Cryptographic Hash Functions

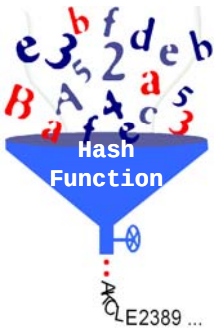
Vincent Rijmen

Thanks to the members of the Krypto group

Overview

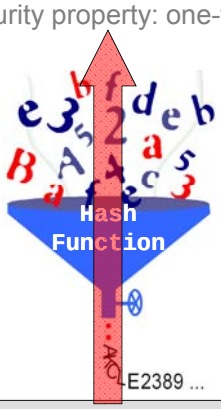
- Definitions & terminology
- Constructions
- Applications
- Differential attacks on hash functions
- Status of standard hash functions

A cryptographic hash function produces cryptographic checksums or fingerprints

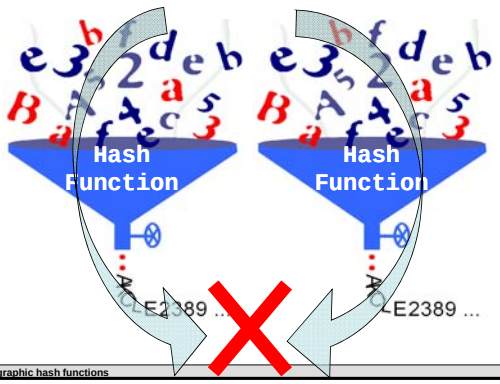


- **Fast**
- **Secure**

First security property: one-wayness



Second security property: collision resistance



Some definition problems

- Information-theoretic
  - Collisions always exist
- Complexity-theoretic
  - Standardised hash functions are fixed algorithms, not classes
  - Finding a collision is difficult only the first time
- Largely ignored by “practical” people

Some other problems

- Designs with provable security often ignore properties which are important in practice
- *Near-collisions:* two inputs give *almost* the same output
  - May interact badly with applications
- One-wayness: for *all* outputs, *most* outputs, *most probable* outputs?

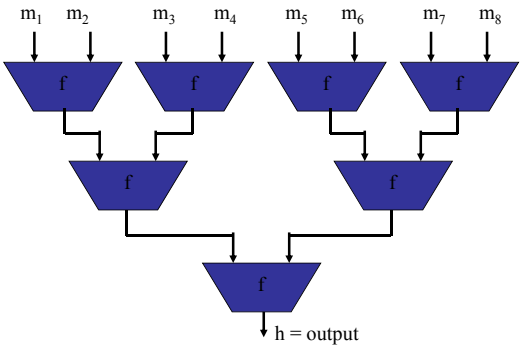
Constructing hash functions

- With or without secret parameter (key)
  - Message Authentication Codes (MAC)
  - Universal hash functions
  - Manipulation Detection Codes (MDC, hash)
- Compression function
- Extension method
  - Tree
  - Merkle's method

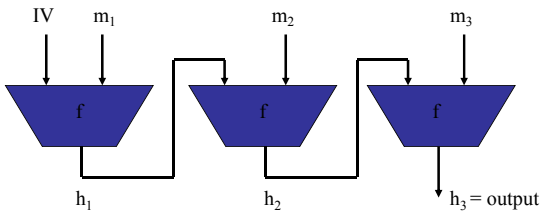
Compression function

- Fixed-size inputs
- Processes a part (*block*) of the message
- Outputs are inputs for next iteration

Tree



Merkle's method



Merkle-Damgaard Theorem

If:

- Unique padding of message until length is multiple of block length
- Length of (original) message included into padding

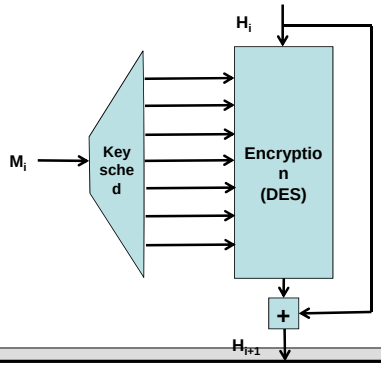
Then:

Collision-resistant compression function used in Merkle method ⇒ collision-resistant hash function

Compression function constructions

- Block cipher based
- Dedicated designs
- Number theoretic constructions

Davies-Meyer (1979)



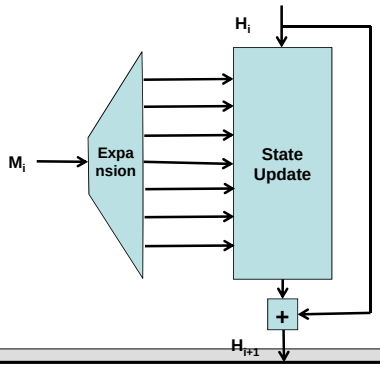
Motivation

- Reduce amount of primitives to be implemented
- Block cipher DES was the only cryptographic primitive with a certification
- Block cipher + feed forward = noninvertible map without apparent structure
- 3 variants exist

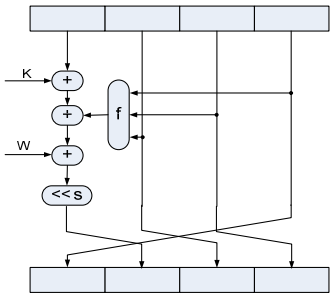
Problems

- Slow (frequent re-keying of block cipher)
- Short output length (generic attacks)
  - Constructions to get larger output size are even slower
- Security based on non-standard assumptions about the block cipher

MD4 (R. Rivest,1990)



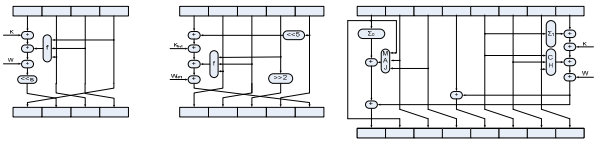
MD4 state update: Unbalanced Feistel Network (48 iterations)



- No arguments for its security
- Fast on 32-bit CPUs

State updates in the MD4 family

MD4 SHA/SHA-1 SHA-256



Design principles copied in MD5, RIPEMD, HAVAL, SHA, SHA-1, **RIPEMD-160, SHA-256, SHA-384, SHA-512, SHA-224, HAVAL-3, HAVAL-4, HAVAL-5, HAS-160, HAS-V, ...**

– All hash functions in use today

Based on number theory

$h(x) = a^x \bmod p$

- If p is large, this is thought to resist collision attacks and preimage attacks
- Problems:
  1. Doesn't compress
  2. Definitely not 'random'

The MASH functions

- Based on difficulty of factoring
  - Slow
- MASH-1(n)

$$y_i = 1111x_{i1}1111x_{i2}1111\ldots x_{it}$$

$$h_{i+1} = (((h_i \oplus y_i) \vee 0xF00\ldots0)^2 \bmod pq) \oplus h_i$$

- MASH-2(n): exponent 2^8+1

Observations

- Very little diversity in designs
- Everyone seemed quite confident about the security of standard hash functions
  - Also reflected in specification of applications

Applications of hash functions

- Payment schemes
- Secure storage of passwords
- Generation of pseudo-random numbers
- Electronic (digital) signatures
- Commitments

Example 1: Coin flip by telephone

Generate random bit *b*

- 0 = heads; 1 = tails

Choice: head or tails

Compare b with choice

Outcome

☹

Commitment

- Generate random number *r*
- Heads/tails = first bit of *r*
- Compute *commitment x* = hash(*r*)

x →

← Choice: heads or tails

r →

Requirements

- Collision resistance
  - If I can find xyz, x'y'z' s.t. h(0xyz) = h(1x'y'z'), then the commitment is not binding
- Preimage resistance
  - If computing $r = h^{-1}(x)$ is feasible, I always lose

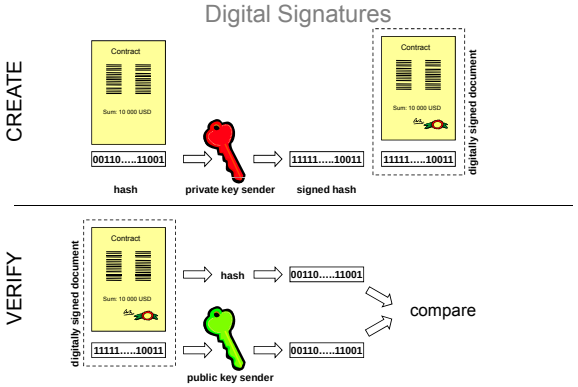
Example 2: Electronic Signature

Cut and paste is easy with digital documents

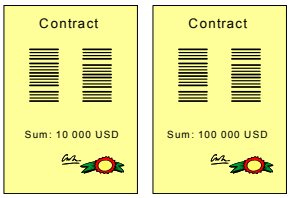
Electronic signatures are *different* for each document

f(fingerprint of document, secret of signer)

Digital Signatures



Attack on Electronic Signature



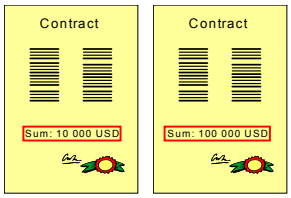
Two documents producing the same fingerprint.

Same fingerprints ⇒ same signatures.

Substitute the original document by a forgery

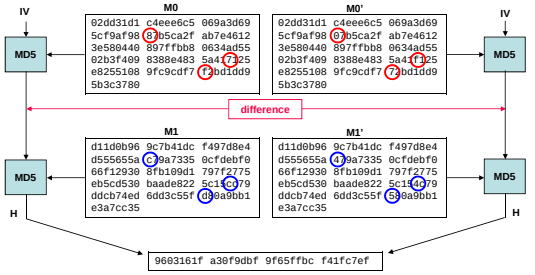
Meaningful Collisions

- Two documents with meaningful content



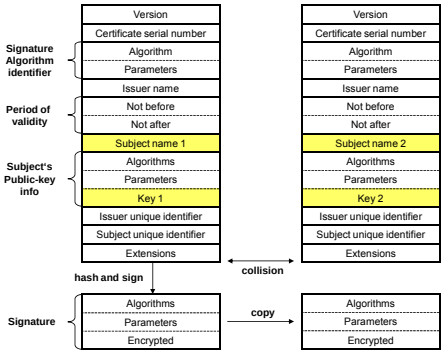
- Adversary has many constraints

Meaningful Collisions are difficult



colliding message pairs for MD5 [WY2005]

Colliding X.509 certificates [SLdW2007]



A collision with meaning (64 steps of SHA-1)

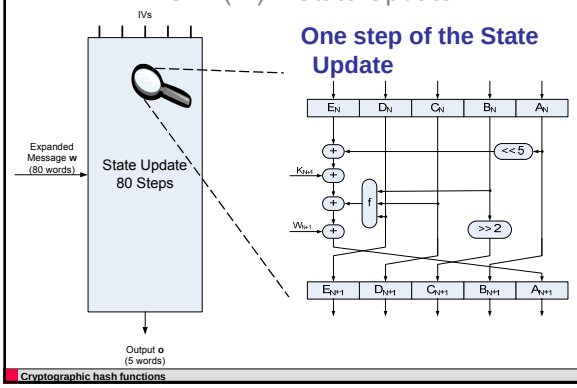
```
4920 6865 7265 6279 2073 6f6c 656d 6e6c  I hereby solemnl
7920 7072 6f6d 6973 6520 746f 2066 696e  y promise to fin
6973 6820 6d79 2050 6844 2074 6865 7369  ish my PhD thesi
7320 6279 2074 6865 2065 6e64 206f 6620  s by the end of
3230 3035 200a 0aea cbd7 029e 9f21 9821  2005 ........!.!
f0f0 ff92 13e4 3df4 07ca 4a69 0673 6850  ......=...Ji.shP
7f39 7c77 dddf 45c1 52ac 0ab0 9d15 11cf  .9|w..E.R.......
a15f dc78 9f4d 8621 5d1d 41f3 c2a7 3c6a  ._.x.M.!].A...<j
c2b5 d3a1 1ebb 7dee ffc2 7fb5 5c31 535c  ......}.....\1S\
8fb1 3dce c26a 4b89 0e82 d260 8ce7 31fb  ..=..jK....`..1.
383b 24d9 37fb eca9 f5e3 90b6 c123 15d5  8;$.7........#..
c1a4 8abe 9ad3 c1df f6d5 50c9 6bd9 572d  ..........P.k.W-

2nd (colliding) message:
4920 6865 7265 6279 2073 6f6c 656d 6e6c  I hereby solemnl
7920 7072 6f6d 6973 6520 746f 2066 696e  y promise to fin
6973 6820 6d79 2050 6844 2074 6865 7369  ish my PhD thesi
7320 6279 2074 6865 2065 6e64 206f 6620  s by the end of
3230 3036 600a 0ab8 8bd7 02de 7f21 9873  2006`........!.s
50f0 ff92 93e4 3db4 27ca 4a68 2673 6830  P.....=.'.Jh&sh0
ff39 7c76 9ddf 4583 92ac 0af3 dd15 11ed  .9|v..E.........
a15f dc7b df4d 8663 9d1d 41b0 02a7 3c48  ._.{.M.c..A...<H
c2b5 d3a2 5ebb 7dbc bfc2 7ff5 bc31 530e  ....^.}......1S.
2fb1 3dce 426a 4bc9 2e82 d261 ace7 319b  /.=.BjK....a..1.
b83b 24d8 77fb eceb 35e3 90f5 8123 15f7  .;$.w...5....#..
c1a4 8abd dad3 c19d 36d5 508a abd9 570f  ........6.P...W.
```


Hash function crisis [2004-2005]

- New cryptanalysis technique announced
  - Novel method to do differential cryptanalysis
  - Collision = output difference zero
- Collisions for MD4, MD5, RIPEMD in minutes
- Collisions for SHA (SHA-0) in hours
- Collisions for SHA-1 “theoretically possible”
  - 2^{69} hashing operations

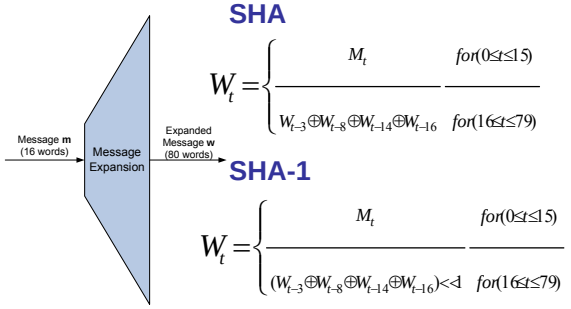
SHA(-1) – State Update



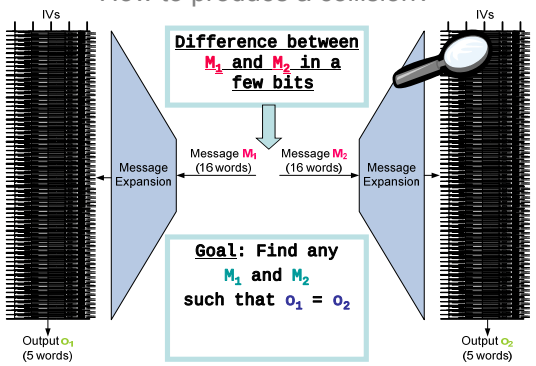
SHA(-1) State update components

- Modular additions
- Rotations
- Boolean function f
  - Steps 1-20: “IF”
  - Steps 21-40: XOR
  - Steps 41-60: “MAJ”
  - Steps 61-80: XOR

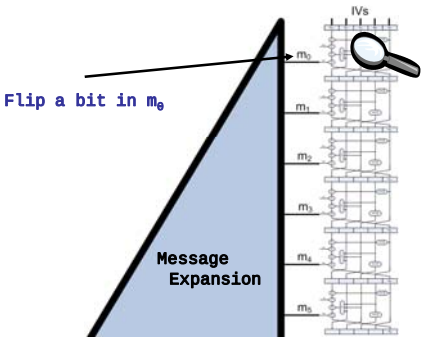
SHA(-1) Message Expansion



How to produce a collision?

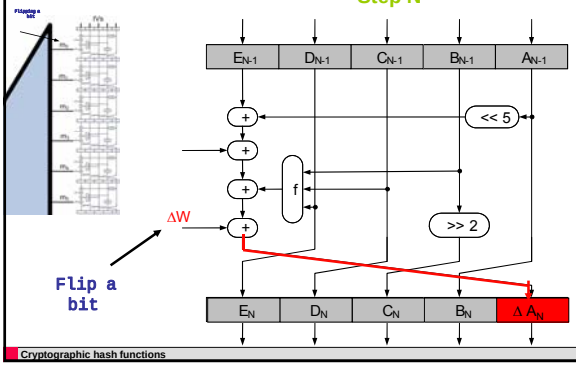


Propagation of a small difference



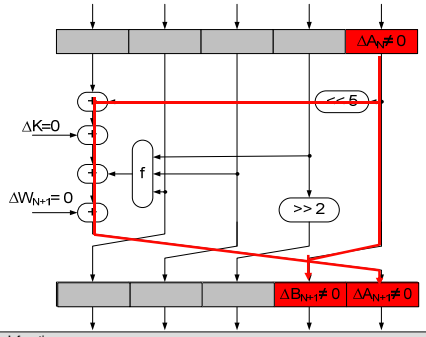
Propagation of a small difference

Step N



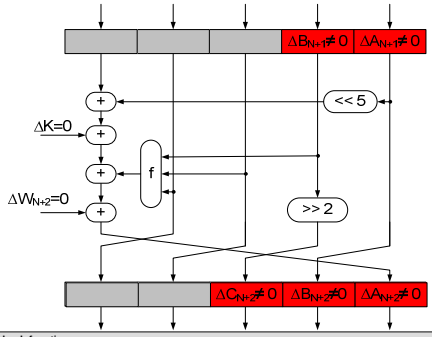
Propagation of a small difference

Step N+1



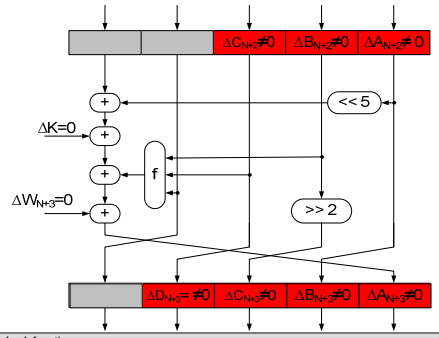
Propagation of a small difference

Step N+2



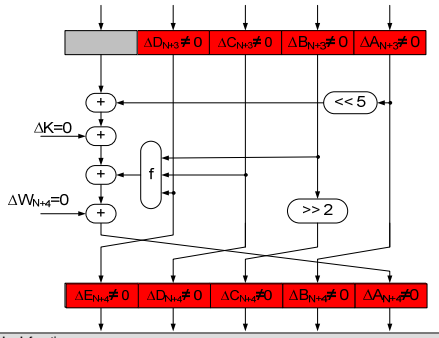
Propagation of a small difference

Step N+3

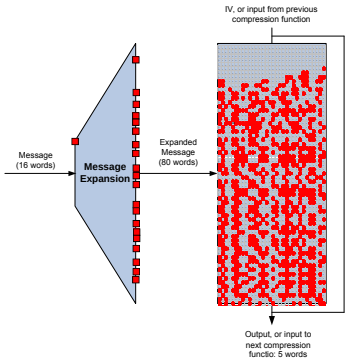


Propagation of a small difference

Step N+4



Effect of a single bit flip

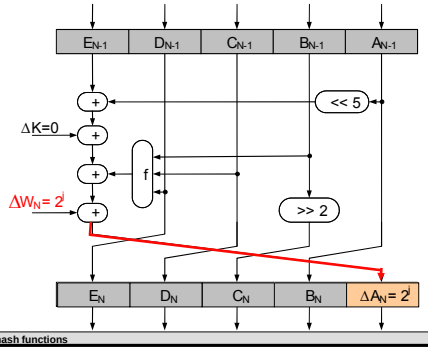


Differential cryptanalysis of SHA: Chabaud & Joux [1998]

- *Perturbations* and *corrections*: bit flips can cancel one another
- Nonlinear functions expand bit flips *usually*, but not always
- Look for patterns and inputs where few bits change
- Theoretical attack on SHA
- Later on improved to practical attack [2004]

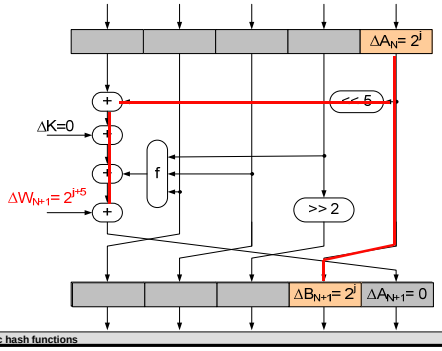
Perturbation

Step N

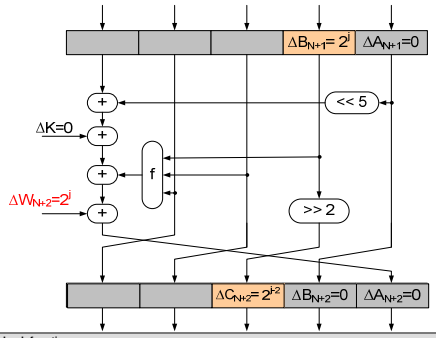


Correction 1

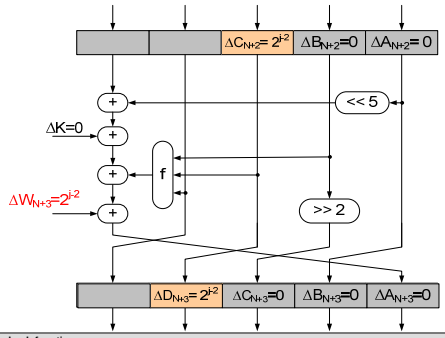
Step N+1



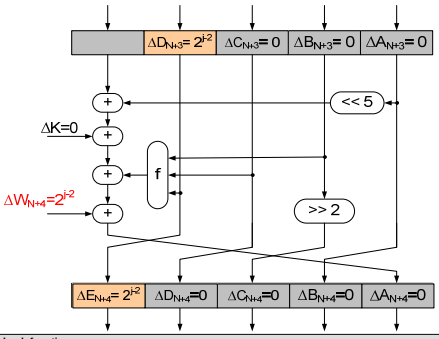
Correction 2
Step N+2



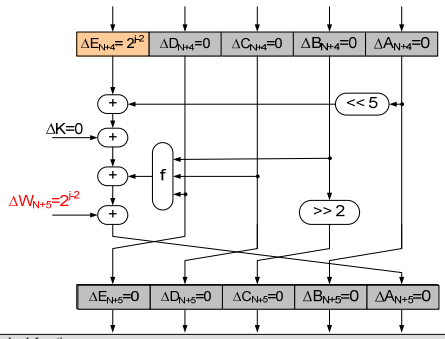
Correction 3
Step N+3



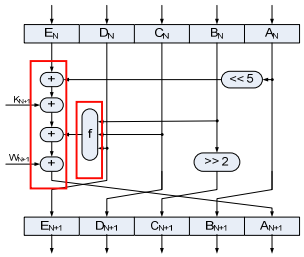
Correction 4
Step N+4



Correction 5
Step N+5



Conditions imposed by nonlinear elements



- Boolean function f
- Modular addition

Modular addition

- Linear except for carry effects
  - Carry = 0 with probability 1/2
  - Carry moves upwards only
  - No carry from MSB
- Requirement: difference propagation as with XOR

The Boolean functions

- 40 steps have XOR: linear!
- Bitwise parallel: every input bit affects at most 1 output bit

- We set as requirement: difference propagations as with XOR
  - High probability
  - Regularity
  - Easy to find good characteristics

Local collision

- Resynchronisation of internal state
- One perturbation and 5 corrections
- Creating local collisions is not so difficult

- Problem: message expansion

Effect of the message expansion

$$W_t = \begin{cases} M_t & for (0 \le t \le 15) \\ (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) <<< 1 & for (16 \le t \le 79) \end{cases}$$

- Every bit changed in a W_t influences other bits
- Impossible to find m_1, m_2 such that $E(m_1)$, $E(m_2)$ differ in 6 bits only

- Constructing a global collision = combination of local collisions = finding a good characteristic

Finding a good characteristic

- Linear approximation
  - Boolean functions → XOR
  - Additions → XOR
- Determine message differences that produce collision for the linear approximation
  - Set of linear equations
- Find solution with low weight

Linear code approach

- L-SHA:
  - E: Expansion matrix
  - A, B: State update matrices

$$h = A\ E\ m \oplus B\ iv$$

$$h_2 \oplus h_1 = A\ E\ (m_2 \oplus m_1) = 0$$

- A E: check matrix of linear code
- $(m_2 \oplus m_1)$: low-weight code word
  - Heuristic algorithms

Construct right pair

- With specified input difference
- Such that differences propagate as specified
- Conditions
  - On inputs of IF, MAJ
  - On inputs of addition

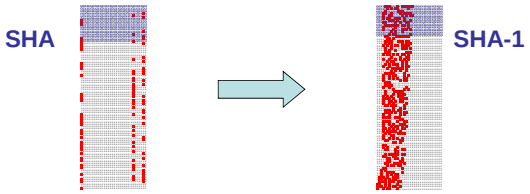
Example

IF(x,y,z) = $xy \oplus (1 \oplus x)z$

- Input difference: (0,0,1)
  - Desired output difference: 1
  - Condition: $x = 0$
- Input difference: (1,0,1)
  - Desired output difference: 0
  - Condition: $x \oplus z = 1$

Result: attack on SHA

- Low- weight patterns exist for SHA => break

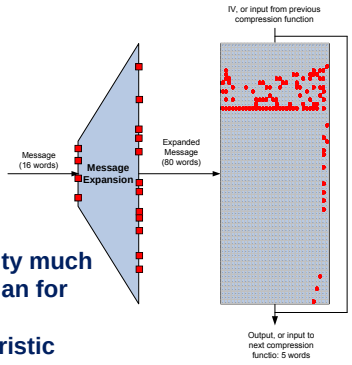


- For SHA-1: weight is too high

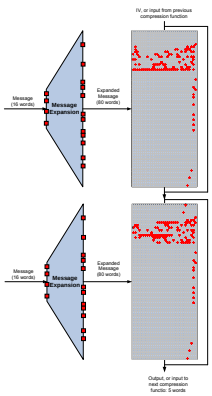
Improvements

- Better characteristics
  - 1-block → multi-block
  - Better suited for hash functions
- Construct right pairs
  - Instead of random trials
  - Message modification

A 1-block near-collision characteristic for SHA-1



Probability much higher than for collision characteristic



Attack on SHA-1

1st block:
- Near-collision

2nd block:
- Same message difference
- Same near-collision

2 near-collisions cancel out
→ **Collision**

Generalizing differential cryptanalysis

- Generalizing conditions

$x_i x_i^*$
0 0
0 1
1 0
1 1

Generalized Conditions - Notation

(x_i, x_i^*)	$(0,0)$	$(1,0)$	$(0,1)$	$(1,1)$
?	✓	✓	✓	✓
-	✓	-	-	✓
x	-	✓	✓	-
0	✓	-	-	-
u	-	✓	-	-
n	-	-	✓	-
1	-	-	-	✓
#	-	-	-	-

Automated search for generalized characteristic

Finding collisions: equation solving strategy

- Equations get more complex in later steps
- Guess critical bits in a favourable way
- Simplify remaining equations
- Maximize remaining number of solutions

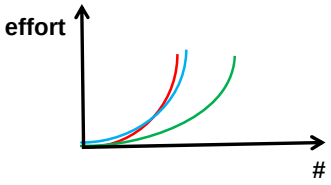
- But: don't spend too much effort

- Influenced by choice of characteristic

There are many different approaches

- **Neutral-bit technique [Biham and Chen]**
- **Advanced message modification [Wang et al.]**
- **Greedy method [De Cannière and Rechberger]**
- **Symbolic computation [Sugita et al.]**
- **Boomerang method [Joux and Peyrin]**

The gains of the methods are difficult to compare



- Exponential curves: extrapolation problematic
- What is the unit operation?
- Only way to be fair: implement and run it

The best result so far: collision for SHA-1 / 70 steps [DCMR 2007]

i	Message 1 (m_0), first block				Message 1 (m_1), second block			
1–4	3BB33AAE	85AECBBB	57A88417	8137CB9C	ABDDBEE2	42A20AC7	A915E04D	5063E027
5–8	4DE99220	5DCF12C7	726DD940	E37CE3D0	4DDF909A	E0020CF7	7FFD00F4	EFEFE0A7
9–12	23607799	239D2F1D	AAC76D94	E0009A1E	0FFBC2F0	C0DE16DF	01DDEG75	254429CD
13–16	C24DE871	5B7C3CD8	000359F5	90F9ED31	5F37A2C6	CD1963D3	FFCA1CB9	9642CB56
i	Message 2 (m_0^*), first block				Message 2 (m_1^*), second block			
1–4	ABB33ADE	35AECEE3	67A8841F	8137CBDF	3BDDBE92	F2A20A94	9915E045	5063E054
5–8	9DE99252	EB6F12D7	826BD92A	23F6E9FA	9DDF98E8	50020CE7	8FFD0096	2FEFE0E5
9–12	236077A9	C29B2F5F	9AC76BF4	C0009A5F	0FFBC2C0	20DE16FD	A1BDE615	C544299A
13–16	E24DE821	9B7C3C09	E0035997	30F9ED32	7F37A296	0D196362	1FCA1CCB	3642CB55
i	XOR-difference are the same for both blocks							
1–4	90000070	B0000C53	30000008	C0000043	90000070	B0000E53	30000008	C0000043
5–8	D0000072	B0000010	F0000062	C0000042	D0000072	B0000010	F0000062	C0000042
9–12	00000030	E0000042	20000060	E0000041	00000030	E0000042	20000060	E0000041
13–16	20000050	C0000041	E0000072	A0000003	20000050	C0000041	E0000072	A0000003
i	The colliding hash values							
1–5	151966D6	F7040D04	29E73695	C4D97E19	97DA712B			

Effort: 2^{44} calls to 70-step SHA-1 (OpenSSL impl.)

Progress on SHA-1

- 2005: Collision for 58-step variant
- 2006: Collision for 64-step variant
- 2007: Collision for 70-step variant

When will we have a collision for full SHA-1?

That depends on you!

Distributed computing effort

http://boinc.iaik.tugraz.at

Impact

- SHA-1 collisions can be found, so what?
- The colliding blocks have no meaning anyway
- We still have SHA-256 ...
- ... and the forthcoming SHA-3
- Many applications don't need collision resistance

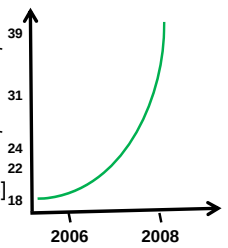
SHA-256

- More complicated expansion (nonlinear)
- Larger state, more complicated state update
- Breaking SHA-256 is a difficult problem

- A difficult problem is a problem that nobody works on (J. Massey)

Current results on SHA-256

- Collision for 18 steps [MPRR 2006]
- Collision for 21 steps, near-collision for 25 steps [NB 2/2008]
- Collision for 22 steps [SS 3/2008]
- Collision for 24 steps, near-collision for 31 steps [IMPRR 3/2008]
- Non-randomness for 39 steps [W 4/'08]

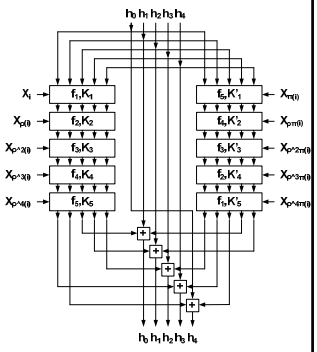


The RIPEMD-family

- RIPEMD
  - Results by Dobbertin (round reduced)
  - Collisions announced in 2004 by Wang et al.
- Introduction of two strengthened versions
  - RIPEMD-128
  - RIPEMD-160
- RIPEMD-160 is frequently recommended
  - Attacks extendable to RIPEMD-160?

RIPEMD-160 / 128

- RIPEMD-160
  - Output is 160 bits
  - Process message in 16 words (512-bit)
  - Uses 10 rounds of 16 steps in **2 parallel lines of 5**
- RIPEMD-128
  - Output 128 bits
  - Uses 8 rounds of 16 steps in **2 parallel lines of 4**

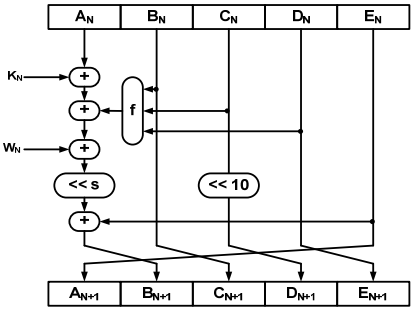


Step Function of RIPEMD-160

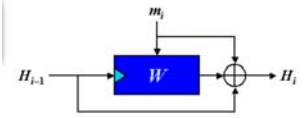
Modular Additions

Boolean Functions

Variable Rotation



Whirlpool



- 512-bit output
- Based on dedicated block cipher W (*AES like*)

Conclusions

- Hash functions are not block ciphers
- Differential cryptanalysis becomes more powerful
  - But, manually developing the attack becomes infeasible
- New design theory needed