

ECRYPT II

 www.ecrypt.eu.org


Hash Functions: Design and introduction to cryptanalysis

Bart Preneel
 Katholieke Universiteit Leuven - COSIC
 firstname.lastname@esat.kuleuven.be




Hash functions

X.509 Annex D
MDC-2
MD2, MD4, MD5
SHA-1

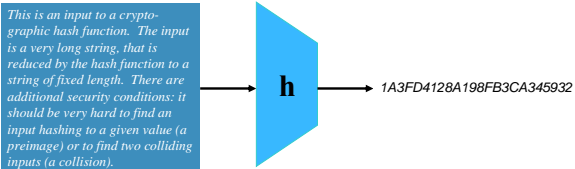


RIPEMD-160
SHA-256
SHA-512



SHA-3

This is an input to a cryptographic 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).



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)

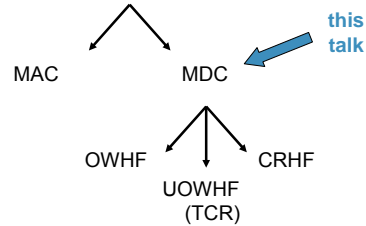
Compression functions

SHA-{0,1,2}

SHA-3 bits and bytes

Hash function flavours

cryptographic hash function



Informal definitions

- 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
 - 2nd preimage resistance
- Collision Resistant Hash Function (CRHF): OWHF +
 - collision resistant

Security requirements (n-bit result)

preimage

2^n

2nd preimage

$x \neq ?$

2^n

collision

$? \neq ?$

$2^{n/2}$

Preimage resistance

2^n

- 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

$x \neq ?$

2^n

x → Channel 1: high capacity and insecure

$h(x)$ → Channel 2: low capacity but secure (= authenticated – cannot be modified)

- 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

$x \neq x'$

$2^{n/2}$

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 $x' \neq x$ she can cheat
- if Bob can find a **preimage**, he can learn x and cheat

collision

$x \neq x'$

$2^{n/2}$

Brute force (2nd) 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 $p \geq 0.6 r(r-1)/2S$

13

Functional graph of $f(x) = x^2 + 7 \pmod{11}$

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

14

Brute force collision search

- Consider the functional graph of h

15

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

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

16

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]

17

Relation between properties

[Rogaway-Shrimpton'04]
 [Stinson'06]
 [Reyhanitabar-Susilo-Mu'10]
 [Andreeva-Stam'10]

Even if $\text{Coll} \Rightarrow \text{xSEC/Pre}$: bound always $2^{n/2} \ll 2^n$

18



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)

19

Quantum computers

- in principle exponential parallelism
- inverting a one-way function: 2ⁿ 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})

20

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 (indifferentiable from a random oracle) – but see [Ristenpart-Shacham-Shrimpton'11]
 - near-collision resistance
 - partial preimage resistance (most of input known)
 - multiplication freeness
- how to formalize these requirements and the relation between them?

21

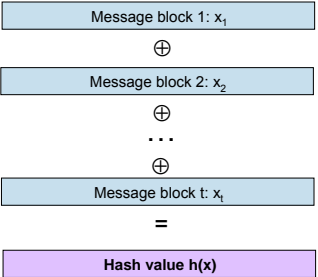
Iteration

(mode of compression function)

22

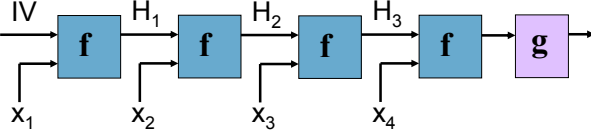
How not to construct a hash function

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



23

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 ...

24

Security relation between f and h

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

25

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 \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

26

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

27

Property preservation

[Andreeva-Mennink-P'10] for overview

Sec/Pre preservation seems to be problematic
Is Pre preservation meaningful?

	Coll	Sec	Pre	Pro	aSec	eSec	aPre	ePre
Suffix- & Prefix-free MD					Not applicable			
Envelope MD								
BCM				?				
Haifa								
RMX								
Shoup UOWH								
ROX								

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
- [...]

29

Attacks on MD-type iterations

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

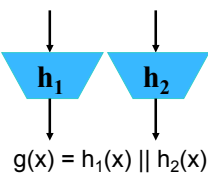
30

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

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

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....



$g(x) = h_1(x) \parallel h_2(x)$

31

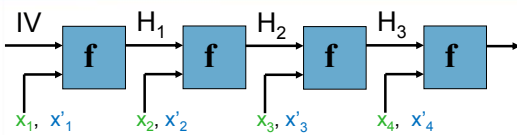
Multiple collisions \neq multi-collision

Assume “ideal” hash function h with n -bit result

- $\Theta(2^{n/2})$ evaluations of h (or steps): 1 collision
– $h(x)=h(x')$
- $\Theta(r \cdot 2^{n/2})$ steps: r^2 collisions
– $h(x_1)=h(x'_1); h(x_2)=h(x'_2); \dots; h(x_{r^2})=h(x'_{r^2})$
- $\Theta(2^{2n/3})$ steps: a 3-collision
– $h(x)=h(x')=h(x'')$
- $\Theta(2^{n(t-1)/t})$ steps: a t -fold collision (multi-collision)
– $h(x_1)=h(x_2)=\dots=h(x_t)$

32

Multi-collisions on iterated hash function (2)



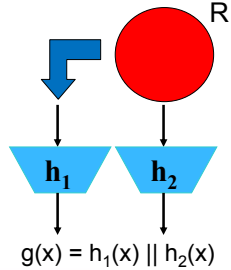
- for H_1 : collision for block 1: x_1, x'_1
- for H_2 : collision for block 2: x_2, x'_2
- for H_3 : collision for block 3: x_3, x'_3
- for H_4 : collision for block 4: x_4, x'_4

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

33

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 $R = 2^{n/2}$ -fold multi-collision for h_2
 - in R : search by brute force for h_1
- Time: $n \cdot 2^{n/2} + 2^{n/2} \ll 2^{(n_1 + n_2)/2}$



$g(x) = h_1(x) \parallel h_2(x)$

34

Multi-collisions [Joux '04]

consider h_1 (n_1 -bit result) and h_2 (n_2 -bit result), with $n_1 \geq n_2$. concatenation of 2 iterated hash functions ($g(x) = h_1(x) \parallel h_2(x)$) is **as most as strong as the strongest** of the two (even if both are independent)

- cost of collision attack against g at most $n_1 \cdot 2^{n_2/2} + 2^{n_1/2} \ll 2^{(n_1 + n_2)/2}$
- cost of (2nd) preimage attack against g at most $n_1 \cdot 2^{n_2/2} + 2^{n_1} + 2^{n_2} \ll 2^{n_1 + n_2}$
- if either of the functions is weak, the attacks may work better

35

Summary



36

Improving MD iteration

salt + output transformation + counter + wide pipe

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

37

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]

38

Compression functions

39

Block cipher (E_K) based

Davies-Meyer

Miyaguchi-Preneel

- 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],...

40

Blake

Davies-Meyer + double pipe internally

41

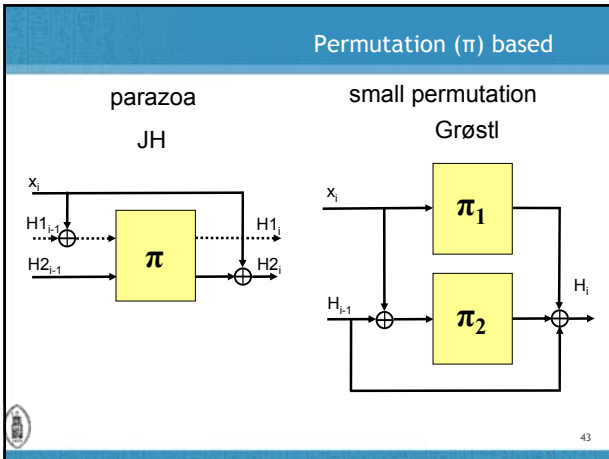
Permutation (π) based: sponge

absorb squeeze

Examples: Panama, RadioGatun
 Keccak (no buffer)

Generalization called Paraozo
 JH, Cubehash, Fugue, Grindahl, Hamsi, Luffa

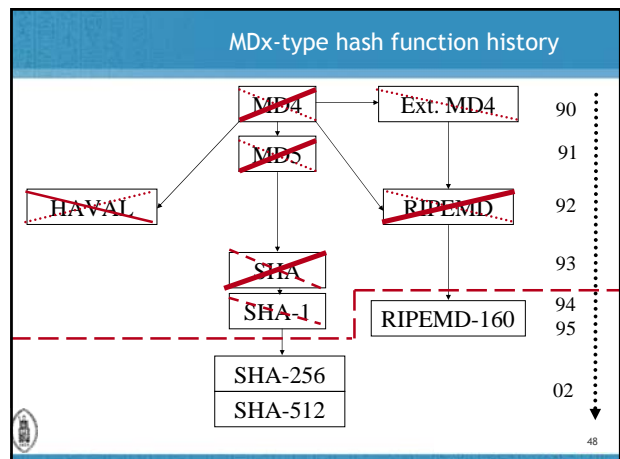
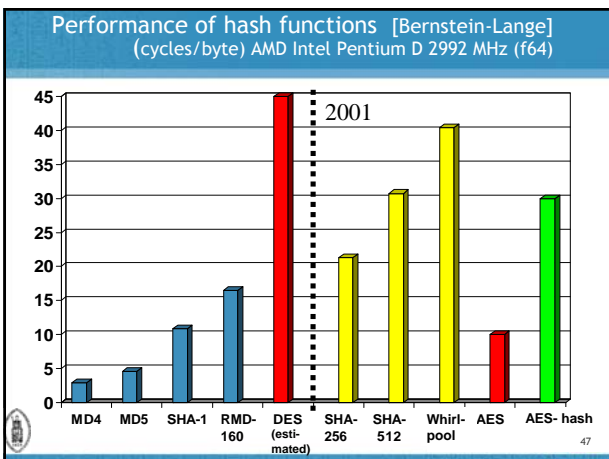
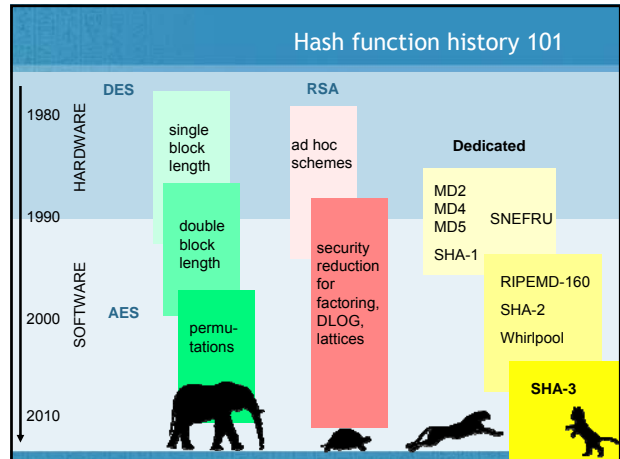
42

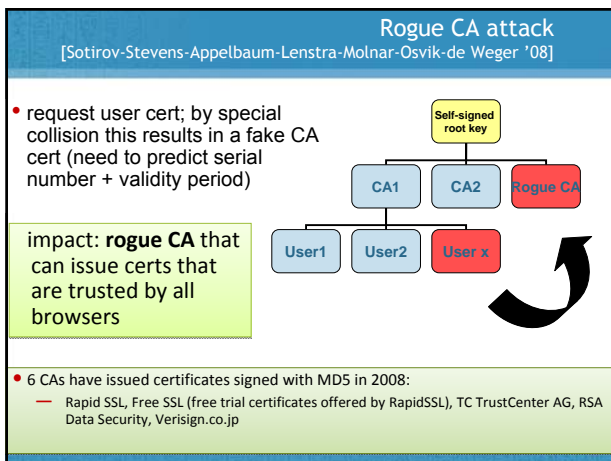
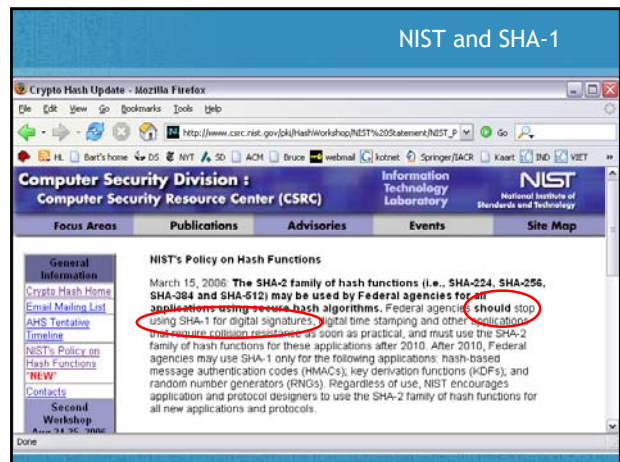
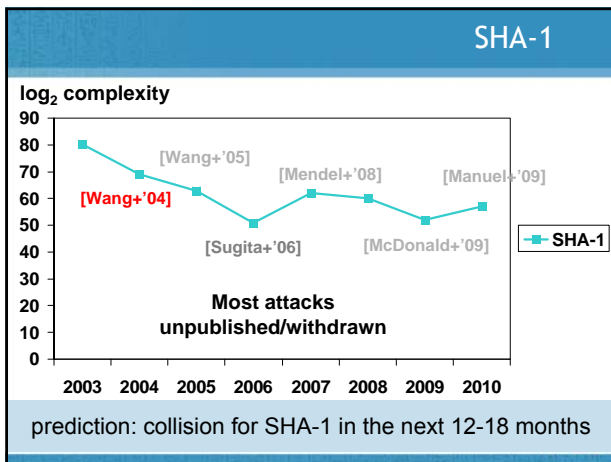
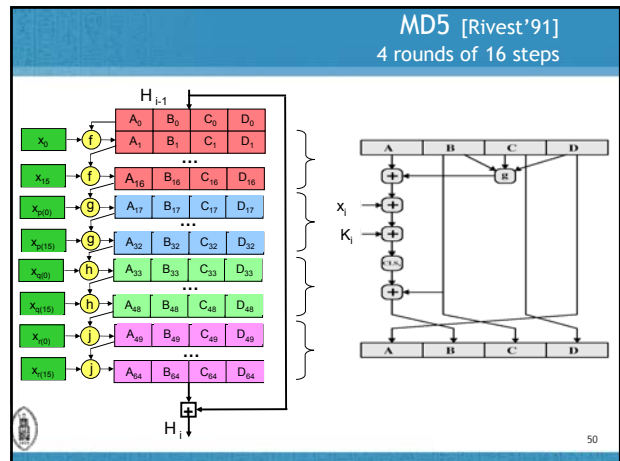
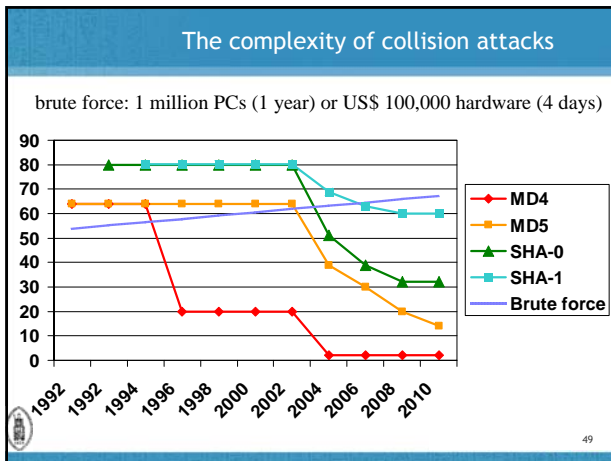


- ### Iteration modes and compression functions
- security of simple modes well understood
 - powerful tools available
 - analysis of slightly more complex schemes very difficult
 - which properties are meaningful?
 - which properties are preserved?
 - MD versus sponge is still open debate
- 44

SHA-{0,1,2}

45





- ### 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)

55

SHA-3

(bits and bytes)

56

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

Round	Time Period	Number of Candidates
Round 1	Q4/08	64
Round 2	Q3/09	51
Round 2	Q4/10	14
Final	Q3/12	5

57

The candidates

31/10/2008

58

Slide credit: Christophe De Cannière

Preliminary cryptanalysis

16/06/2009

59

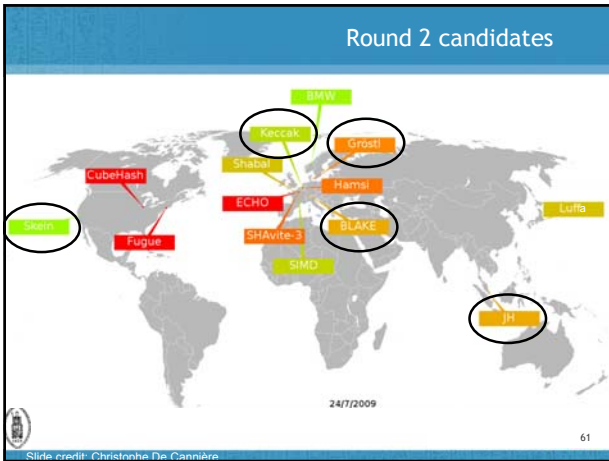
Slide credit: Christophe De Cannière

End of Round 1 candidates

8/7/2009

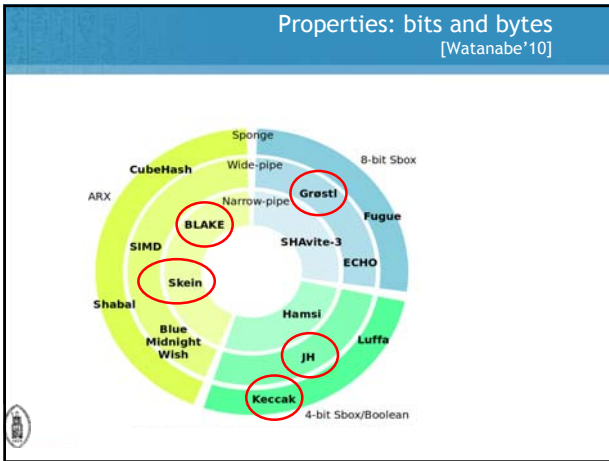
60

Slide credit: Christophe De Cannière



Compression function/iteration

	Block cipher	Permutation	MD/HAIFA
Blake			HAIFA
Groestl		2-permutation	MD
JH			JH-specific
Keccak		Sponge	
Skein	MMO		MD*/Tree (UBI)
BMW	PGV variant		MD
Cubehash		Sponge-type	
ECHO			HAIFA
Fugue		Spong-type	
Hamsi			
Luffa		Sponge-type	
Shabal		Sponge-type	
Shavite-3	Davies-Meyer		HAIFA
SIMD	PGV variant		MD



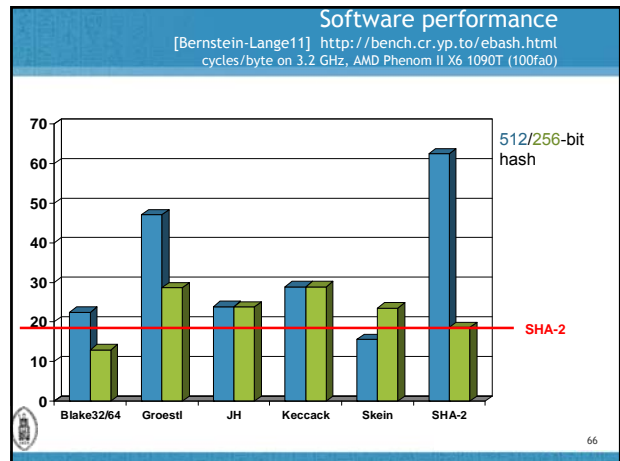
Security reductions [Andreeva-Mennink-P'10]

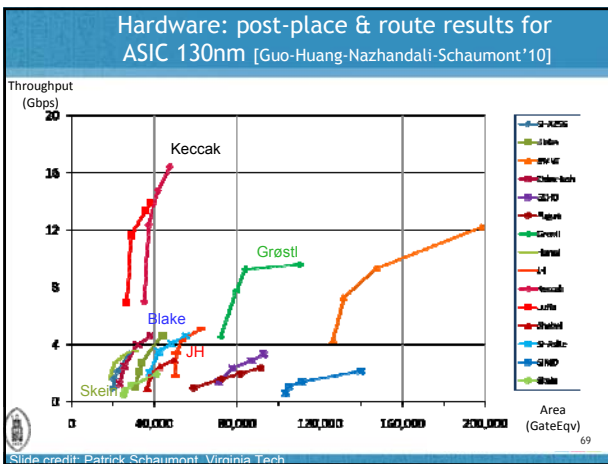
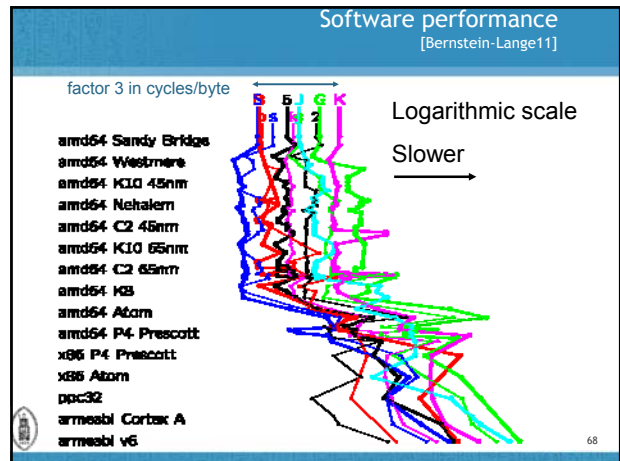
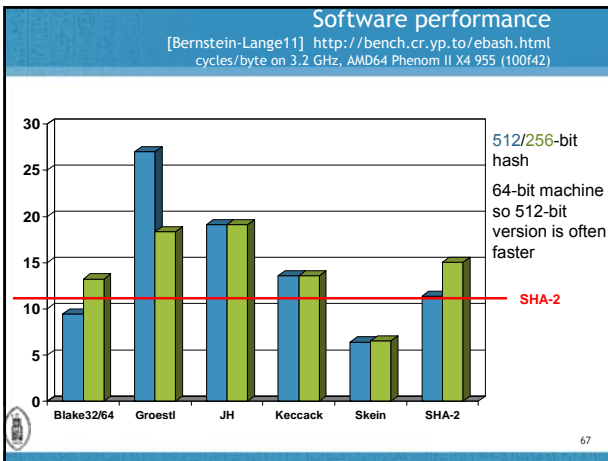
	compression function			hash function			
	pre	sec	col	pre	sec	col	indiff
BLAKE	?		?				
BMW	?		?				?
CubeHash							
ECHO					?		?
Fugue							
Groestl							
Hamsi							?
JH							
Keccak							
Luffa							?
Shabal							
SHAvite-3							
SIMD							?
Skein							
SHA-2							

Security: SHA-3 Zoo

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

Hash Name	Principal Submitter	Best attack on Merkle DAG	Best attack on other Merkle Requirements
Blake2	Jean-Philippe Aumasson		
Groestl	Lars R. Knudsen		
JH	Junjiro Aoki		
Keccak	The Keccak Team		
Skein	Bruce Schneier		





- ### 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
- 70

- ### 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
 - 4/5 tweaked before final
 - NIST expects that SHA-2 and SHA-3 will co-exist
 - variable number of rounds?
- 71

- ### 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
 - New challenge: lightweight hashing:
 - Armadillo, Photon, Quark, Spongnet,...
- 72

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

