post-place & route results for ASIC 130nm [Guo-Huang-Nazhandali-Schaumont'10]



Issues arisen during Round 1

- round 1 was very short; several functions received no outside analysis
- security
  - some controversy on complexity and relevance of attacks
  - proofs have not helped much to survive
- performance
  - weak performance resulted in elimination

- 7/14 designs tweaked at the beginning of round 2

Issues arisen during Round 2

- security
  - few real attacks but some weaknesses
  - new design ideas harder to validate
- performance: roughly as fast or faster than SHA-2
  - SHA-2 gets faster every day
  - widely different results for hardware and software
    - software: large difference between high end and embedded
    - hardware: FGPA and ASIC
  - what about lightweight devices and 128-core machines?
- diversity = third selection criterion

- expect more tweaks before final
- variable number of rounds?
- NIST expects that SHA-2 and SHA-3 will co-exist

Final

- Blake
- JH
- Grøstl
- Keccak
- Skein

SHA-4?

- an open competition such as SHA-3 is bound to result in new insights between 2008-2012
- only few of these can be incorporated using “tweaks”
- the winner selected in 2012 will reflect the state of the art in October 2008
- nevertheless, it is unlikely that we will have a SHA-4 competition before 2030

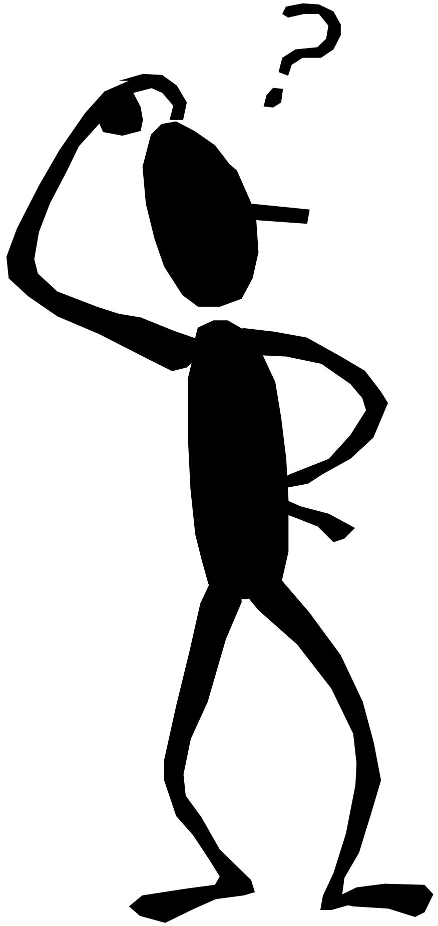
Hash functions: conclusions

- SHA-1 would have needed 128-160 steps instead of 80
- 2004-2009 attacks: cryptographic meltdown but not dramatic for most applications
  - clear warning: upgrade asap
- half-life of a hash function is < 1 year
- theory is developing for more robust iteration modes and extra features; still early for building blocks

- nirwana: efficient hash functions with security reductions

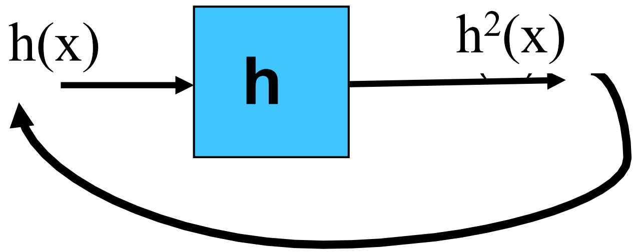
The end

Thank you for your attention



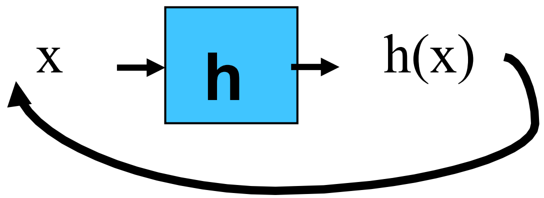
Brute force collision search

- Consider the functional graph of h

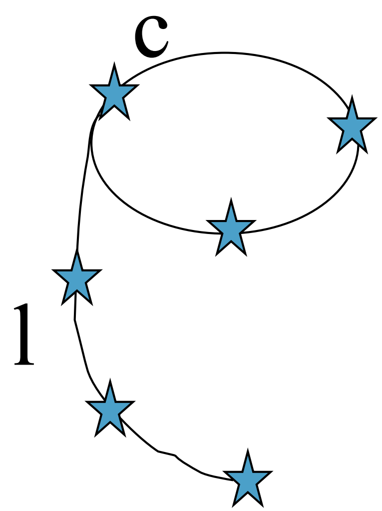


Brute force collision search

- low memory and parallel implementation of the birthday attack
  [Pollard'78][Quisquater'89][Wiener-van Oorschot'94]

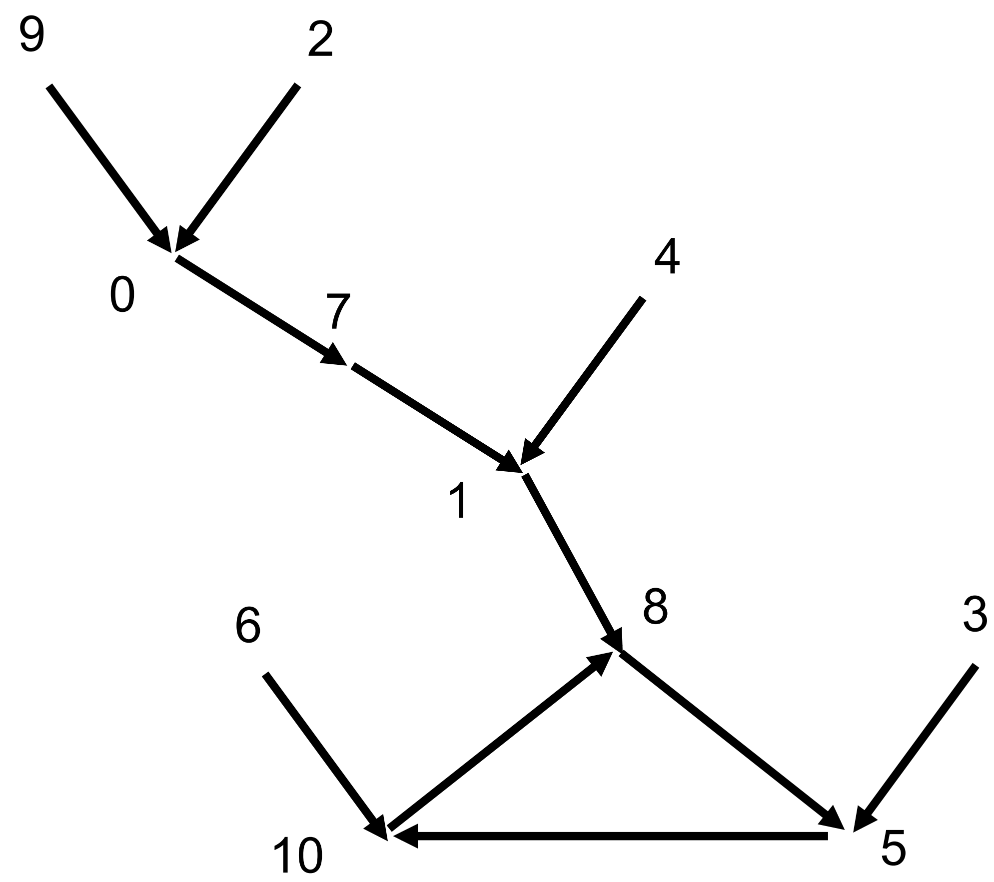


- distinguished point (d bits)
  - $\Theta(e2^{n/2} + e\ 2^{d+1})$ steps with e the cost of one function evaluation
  - $\Theta(n2^{n/2-d})$ memory
  - full cost: $\Theta(e\ n2^{n/2})$ [Wiener'02]



$l = c = (\pi/8)\ 2^{n/2}$

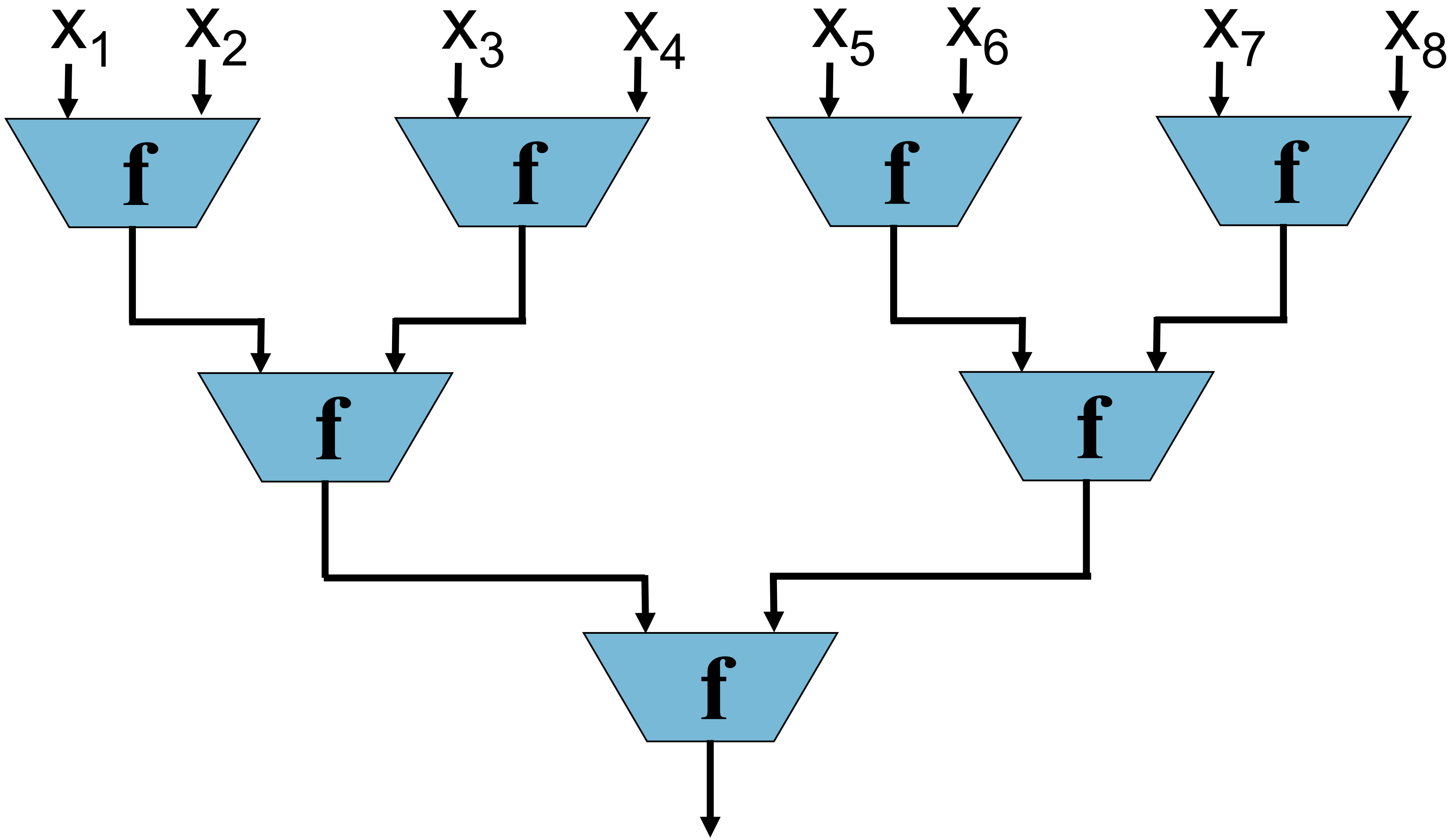
Functional graph of $f(x) = x^2 + 7 \bmod 11$



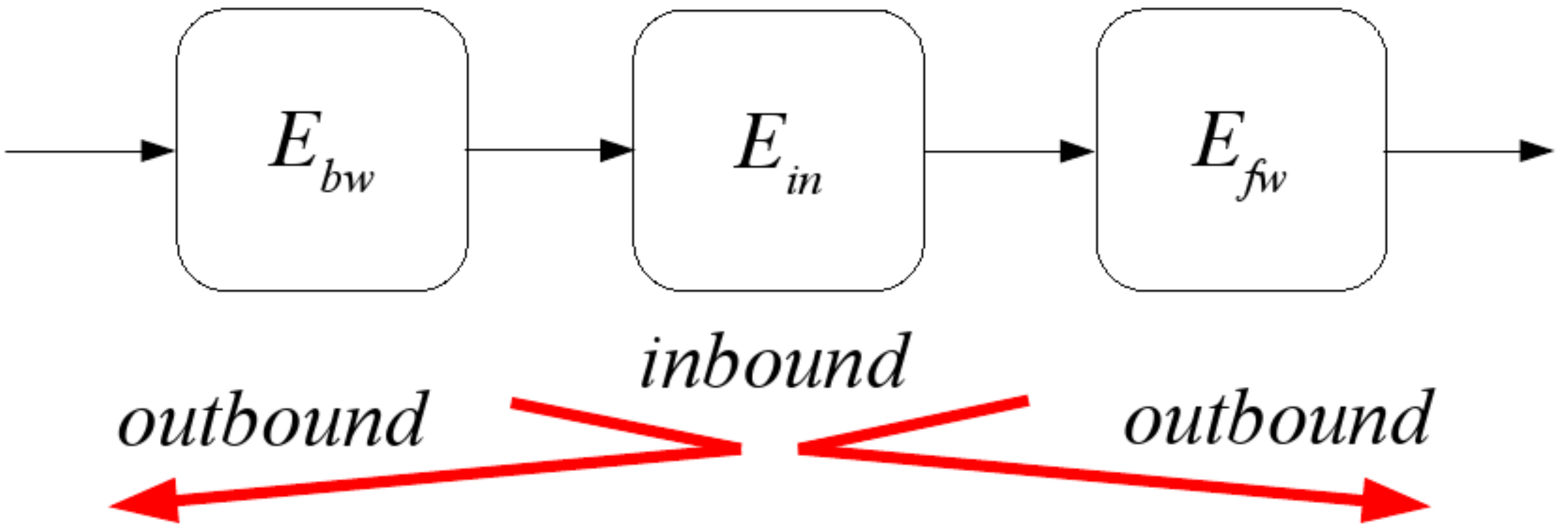
- Exercise: why is the indegree of 5 nodes equal to 0 resp. 2?

Tree structure: parallelism

[Damgård'89], [Pal-Sarkar'03]



a new variant of differential cryptanalysis



developed during the design of Grøstl [MRST09]

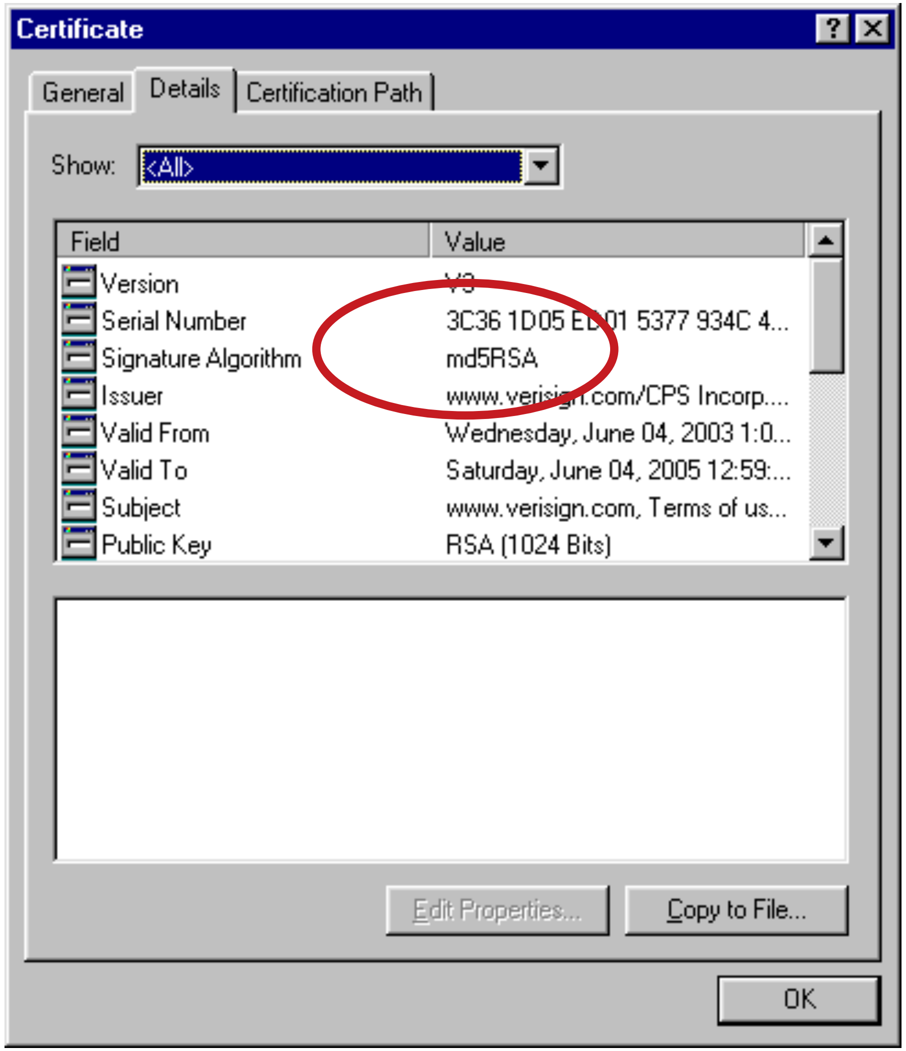
already successfully applied to Whirlpool and the SHA-3 candidates Twister, Lane, and reduced versions of others

Slide credit: Christian Rechberger

MD5 [Rivest'91]

- 4 rounds (64 steps)
- pseudo-collisions [denBoer-Bosselaers'93]
- collisions for compression function [Dobbertin'96]
- collisions for hash function
  - [Wang+'04] – 15 minutes
  - …
  - [Stevens+'09] – milliseconds
  - brute force (2^{64}): 1M$ 8 hours in 2010
- 2^{nd} preimage in 2^{123} [Sasaki-Aoki'09]

- advice (RIPE since ‘92, RSA since ‘96): stop using MD5
- largely ignored by industry until 2009 (click on a cert...)



SHA(-0) [NIST'93]

- now called SHA-0, because of '94 of publication SHA-1
- very similar to MD5:
  - 16 extra steps (from 64 to 80)
  - message expansion uses bitwise code rather than repetition
    $w_j \leftarrow (w_{j-3} \oplus w_{j-8} \oplus w_{j-14} \oplus w_{j-16})\ \ j>15$
  - quasicyclic code with $d_{min} = 23$

- 1994: withdrawn by NIST for unidentified flaw
- 2004: collisions for in 2^{51} [Joux+'04]
- 2005: collisions in 2^{39} [Wang+'05]
- 2007: collisions in 2^{32} [Joux+'07]
- 2008: collisions in 1 hour [Manuel-Peyrin'08]
- 2008: preimages for 52 of 80 steps in $2^{156.6}$ [Aoki-Sasaki'09]

SHA-1 [NIST'95]

- fix to SHA-0
- add rotation to message expansion: quasicyclic code, $d_{min} = 25$
  $w_j \leftarrow (w_{j-3} \oplus w_{j-8} \oplus w_{j-14} \oplus w_{j-16}) >>> 1 \quad j > 15$

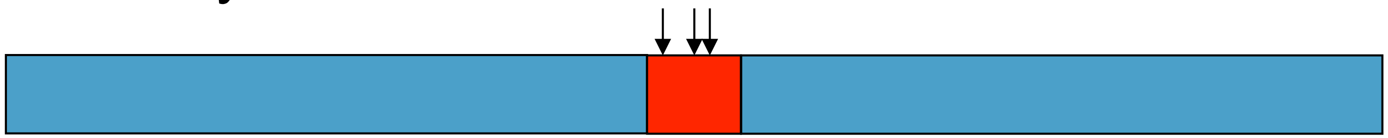
collisions

- 53 steps [Oswald-Rijmen'04 and Biham-Chen'04]
- 58 steps [Wang+'05]
- 64 steps in 2^{35} – highly structured [De Cannière-Rechberger'06-'07]:
- 70 steps in 2^{44} – highly structured [De Cannière-Rechberger'06-'07]:
- 70 steps 2^{39} (4 days on a PC) [Joux-Peyrin'07]
- 2^{69} [Wang+'05]
- 2^{63} ? [Wang+'05 - unpublished]
- 2^{51} ? [Sugita+'06]
- 2^{62} ? [Mendel+'08 - unpublished]
- 2^{52} ?? [McDonald+'09 - unpublished]

preimages for 48/80 steps in $2^{160-\varepsilon}$ [Aoki-Sasaki'09]

Impact of collisions

- collisions for MD5, SHA-0, SHA-1
  - 2 messages differ in a few bits in 1 to 3 512-bit input blocks
  - limited control over message bits in these blocks
  - but arbitrary choice of bits before and after them



- what is achievable for MD5?
  - 2 colliding executables/postscript/gif/…[Lucks-Daum'05]
  - 2 colliding RSA public keys – thus with colliding X.509 certificates [Lenstra+'04]
  - chosen prefix attack: different IDs, same certificate [Stevens+'07]
  - **2 arbitrary colliding files (no constraints) in 8 hours for 1 M$**

Impact of MD5 collisions

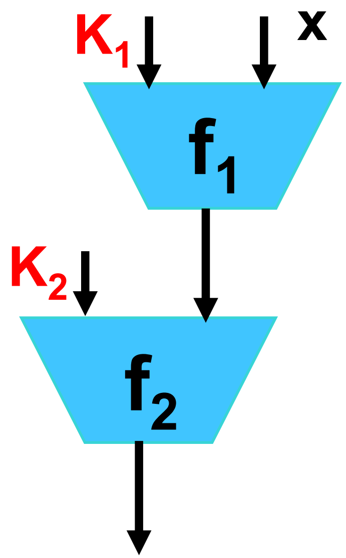
- digital signatures: only an issue if for non-repudiation
- none for signatures computed before attacks were public (1 August 2004)
- ~~none for certificates if public keys are generated at random in a controlled environment~~
- substantial for signatures after 1 August 2005 (cf. traffic tickets in Australia)

And (2^{nd}) preimages?

- security degrades with number of applications
- for large messages even with the number of blocks (cf. supra)
- specific results:
  - MD2: 2^{73} [Knudsen+09]
  - MD4: 2^{102} [Leurent'08]
  - MD5: 2^{123} [Sasaki-Aoki'09]
  - SHA-0: 52 of 80 steps in $2^{156.6}$ [Aoki-Sasaki'09]
  - SHA-1: 48 of 80 steps in $2^{159.3}$ [Aoki-Sasaki'09]

- HMAC keys through the IV (plaintext)
  - collisions for MD5 invalidate current security proof of HMAC-MD5

	Rounds in f2	Rounds in f1	Data complexity
MD4	48	48	2^{72} CP + 2^{77} time
MD5	64	33 of 64	$2^{126.1}$ CP
MD5	64	64	2^{51} CP & 2^{100} time (RK)
SHA-0	80	80	2^{109} CP
SHA-1	80	53 of 80	$2^{98.5}$ CP



SWIFFTX

[Arbitman-Dogon-Lyubashevsky-Micciancio- Peikert-Rosen'08]

- compression function:
  - SWIFFT: FFT-like operation from $(Z_2^{32})^{64}$ to Z_{257}^{64}
  - sandwich: 3xSWIFFT - S-boxes - 1xSWIFFT
- asymptotic proof of security: "it can be formally proved that finding a collision in a randomly-chosen compression function from the SWIFFTX family is at least as hard as finding short vectors in cyclic/ideal lattices over the ring $Z[\alpha]/(\alpha^n+1)$ is in the *worst case*."
- note: SWIFFT mapping is linear and some heuristics are needed to "kill" the linearity
- speed: 57 cpb

FSB [Augot-Finiasz-Gaborit-Manuel-Sendrier'08]

- compression function: multiplication of vector of Hamming weight w with a truncated quasi-cyclic binary matrix
  - can be interpreted as a syndrome computation of an error pattern with weight w
- MD iteration with Whirlpool as output transformation
- security can be reduced to:

(Computational Syndrome Decoding) *Given a binary r x n matrix H, a word $s \in \{0,1\}^r$ and an integer $w > 0$, find a word $e \in \{0,1\}^n$ of Hamming weight $\leq w$ such that $eH^T = s$.*

(Codeword Finding) *Given a binary r x n matrix H and an integer $w > 0$, and a non-zero word $e \in \{0,1\}^n$ of Hamming weight $\leq w$ with an all zero H-syndrome.*

- 324 cpb (can be optimized)

ZesT: a SHA-4 candidate?

- Zémor-Tillich: consider the 2 generators of the group $SL(2; F_{2^n})$

$$A_0 = \begin{pmatrix} x & 1 \\ 1 & 0 \end{pmatrix} \qquad A_1 = \begin{pmatrix} x & x+1 \\ 1 & 1 \end{pmatrix}$$

  the hash value of a string x with elements x[i] is $\prod_{i=1}^{n} A_{x[i]}$
- ZesT = vectorial version of the Zémor-Tillich function iterated 2x
- security: ZesT is collision resistant if and only if the balance problem is hard and in particular if the representation problem is hard for the group $SL(2; F_{2^n})$ and the generators A_0 and A_1
- performance: 10-20 times slower than SHA-512 but parallelism

More details: PhD thesis of Christophe Petit, UCL, May 2009

Original ZT scheme broken in 2009
see IACR eprint [Grassl-Ilic-Magliveras-Steinwandt'09]