The State of Hash Functions and the NIST SHA-3 Competition

Bart Preneel

COSIC – Katholieke Universiteit Leuven, Belgium

bart.preneel(AT)esat.kuleuven.be

February 2009

Outline

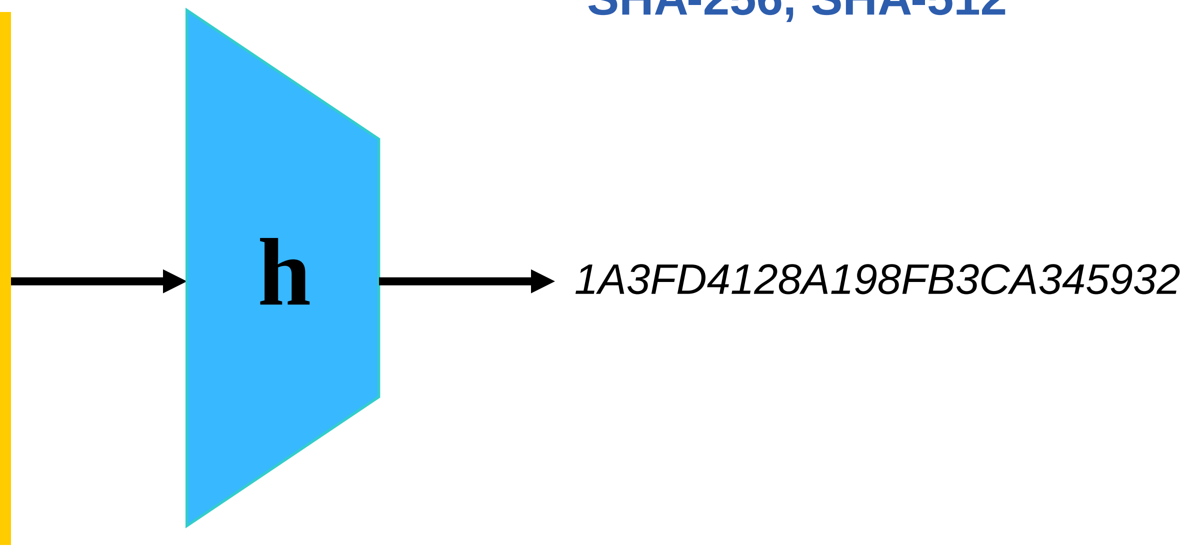
- definitions
- applications
- generic attacks
- attacks on iterated constructions
- attacks on custom designed hash functions: SHA-1
- the NIST AHS competition (SHA-3)
- conclusions

Hash functions

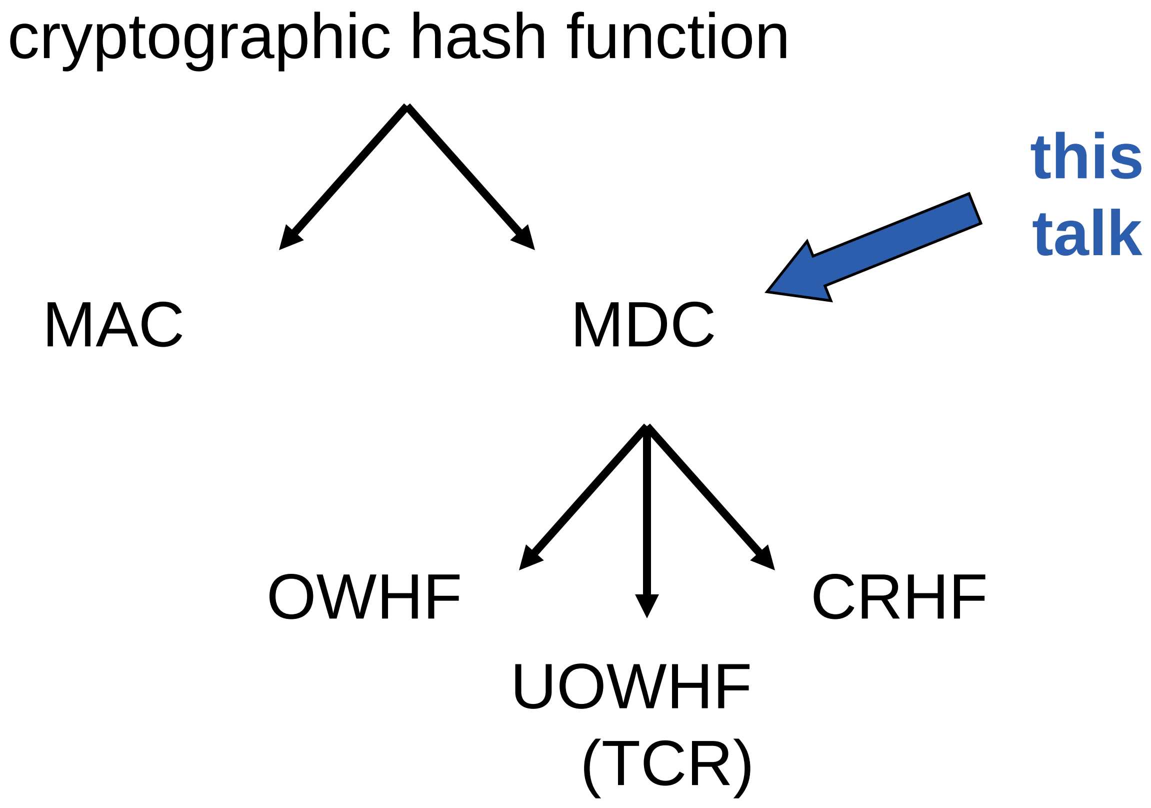
- MDC (manipulation detection code)
- Protect short hash value rather than long text

- **(MDC-2)**
- **(MD5)**
- **(SHA-1)**
- **RIPEMD-160**
- **SHA-256, SHA-512**

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).



Hash function flavours

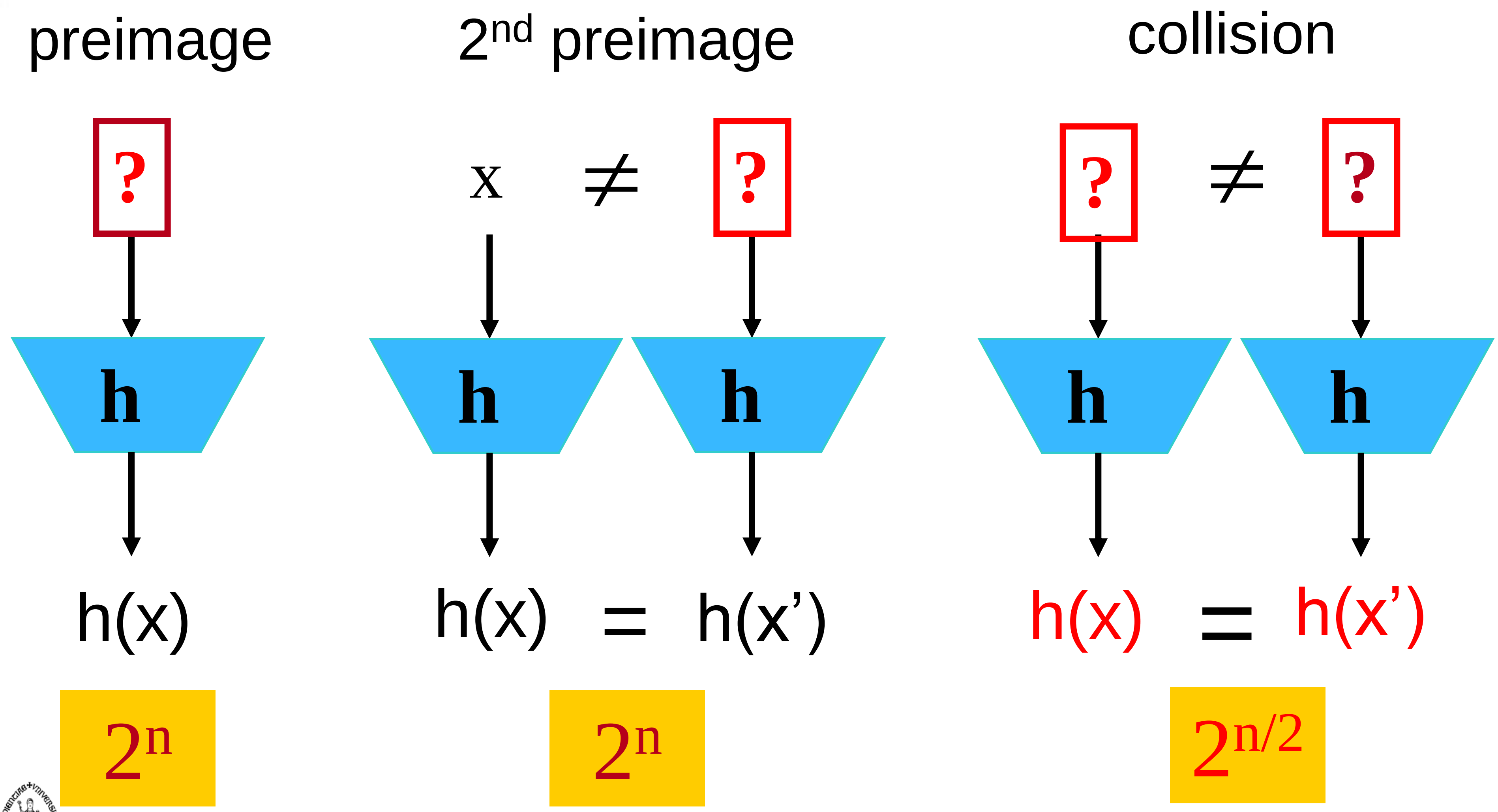


Informal definitions (1)

- no secret parameters
- input string x of arbitrary length $\Rightarrow$ output h(x) of fixed bitlength n
- computation "easy"

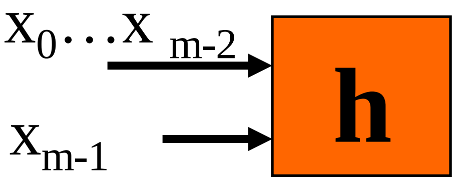
- One Way Hash Function (OWHF)
  - preimage resistance
  - 2^{nd} preimage resistance
- Collision Resistant Hash Function (CRHF): OWHF +
  - collision resistant

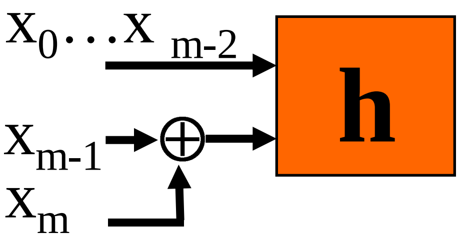
Security requirements (n-bit result)



Informal definitions (2)

- preimage resistant $\not\Rightarrow$ 2^{nd} preimage resistant
  - take a preimage resistant hash function; add an input bit b and replace one input bit by the sum modulo 2 of this input bit and b





- 2nd preimage resistant $\not\Rightarrow$ preimage resistant
  - if h is OWHF, $\underline{h}$ is 2nd preimage resistant but not preimage resistant:

$$\underline{h}(x) = \begin{cases} 0 \,\|\, x & \text{if } |x| \le n \\ 1 \,\|\, h(X) & \text{otherwise} \end{cases}$$

- collision resistant $\Rightarrow$ 2nd preimage resistant

- [Simon'98] one cannot derive collision resistance from "general" preimage resistance (there exists no black box reduction)

Applications

- digital signatures: OWHF/CRHF, `destroy algebraic structure‘
- information authentication: protect authenticity of hash result
- protection of passwords: preimage resistant
- confirmation of knowledge/commitment: OWHF/CRHF
- pseudo-random string generation/key derivation
- micropayments (e.g., micromint)
- construction of MAC algorithms, stream ciphers, block ciphers
- (redundancy: hash result appended to data before encryption)

Applications (2)

- Collision resistance is not always necessary
- Other properties are needed:
  - pseudo-randomness if keyed (with secret key)
  - near-collision resistance
  - partial preimage resistance
  - multiplication freeness
  - random oracle property
- how to formalize these requirements and the relation between them?

Brute force (2nd) preimage

- If one can attack 2^t simultaneous targets, the effort to find a single preimage is 2^{n-t}
  - note for $t = n/2$ this is $2^{n/2}$
- [Hellman'80] if one has to find (second) preimages for many targets, one can use a time-memory trade-off with $\Theta(2^n)$ precomputation and storage $\Theta(2^{2n/3})$
  - inversion of one message in time $\Theta(2^{2n/3})$
- [Wiener'02] if $\Theta(2^{3n/5})$ targets are attacked, the full cost per (2nd) preimage decreases from $\Theta(2^n)$ to $\Theta(2^{2n/5})$
- answer: randomize hash function
  - salt, spice, "key": parameter to index family of functions

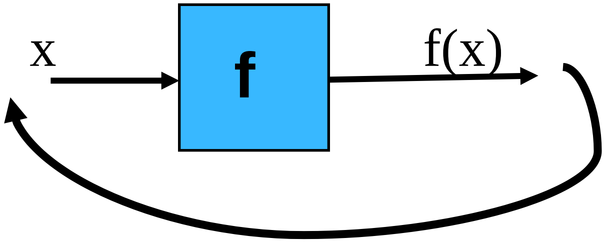
The birthday paradox for collisions

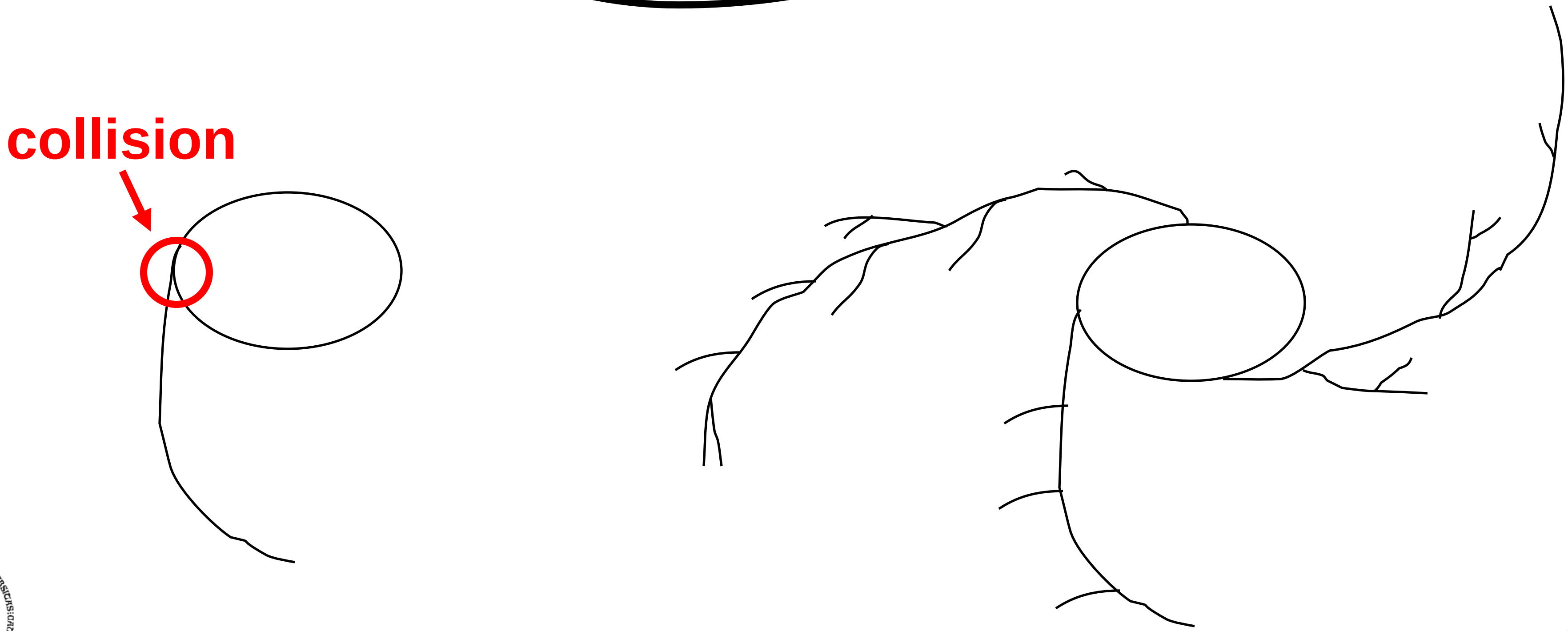
- Given a set with S elements
- Choose r elements at random (with replacements) with r « S
- The probability p that there are at least 2 equal elements (a collision) is 1 - exp $(- r(r-1)/2S)$

- S large, $r = \sqrt{S}$, $p = 0.39$: finding a collision takes computation and memory $\sqrt{S}$
  - for birthdays: $S = 365$, $r = 23$, $p = 0.50$

Brute force collision search: low memory

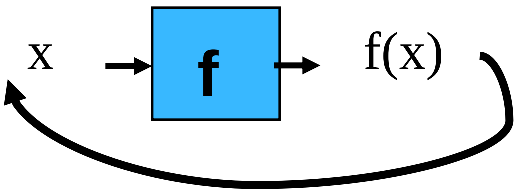
- Consider the functional graph of f

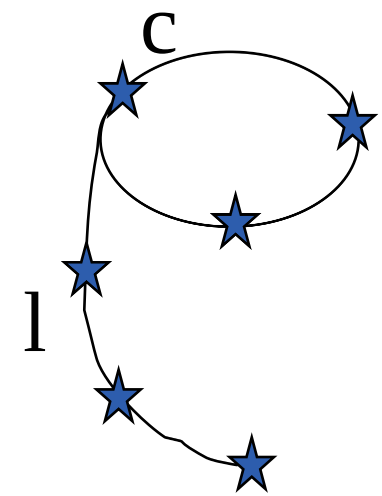




Brute force collision search: low memory

- Efficient implementation of the birthday attack [Pollard'78][Quisquater'89]
  - very little memory (cycle finding algorithm)
  - full parallelism [Wiener-van Oorschot'94]
- Distinguished point (d bits)
  - $\Theta(e2^{n/2} + e\ 2^{d+1})$ steps
  - $\Theta(n2^{n/2-d})$ memory
  - with e the cost of one function evaluation
- [Wiener'02] full cost: $\Theta(e\ n2^{n/2})$





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

Brute force attacks in practice

- (2^{nd}) preimage search
  - n = 128: 60 M$ for 1 year if one can attack $\mathbf{2^{48}}$ targets in parallel
  - n = 128: 60 B$ for 1 year if one can attack $\mathbf{2^{38}}$ targets in parallel
- parallel collision search
  - n = 128: 15 K$ for 10 days
  - n = 160: 60 M$ for 4 months
  - need 256-bit result for long term security (25 years or more)

Can we get rid of collision resistance?

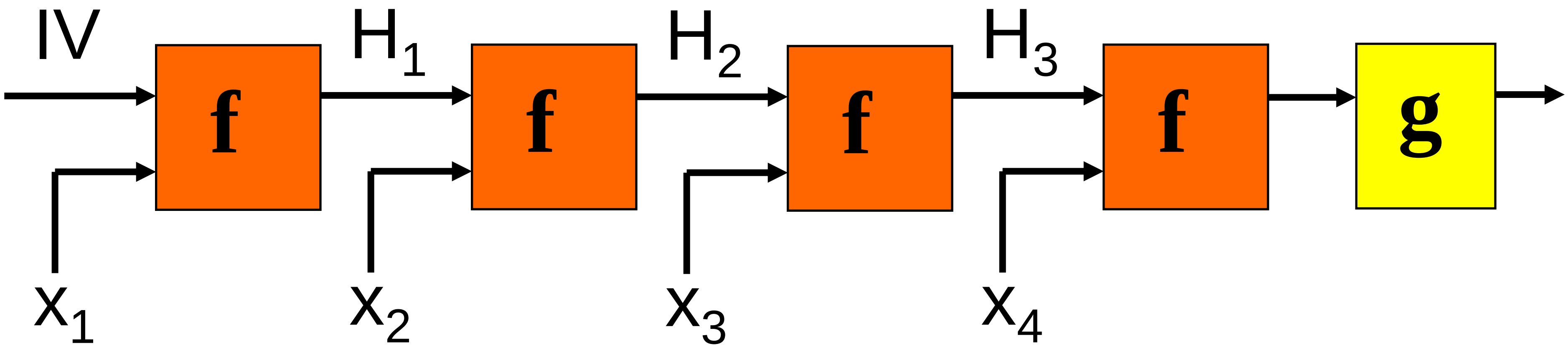
- collision resistance
  - requires double output lengths
  - requires family of functions for formalization
  - is hard to achieve (e.g., not by black box reduction from one-wayness)

- UOWHF (TCR, eSec) randomize hash function after choosing the message
- [Halevi-Krawczyk'05] randomized hashing = RMX mode:

$$H(r \,||\, x_1 \oplus r \,||\, x_2 \oplus r \,||\, \ldots \,||\, x_t \oplus r)$$

  - needs e-SPR (not met by MD5 and SHA-1 reduced to 53 rounds)
  - issues with insider attacks (i.e. attacks by the signer)

Hash function: iterated structure

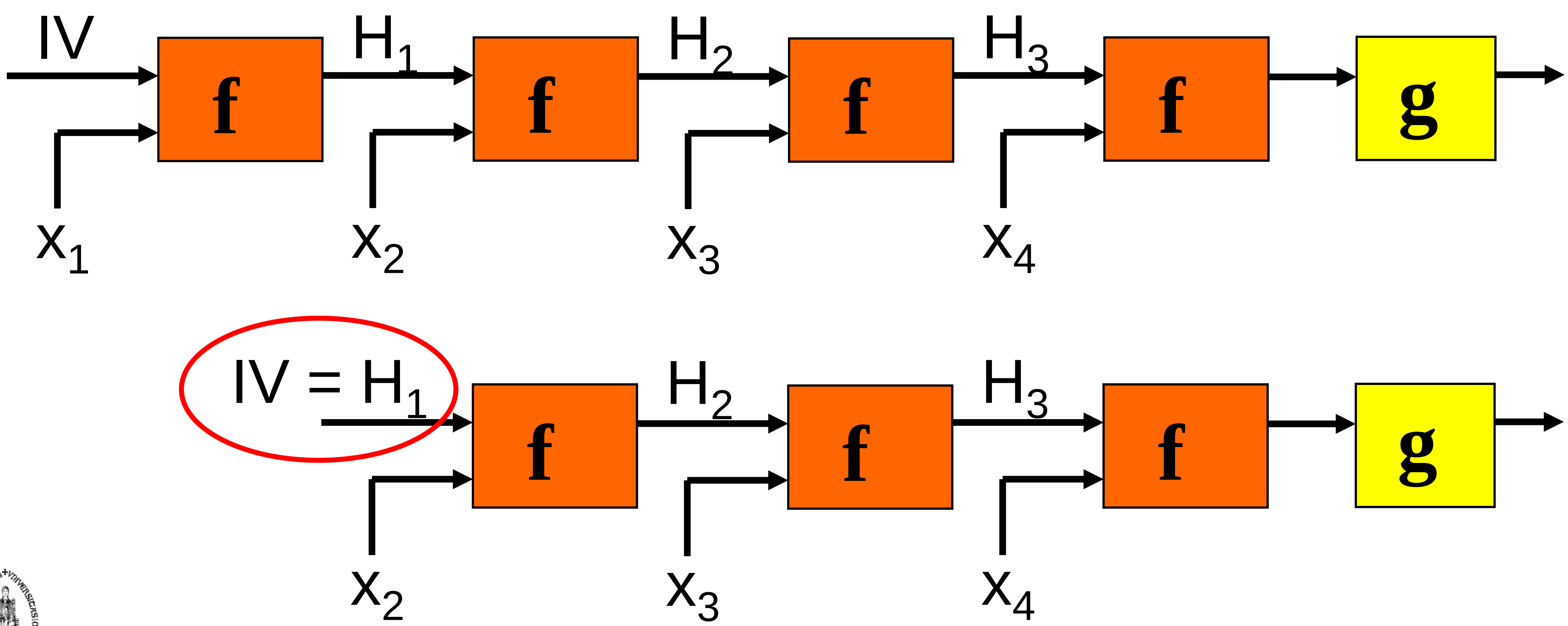


Split messages into blocks of fixed length and hash them block by block with a compression function f

Efficient and elegant
But many problems…

Security relation between f and h

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



Security relation between f and h

- Solution: Merkle-Damgard (MD) strengthening (popular!)
  fix IV, use unambiguous padding and insert length at the end

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

Construction: relation between f and h (2)

[Damgård-Merkle'89]

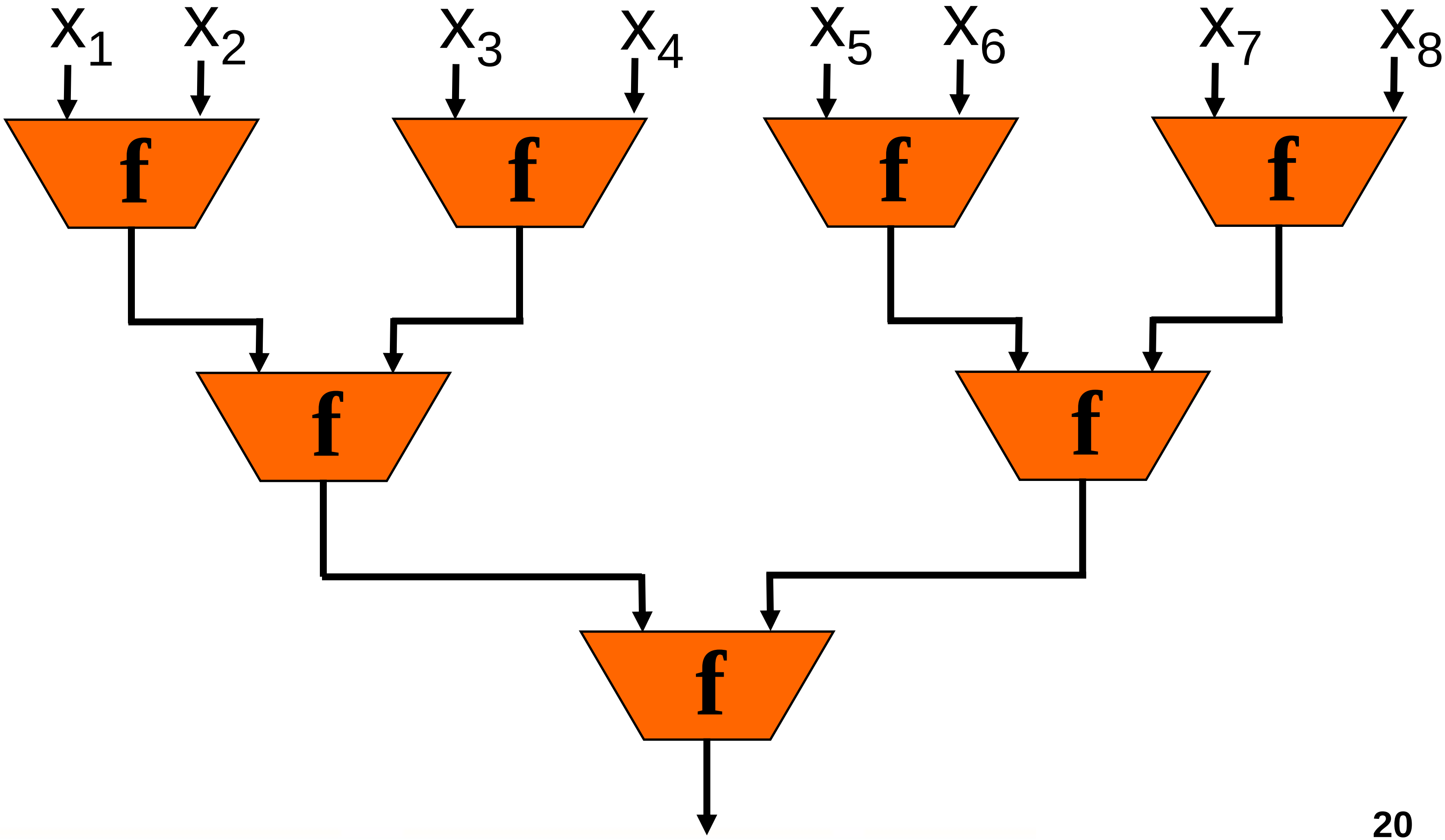
Let *f* be a collision resistant function mapping *l* to *n* bits (with $l > n$).

- If the padding contains the length of the input string, and if *f* is preimage resistant, the iterated hash function *h* based on *f* will be a CRHF.
- If an unambiguous padding rule is used, the following construction will yield a CRHF ($l-n>1$):

$$H_1 = f(H_0 \,||\, 0 \,||\, x_1)$$
$$H_i = f(H_{i-1} \,||\, 1 \,||\, x_i) \quad i=2,3,\ldots t.$$

Comment: tree structure

already suggested by Damgård in 1989; further work by Sarkar et al.



Construction: relation between f and h (3)

[Lai-Massey'92]

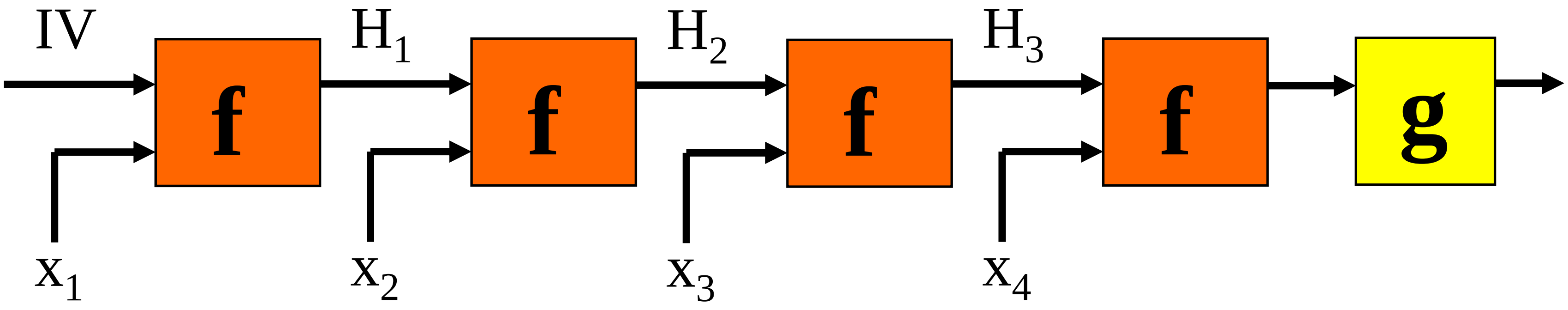
Assume that the padding contains the length of the input string, and that the message x (without padding) contains at least two blocks.

Then finding a second preimage for h with a fixed IV requires 2^n operations **iff** finding a second preimage for f with arbitrarily chosen H_{i-1} requires 2^n operations.

- this theorem is not quite right (see below)
- very few hash functions have a strong compression function
- very few hash functions are designed based on a strong compression function in the sense that they treat x_i and H_{i-1} in the *same way*.

Security relation between f and h (4)

- MD does **not** work for UOWHF [BellareRogaway'97]
- MD with envelope method (prepend and append secret key) works for pseudo-randomness/MAC [BCK'96]
  - but there are some problems and HMAC is a better construction
- MD needs output transformation for random oracle properties [Coron+05]
  - if one knows *h(x),* easy to compute *h(x || y)* without knowing *x*



Attacks on MD

- Long message 2nd preimage attack
- Multi-collision attack and impact on concatenation
- Herding attack

Defeating MD for 2^{nd} preimages

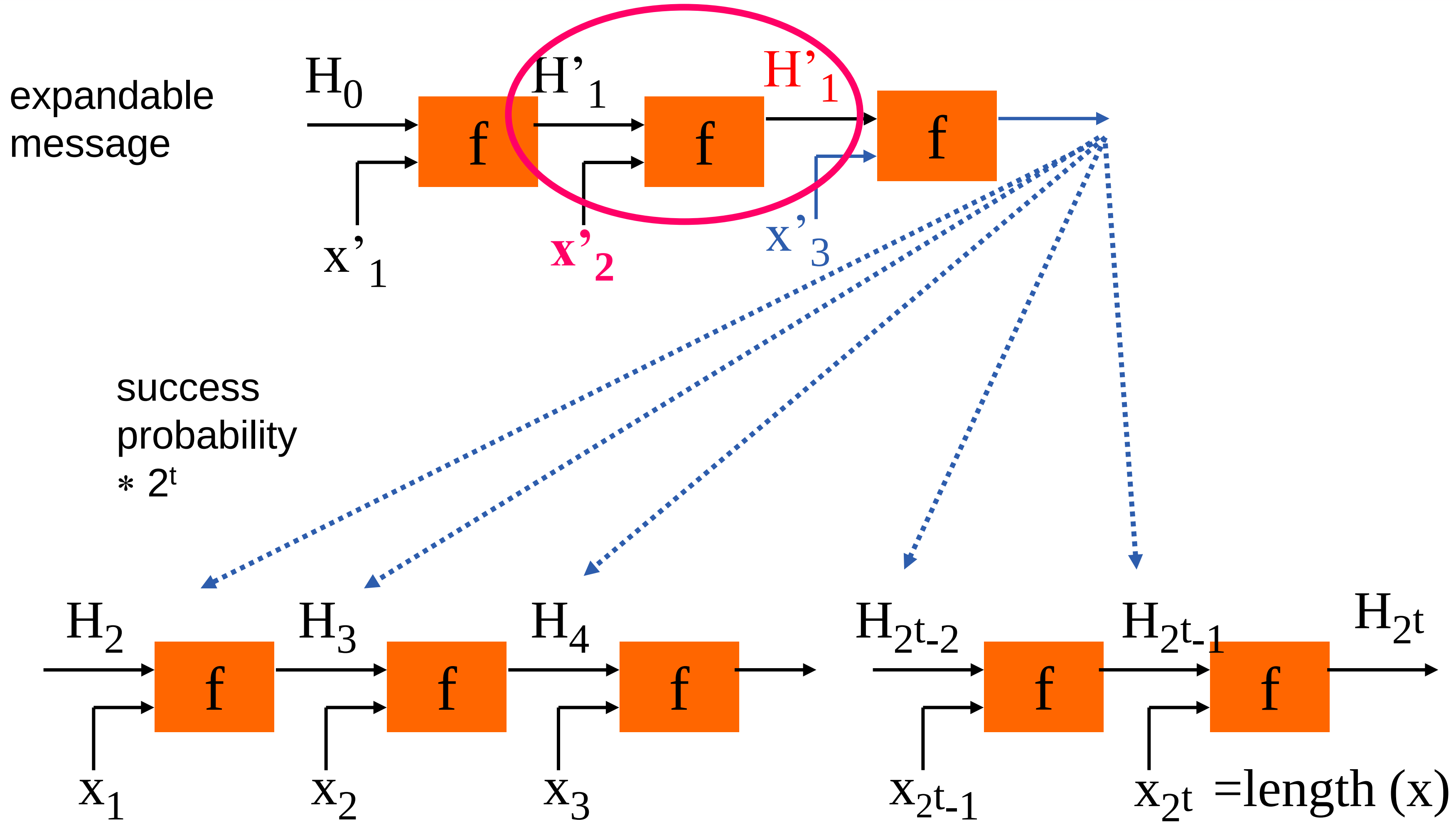
[Dean-Felten-Hu'99] and [Kelsey-Schneier'05]

[Merkle'79]: if one hashes **2^t messages**, the average effort to find a second preimage for one of them is 2^{n-t}

New: if one hashes **2^t message blocks** with an iterated hash function, the effort to find a second preimage is only $2^{n-t+1} + t\, 2^{n/2+1}$

- idea: create expandable message using fixed points
  - Finding fixed points can be easy (e.g., Davies-Meyer)
- find 2^{nd} preimage that hits any of the 2^t chaining values in the calculation
- stretch the expandable message to match the length (and thus the length field)
- But still very long messages for attack to be meaningful
  - n=128, t=32, complexity reduced from 2^{128} to 2^{97}, length is 256 Gbyte

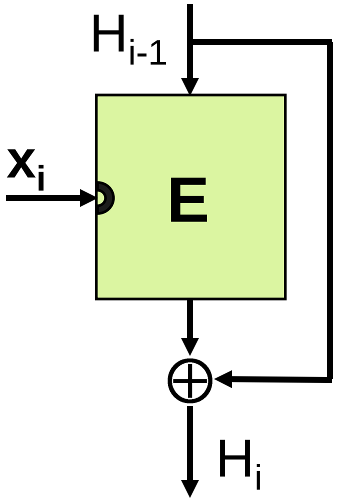
Defeating MD for 2nd preimages (2)



$h(x'_1 \| x'_2 \| x'_2 \| x'_2 \| x'_2 \| x'_3 \|\ldots\| x_{2t-1} \| x_{2t}) = h(x_1 \| x_2 \| x_3 \|\ldots\| x_{2t-1} \| x_{2t})$

How to find fix points?

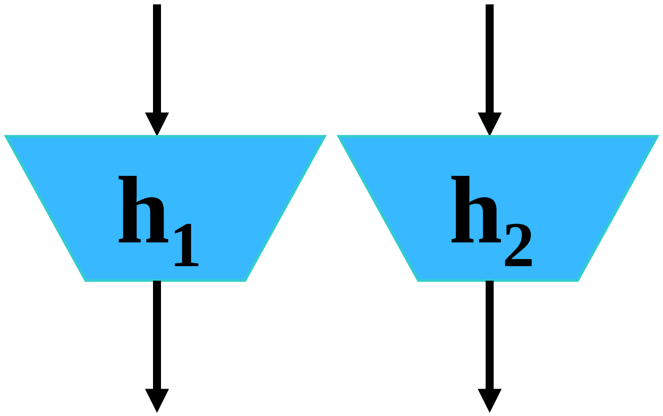
- Davies-Meier: $E_{x_i}(H_{i-1}) \oplus H_{i-1}$
- Fix point $H_{i-1} = D_{x_i}(0)$ for any x_i
  - Proof: $E_{x_i}(H_{i-1}) \oplus H_{i-1} = H_{i-1}$ implies $E_{x_i}(H_{i-1}) = 0$



- Expandable message using meet-in-the-middle
  - Generate $2^{n/2}$ values x_2 and compute $H_1 = D_{x_2}(0)$
  - Generate $2^{n/2}$ values x_1 and compute $H_1 = E_{x_1}(H_0) \oplus H_0$
  - Find a match with high probability
- For non-Davies-Meier: use the trick of Joux

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….

Multi-collisions [Joux '04]

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

The concatenation of two 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

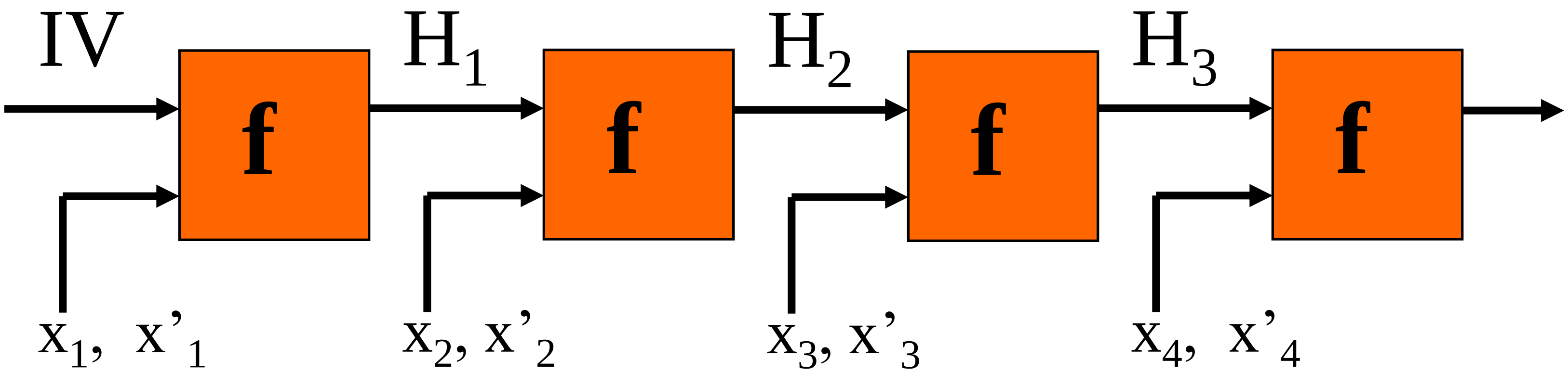
  $$n1 \cdot 2^{n2/2} + 2^{n1/2} \ll 2^{(n1 + n2)/2}$$

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

  $$n1 \cdot 2^{n2/2} + 2^{n1} + 2^{n2} \ll 2^{n1 + n2}$$

- If either of the functions is weak, the attacks may work better.
- Main observation: finding multiple 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)

Multi-collisions (2) [Joux '04]



- 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**

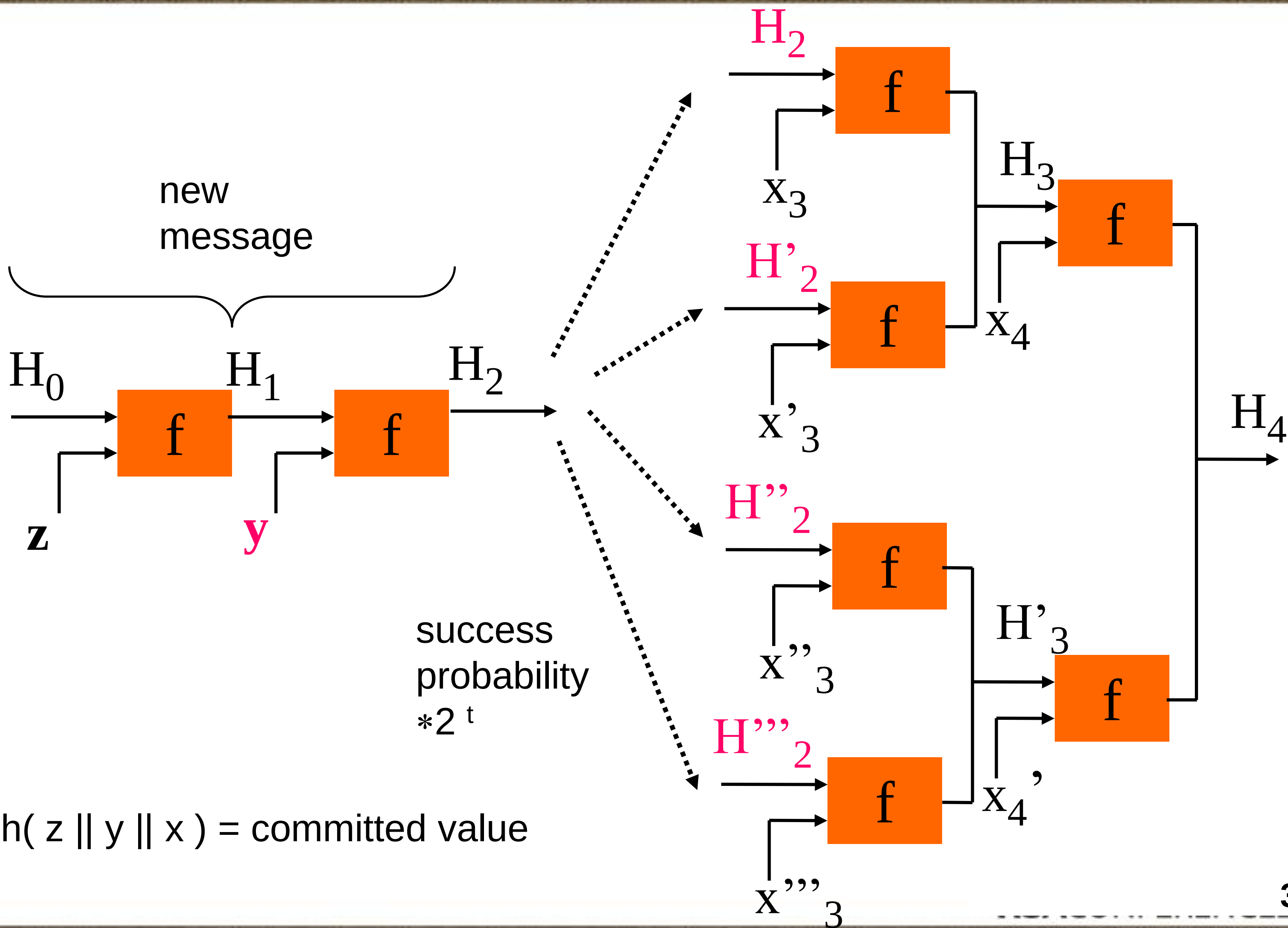
Other issues with MD iteration: herding

- Herding attack [Kelsey,Kohno'06]
  - reduces security of commitment using a hash function
  - on-line 2^{n-t} + precomputation $2.2^{(n+t)/2}$ + storage 2^t
  - example (n=128, t=42): with a storage of 100 Terabyte and a precomputation of 2^{86} steps, a 128-bit commitment computed using an iterated hash function can be spoofed with effort 2^{86} steps

Herding attack (2)

- protocol: publish $h(x)$, reveal x at later date
- find second preimage $x' = z \,||\, y \,||\, x$ with z and y selected in 2020
- approach: generate collision tree (diamond structure) of 2^t values H_{i-1} and x_i hashing to the same value (cost $2 \cdot 2^{t/2} \cdot 2^{n/2}$)
  - work factor for first layer: $x^2/2^{n+1} = 2^t$ or $x = \sqrt{2} \cdot 2^{t/2} \cdot 2^{n/2}$
- z = result of all Champions League finals between 2010 and 2020
- try in 2020 random strings y until $h(z \,||\, y) = H_{j-1}$ for some j (cost 2^{n-t})
- then $h(z \,||\, y \,||\, x_j) = h(x)$, so you can claim that you "knew" z in 2008

Herding attack (3)



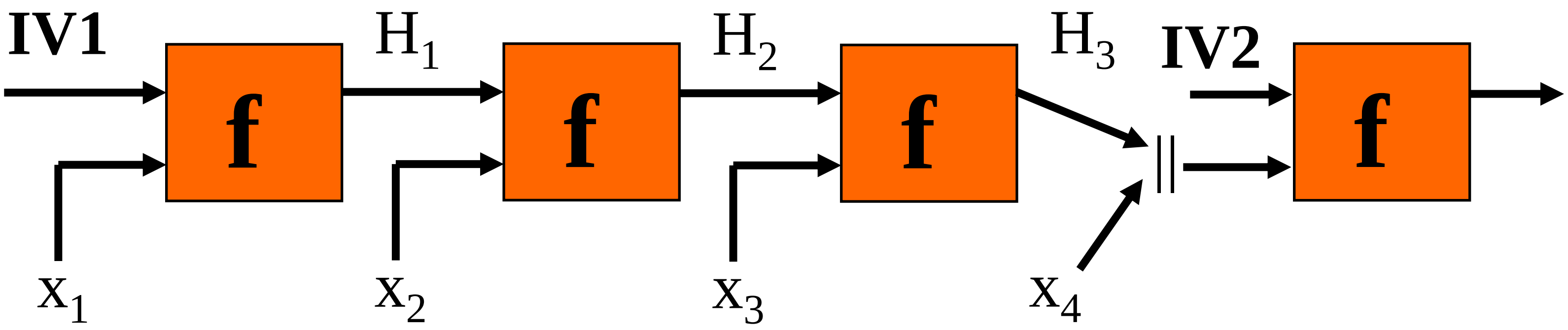
h(z || y || x) = committed value

Improving MD iteration

- degradation with use: salting (family of functions, randomization)
- extension attack: strong output transformation g (which includes total length and salt)
- long message 2nd preimage: preclude fix points
  - counter $f \rightarrow f_i$ [Biham-Dunkelman] or dithering [Rivest]
- multi-collisions, herding: avoid breakdown at $2^{n/2}$ with larger internal memory: known as wide pipe
  - e.g., extended MD4, RIPEMD, [Lucks'05]

Many more ideas….

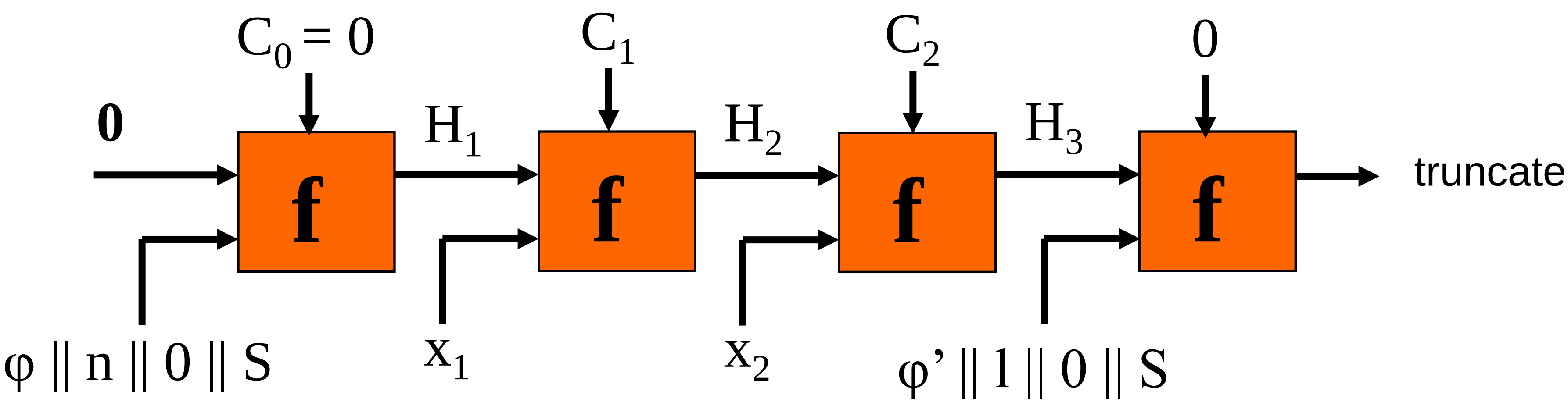
- [Biham-Dunkelman'06] Haifa: bit counter and salt input to f
- [Bellare-Ristenpart'06] EMD transform (envelope MD): preserves CR, PRF, RO



- [Andreeva+'06] analysis of preservation of CR, (e/a/-)PR, (e/a/-) SPR, (RO, PRF)

Many more ideas….

- LANE (SHA-3 submission by [Indesteege+], COSIC



C_i number of bits hashed so far

φ flag that indicates presence/absence of salt S

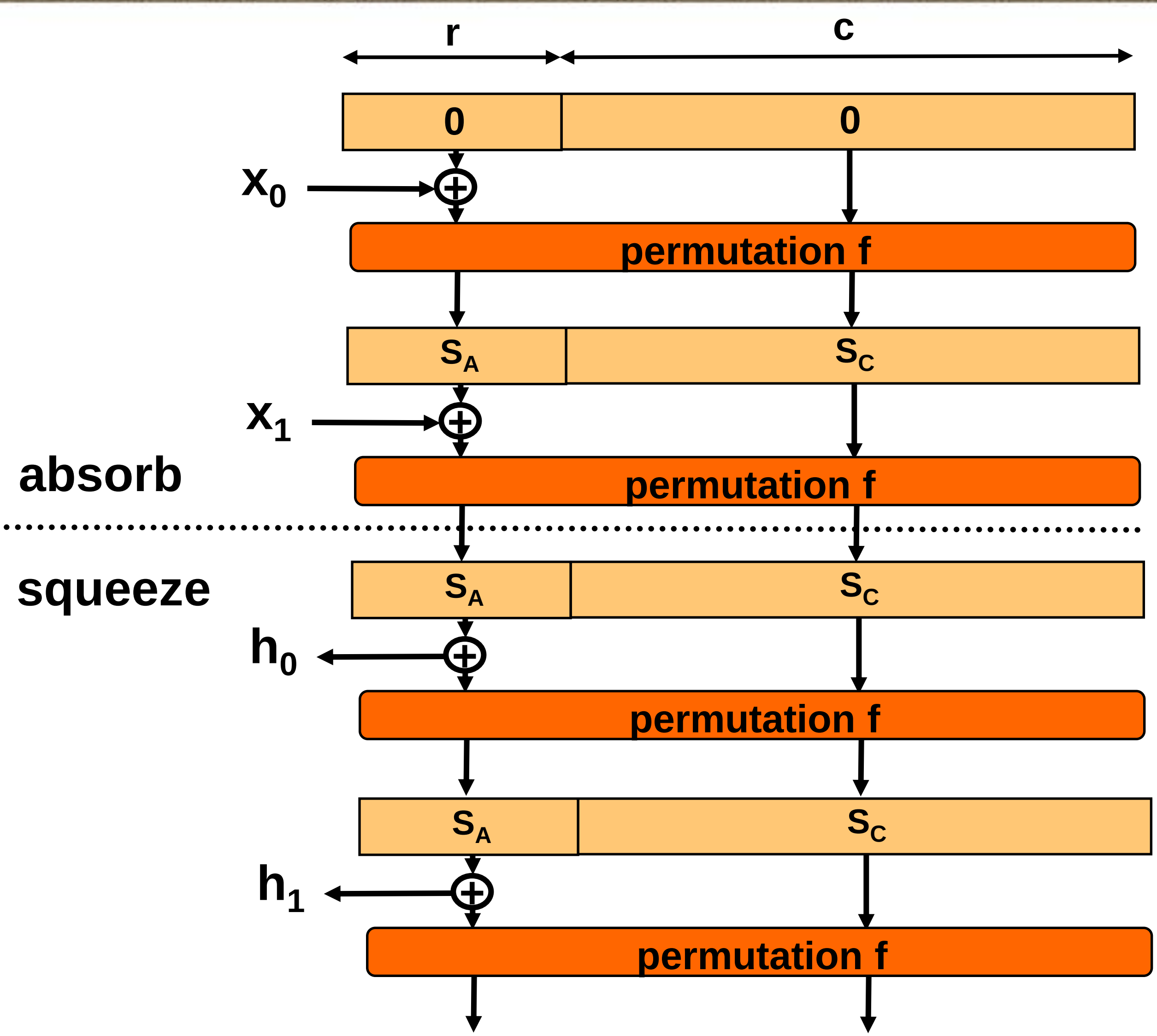
n output length

l total message length in bits

Sponge functions

Examples

- Panama
- RadioGatun
- Grihndahl
- Keccak



Outline

- definitions
- applications
- generic attacks
- attacks on iterated constructions
- attacks on custom designed hash functions: SHA-1
- the NIST AHS competition (SHA-3)
- conclusions

Hash function constructions

block cipher based

- well studied but need very strong assumption on block cipher
- due to key schedule for every encryption at least 3-4 times slower than AES
- 30 proposals, more than half broken

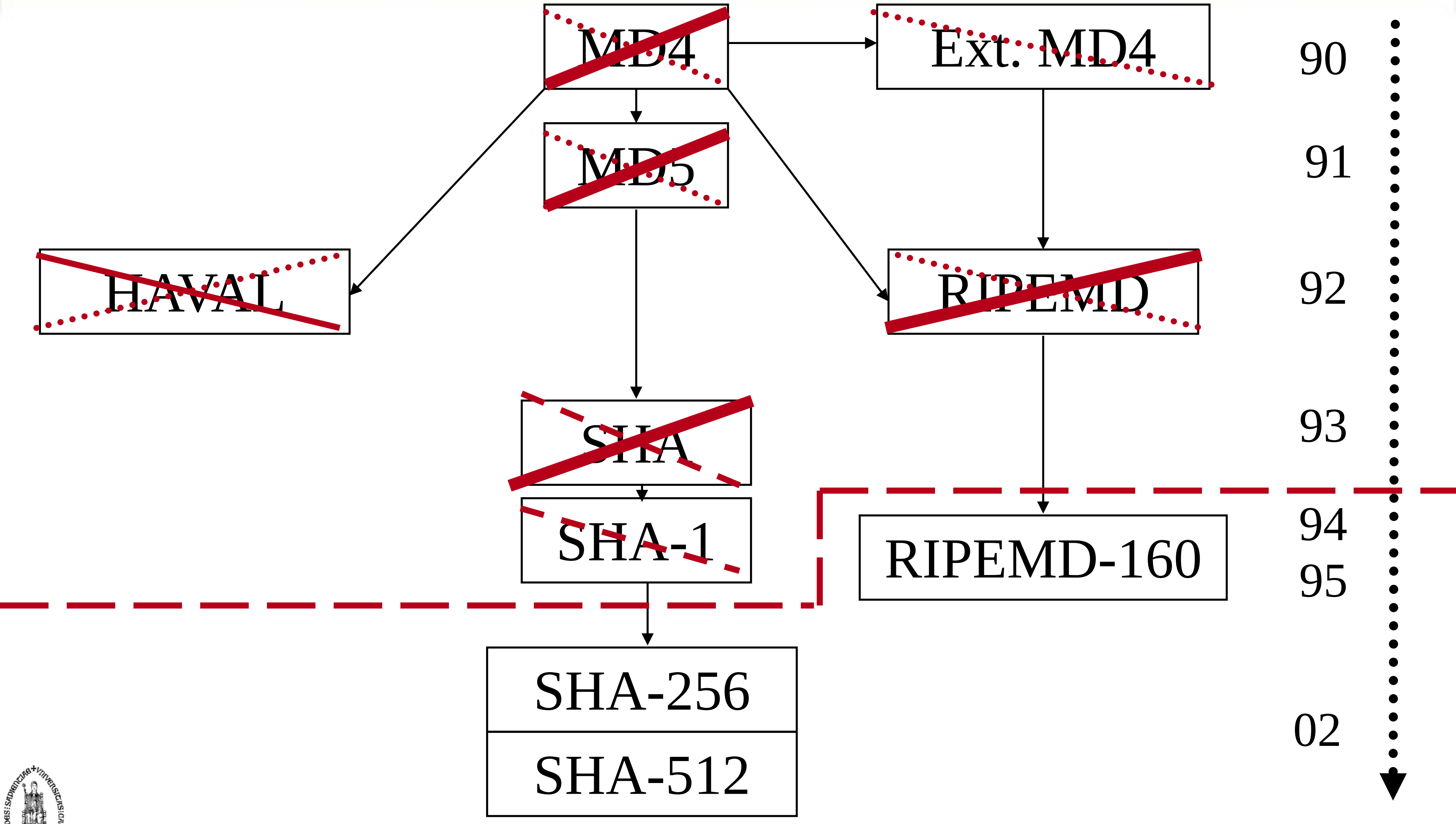
based on algebraic constructions with security proof

- factoring, discrete log, ECC: very slow
- additive: lattices
- multiplicative: matrices

dedicated hash functions

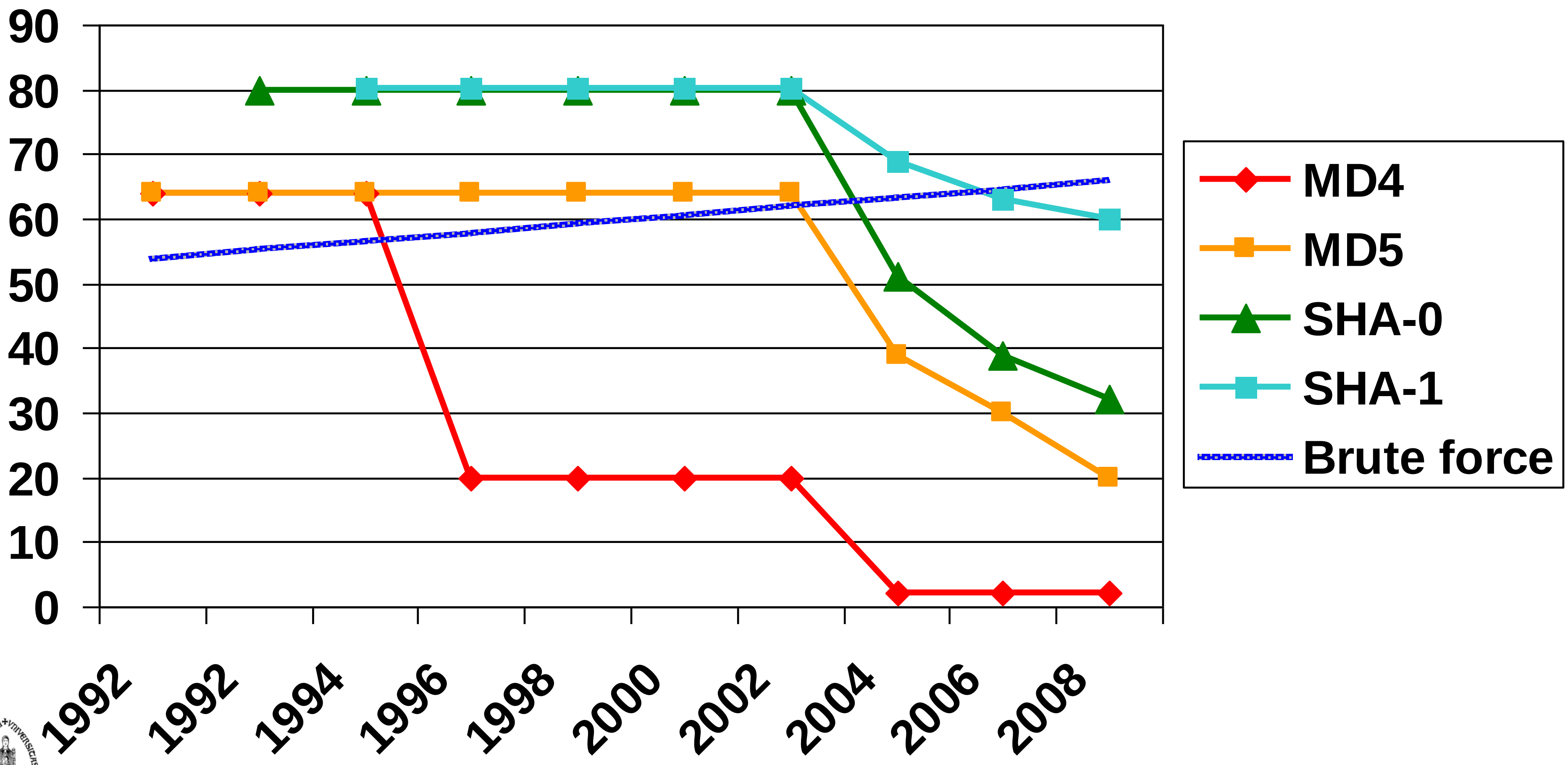
- >40 designs until 2008, about 30 broken: X.509 Annex D, FFT-hash I and II, N-hash, Snefru, MD2, …
- fast schemes for 32-bit machines:
  - most popular designs: MD4 and MD5
  - US government (NIST): SHA (aka SHA-0) and SHA-1
  - Europe: RIPEMD-160
- the next generation: SHA-2 (SHA-256, SHA-512), Whirlpool,…

MDx-type hash function history



The complexity of collision attacks

Brute force: 1 million PCs or US$ 100 000 hardware



SHA-1

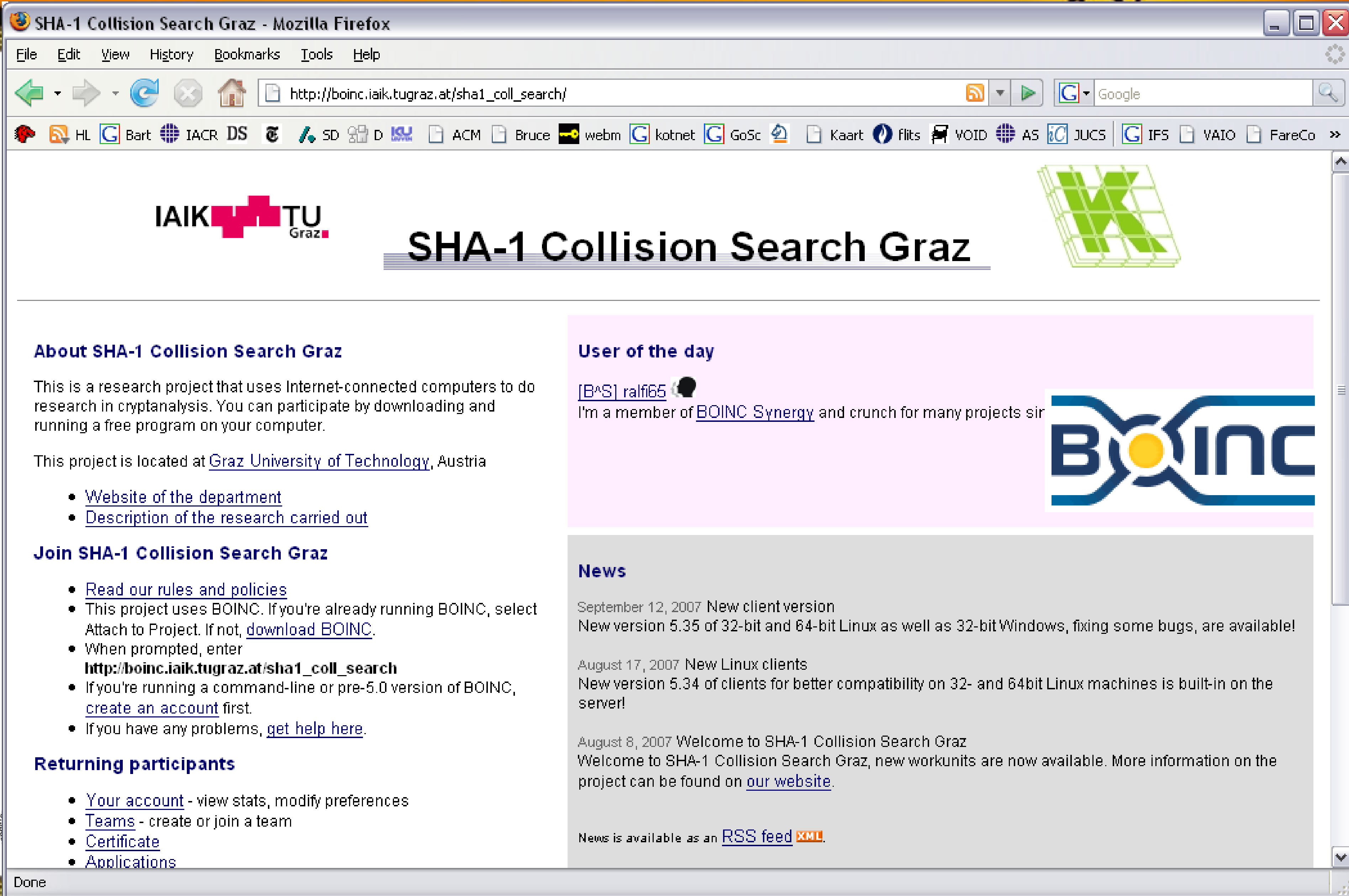
- SHA designed by NIST (NSA) in ‘93 (80 rounds)
- redesign after 2 years (’95) to SHA-1

- collisions for 53 rounds of SHA-1 [Oswald-Rijmen’04 and Biham-Chen’04]
- collisions for 58 rounds of SHA-1 [Wang+’05]
- collisions for SHA-1 in 2^{69} [Wang+’05] and 2^{63} [Wang+’05 - unpublished]
- automated search for characteristics [De Cannière-Rechberger’06+’07]:
  - collision for 64 out of 80 rounds in 2^{35} – highly structured
  - collision for 70 out of 80 rounds in 2^{44} – highly structured
- collisions for 70 rounds of SHA-1 in 2^{39} (4 days on a PC) [Joux-Peyrin’07]

- **collisions for SHA-1 in 2^{60} [Mendel+’08 - unpublished]**

Prediction: collision for SHA-1 in the next 12 months

SHA-1 collision search



SHA-1 Collision Search Graz - Mozilla Firefox

http://boinc.iaik.tugraz.at/sha1_coll_search/

IAIK TU Graz

SHA-1 Collision Search Graz

About SHA-1 Collision Search Graz

This is a research project that uses Internet-connected computers to do research in cryptanalysis. You can participate by downloading and running a free program on your computer.

This project is located at Graz University of Technology, Austria

- Website of the department
- Description of the research carried out

Join SHA-1 Collision Search Graz

- Read our rules and policies
- This project uses BOINC. If you're already running BOINC, select Attach to Project. If not, download BOINC.
- When prompted, enter **http://boinc.iaik.tugraz.at/sha1_coll_search**
- If you're running a command-line or pre-5.0 version of BOINC, create an account first.
- If you have any problems, get help here.

Returning participants

- Your account - view stats, modify preferences
- Teams - create or join a team
- Certificate
- Applications

User of the day

[B^S] ralfi65

I'm a member of BOINC Synergy and crunch for many projects sir

News

September 12, 2007 New client version
New version 5.35 of 32-bit and 64-bit Linux as well as 32-bit Windows, fixing some bugs, are available!

August 17, 2007 New Linux clients
New version 5.34 of clients for better compatibility on 32- and 64bit Linux machines is built-in on the server!

August 8, 2007 Welcome to SHA-1 Collision Search Graz
Welcome to SHA-1 Collision Search Graz, new workunits are now available. More information on the project can be found on our website.

News is available as an RSS feed XML.

Done

Crypto Hash Update - Mozilla Firefox

File Edit View Go Bookmarks Tools Help

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

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

How to Choose an Algorithm

msdn

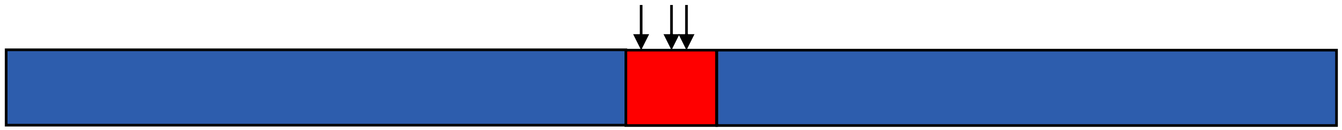
- For example, SHA1 uses a 160-bit encryption key, whereas MD5 uses a 128-bit encryption key; thus, SHA1 is more secure than MD5 and thus is a much harder hash to break.

- Another point to consider about hashing algorithms is whether or not there are practical or theoretical possibilities of collisions. Collisions are bad since two different words could produce the same hash. SHA1, for example, has no practical or theoretical possibilities of collision. MD5 has the possibility of theoretical collisions, but no practical possibilities. So choosing an algorithm comes down to the level of security you need.

This “information” was available on MSDN until Summer 2008

Impact of collisions (1)

- 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
  - 2 colliding postscript/gif/… documents [Lucks-Daum’05]
  - 2 colliding RSA public keys – thus with colliding X.509 certificates [Lenstra-Wang-de Weger ’04]
  - chosen prefix attack: different IDs, same certificate [Stevens+’07]
  - **2 arbitrary colliding files (no constraints) for 15K$**

Impact of collisions (2)

- [**Sotirov-Stevens-Appelbaum,-Lenstra-Molnar-Osvik-de Weger** ’08] **MD5 considered harmful today**
  - fake CA certificate.
  - results in a rogue CA: its certificates are trusted by all common browsers
  - need to predict serial number + validity period

- 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

Impact of collisions (3)

- 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 (2nd) preimages?

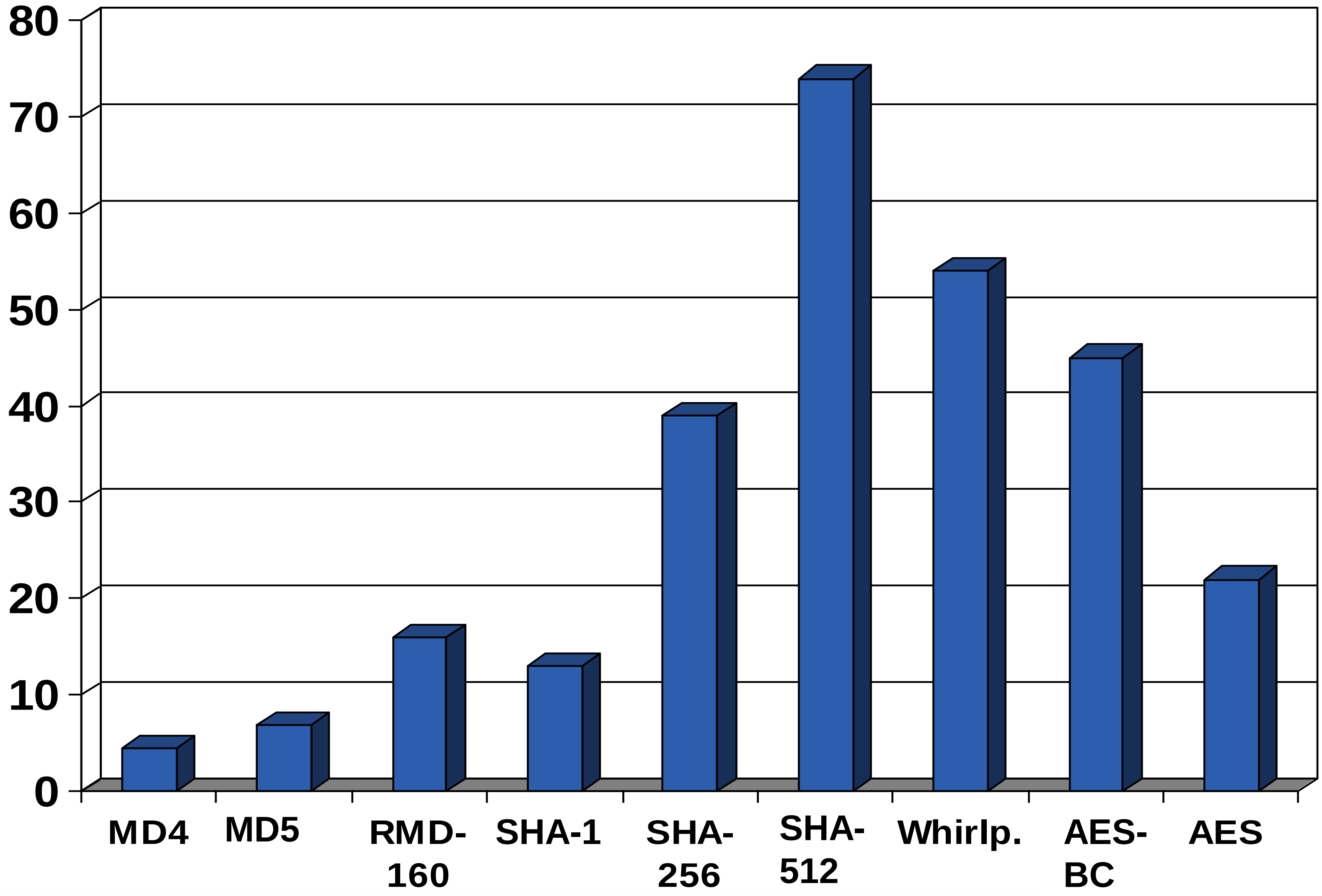
- security degrades with number of applications
- for large messages even with the number of blocks (cf. supra)
- specific results:
  - MD2: 2^{73}
  - MD4: 2^{102} [Leurent'08]
  - MD5: 2^{12x} [Asaki-Aoki'08]
  - SHA-1: 45 of 80 steps [De Cannière-Rechberger'08]

Fixes/Alternatives

- RIPEMD-160 seems more secure than SHA-1 ☺
- message precoding for SHA-1
- small patches to SHA-1

- use more recent standards (slower on 32-bit machines)
  - SHA-2 family: SHA-256, SHA-512
  - Whirlpool

Performance of hash functions
(cycles/byte) Pentium III

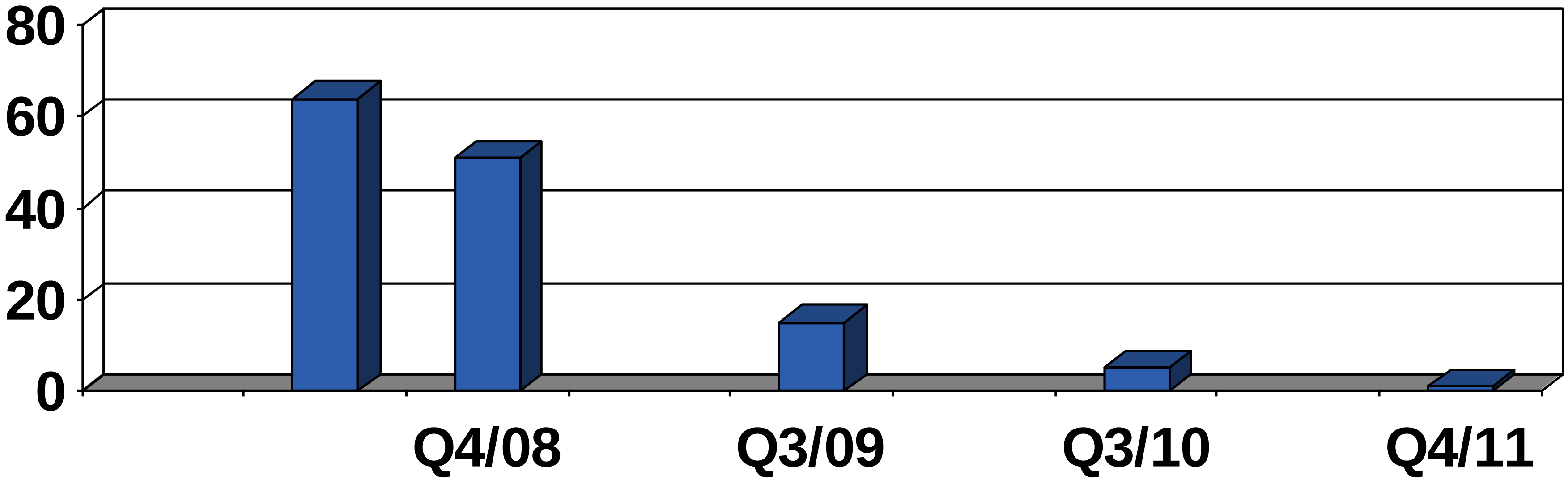


Outline

- definitions
- applications
- generic attacks
- attacks on iterated constructions
- attacks on custom designed hash functions: SHA-1
- the NIST AHS competition (SHA-3)
- conclusions

NIST Advanced Hash Function competition (AHS)

- SHA-3 must support 224, 256, 384, and 512-bit message digests, and must support a maximum message length of at least $\mathbf{2^{64}}$ bits
- standard will be published in 2012



Call: 02/11/07

Deadline (64): 31/10/07

Phase 1 (51): 9/12/07

phase 1

phase 2

final

Accepted submission to AHS competition (1)

- at least 12 out of 51 have been broken
  - 10 designers have conceded so far
- 13 not accepted, but only 4 are public (all 4 have been broken)

Analysis of 24/51 (designs available publicly + unbroken after 1 month)

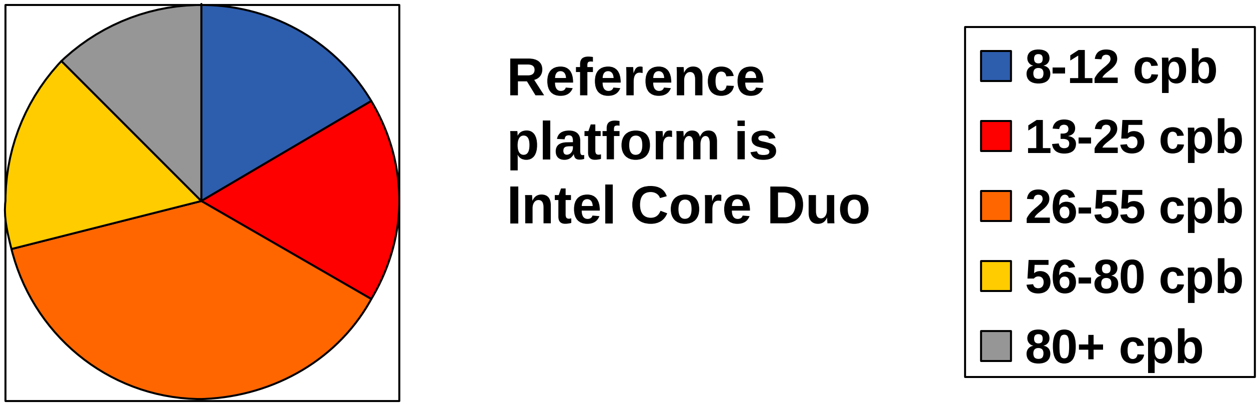
Accepted submission to AHS competition (2)

Analysis of 24/51 (designs available publicly + unbroken after 1 month)

- All wide pipe + sponge designs have an output transformation
- Four narrow designs do not have an output transformation
- Most narrow designs have a counter (2 do not)

Accepted submission to AHS competition (3)

Analysis of 24/51 (designs available publicly + unbroken after 1 month)



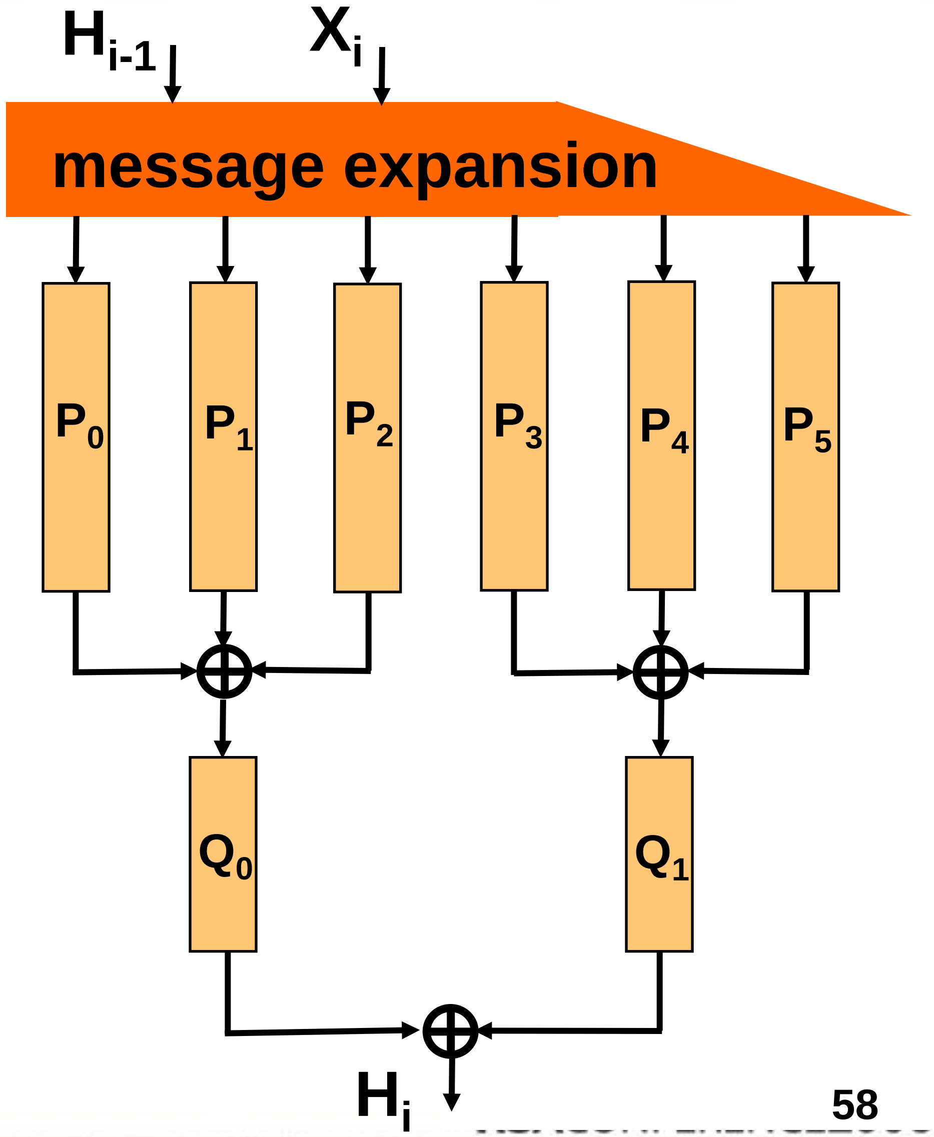
Reference platform is Intel Core Duo

Selected designs (highly subjective)

- ARIRANG [KO] – J. Lim
- Blake [CH] – J.-P. Aumasson
- Cubehash [US] – D.J. Bernstein
- Echo [FR] – H. Gilbert
- Fugue [US] – C. Jutla
- Grøstl [DK/AT/PO] – L.R. Knudsen
- JH [Singapore] – H. Wu
- Keccak [BE/IT] – J. Daemen
- LANE [BE] – S. Indesteege
- Lesamnta [JP] – H. Yoshida
- Luffa [JP] – D. Watanabe
- MD6 [USA] – R.L. Rivest
- SHAvite-3 [IL] – O. Dunkelman
- SIMD [FR] – G. Leurent
- SKEIN [USA] – B. Schneier

Advertisement: LANE

- Designer: S. Indesteege (COSIC)
- H_i 256 bit and X_i 512 bit
- Expanded linearly to 6 256-bit words
- P_i /Q_i consist of 6/3 AES parallel rounds
  - AddRoundKey: add round constant and counter
  - SwapColumn to mix two 128-bit halves



Issues

- Security
  - How to define an attack, e.g. pseudo-near collision, attacks with huge memory?
  - Importance of proofs
- Performance
  - Designs with tunable security/performance tradeoff: how important are the nominal parameters?
  - Do we care about a very large memory (500-700 bytes) which may be a problem for small devices?
  - Can we exploit 64 or 128 cores? Intel AES instruction?
- Note that the winner selected in 2012 will reflect the state of the art in October 2008

Hash functions: conclusions

- hash functions such as SHA-1 would have needed 128-160 rounds instead of 80
- recent attacks are not dramatic for all applications, but they form a clear warning: upgrade asap
- limited understanding (theory and practice)
- use weaker security assumptions if possible (UOWHF?)
- research on new and more robust designs with extra features

Hash functions: further reading

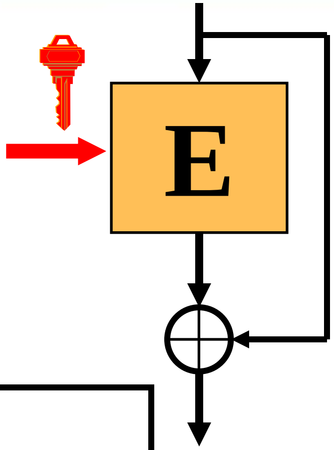
- NIST http://csrc.nist.gov/groups/ST/hash/index.html
  - first SHA-3 candidate conference: February 25-28, 2009, Leuven
  - workshop October 31-November 1, 2005 and August 24-25, 2006
- ECRYPT: http://www.ecrypt.eu.org
  - SHA-3 Zoo http://ehash.iaik.tugraz.at/wiki/The_SHA-3_Zoo
  - workshops in May 2007 and June 2005 + statement on hash functions
- The IACR eprint server http://eprint.iacr.org
- My 1993 PhD thesis http://homes.esat.kuleuven.be/~preneel
- Overview paper from 1998 (LNCS 1528) http://www.cosic.esat.kuleuven.be/publications/article-246.pdf

Thank you for your attention

Fixes/Alternatives (2)

- number theoretic schemes
  - secure but very slow (1 multiplication per bit)
  - speedup by [Contini,Lenstra,Steinfeld'05] VSH
    - still 20 times slower than SHA-1
    - only collision resistance; some other weaknesses
  - topic for further research (lattices, matrices)

Hash function: pseudorandom function (1)



- MDx are based on a block cipher with a feedforward: where to put the key?
- if keyed to the message input: related key boomerang distinguisher attacks apply [Kim+'06]

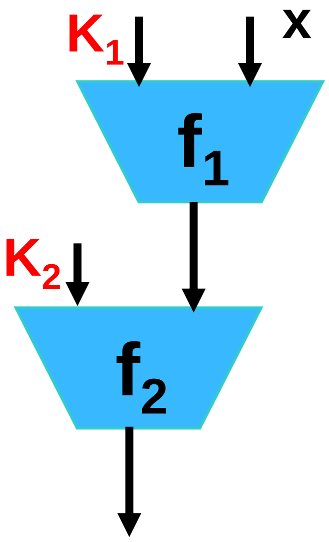
	Rounds of attack	Data complexity
Haval-4	96	$2^{11.6}$ RK-CP + 2^{6} RK-ACC
MD4	48	2^{6} RK-CP + 2^{6} RK-ACC
MD5	64	$2^{13.6}$ RK-CP + $2^{11.6}$ RK-ACC
SHA-1	59 of 80	$2^{70.3}$ RK-CP + $2^{68.3}$ RK-ACC

many hash functions are based on pretty weak block ciphers

Hash function: pseudorandom function (2)

- HMAC keys through the IV (plaintext) [Kim+'06]
  - collisions for MD5 invalidate current security proof of HMAC-MD5
  - new attacks on reduced version of HMAC-MD5 and HMAC-SHA-1

	Rounds in f2	Rounds in f1	Data complexity
Haval-4	128	102 of 128	2^{254} CP
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	80	80	2^{109} CP
SHA-1	80	53 of 80	$2^{98.5}$ CP



no problem yet for most widely used schemes

Hash function: pseudorandom function (3)

- Some applications still use HMAC-MD4!
- NMAC weaker than HMAC
- One application that is vulnerable: APOP (password divided over two bloks)