A Look at the SHA-3 Competition: Design and Analysis of Hash Functions

Gaëtan Leurent
École Normale Supérieure
Paris, France

Universitity of Luxembourg
January 19, 2010
Outline

Introduction
   Hash functions
   The MD4 family

The SHA-3 competition
   New designs
   SIMD

New attacks on SHA-3 candidates
   Self-similarity attacks
   Cancellation cryptanalysis on generalized Feistels
What is a hash function?

- A public function with no structural properties.
  - Cryptographic strength without keys!

\[ F : \{0, 1\}^* \rightarrow \{0, 1\}^n \]
What is a hash function?

- A **public function with no structural properties.**
  - Cryptographic strength without keys!

- \( F : \{0, 1\}^* \rightarrow \{0, 1\}^n \)
Security goals

Preimage attack

Given \( F \) and \( \overline{H} \), find \( M \) s.t. \( F(M) = \overline{H} \).
Ideal security: \( 2^n \).

Second-preimage attack

Given \( F \) and \( M_1 \), find \( M_2 \neq M_1 \) s.t. \( F(M_1) = F(M_2) \).
Ideal security: \( 2^n \).

Collision attack

Given \( F \), find \( M_1 \neq M_2 \) s.t. \( F(M_1) = F(M_2) \).
Ideal security: \( 2^{n/2} \).

Ideal behaviour: random oracle.
## Security goals

### Preimage attack

Given $F$ and $\overline{H}$, find $M$ s.t. $F(M) = \overline{H}$.

Ideal security: $2^n$.

### Second-preimage attack

Given $F$ and $M_1$, find $M_2 \neq M_1$ s.t. $F(M_1) = F(M_2)$.

Ideal security: $2^n$.

### Collision attack

Given $F$, find $M_1 \neq M_2$ s.t. $F(M_1) = F(M_2)$.

Ideal security: $2^{n/2}$.

▶ Ideal behaviour: random oracle.
Security goals

Preimage attack

Given $F$ and $\overline{H}$, find $M$ s.t. $F(M) = \overline{H}$.
Ideal security: $2^n$.

Second-preimage attack

Given $F$ and $M_1$, find $M_2 \neq M_1$ s.t. $F(M_1) = F(M_2)$.
Ideal security: $2^n$.

Collision attack

Given $F$, find $M_1 \neq M_2$ s.t. $F(M_1) = F(M_2)$.
Ideal security: $2^{n/2}$.

▶ Ideal behaviour: random oracle.
Security definitions: difficulties

- A single function cannot be collision resistant.
  - Precomputation is allowed in standard security definition
  - Define a family of function

- Obvious relations between the security definitions do not hold.
  - Even more mess with families of functions!
Use as a one-way function

- **Unix password file**
  - Store $H(pw)$
  - Allow verification of the password without storing the password

- **One-time password**
  - User picks $x$ and server stores $y = H(H(H(x)))$
  - To authenticate, user sends a preimage of $y$
  - First authentication with $H(H(x))$, server now stores $H(H(x))$
  - Second authentication with $H(x)$
  - ...

Use as unique identifiers

- **Hash-and-sign**
  - Signature algorithm are costly
  - Sign \( H(m) \) instead of \( m \)

- **Commitment**
  - Alice commits to \( H(m) \) without revealing \( m \).
  - Later, she reveals \( m \).

- **Time-stamping**
  - Authority certifies that \( H(m) \) was known at time \( t_1 \)
  - \( m \) is revealed at time \( t_2 \)
  - Need a stronger notion that second-preimage resistance: herding attack
Breaking the structure of the input

▶ Key derivation

▶ Full Domain Hash
  ▶ Avoid the structural properties of RSA
  ▶ For a RSA key \((N, e, d)\)
  ▶ \(H\) a hash function to \(\mathbb{Z}_N\)
  ▶ Signature: \(s = H(m)^d\)
  ▶ Verification: \(s^e \equiv H(m)\)

▶ Rabin signatures
  ▶ Compute a square root of \(H(m)\) modulo an RSA number
  ▶ Broken if one can find \(H(m') = -H(m)\)
Use as a MAC

- Message Authentication Code
  - Symmetric signature

- Secret-prefix MAC
  - $\text{MAC}_k(m) = H(k\|m)$

- HMAC
  - $\text{HMAC}_k(m) = H(k \oplus \text{opad} \| H(k \oplus \text{ipad} \| m))$

- Challenge-response authentication
  - Alice sends a random challenge $r$
  - Bob replies with $\text{MAC}_k(r)$
Hash function design

- Build a smaller compression function, and iterate.
  - Cut the message in chunks $M_0, \ldots M_k$
  - $H_i = f(M_i, H_{i-1})$
  - $F(M) = H_k$
Security proof (Merkle, Damgård)

Theorem

If one finds a collision in the hash function, then one has a collision in the compression function.

- If $|M| \neq |M'|$, collision in last block.
- Else, look for last block with $H_i = H'_i$. 
Length extension attack

- Given the hash of an unknown message we can compute the hash of some related messages.

\[ H(M || M') = H_3 \] can be computed from \( H(M) = H_2 \) and \( M' \).

- Breaks secret-prefix MAC.

- Solution: use a finalisation function.
Length extension attack

- Given the hash of an unknown message we can compute the hash of some related messages.

\[ H(M || M') = H_3 \] can be computed from \( H(M) = H_2 \) and \( M' \).
- Breaks secret-prefix MAC.

Solution: use a finalisation function.
Length extension attack

- Given the hash of an unknown message we can compute the hash of some related messages.

\[ H(M \| M') = H_3 \] can be computed from \[ H(M) = H_2 \] and \[ M' \].
- Breaks secret-prefix MAC.

Solution: use a finalisation function.
**Length extension attack**

- Given the hash of an unknown message, we can compute the hash of some related messages.

\[
\begin{align*}
M_0 & \rightarrow f \rightarrow H_0 \\
M_1 & \rightarrow f \rightarrow H_1 \\
M_2 & \rightarrow f \rightarrow H_2 \\
\end{align*}
\]

\[\text{g}\]

\[
H(M || M') = H_3 \text{ can be computed from } H(M) = H_2 \text{ and } M'.
\]

- Breaks secret-prefix MAC.

- Solution: use a **finalisation function**.
Other attacks against Merkle-Damgård

- Long message second-preimage attack.
  - Given a message of length $2^k$, a preimage costs $2^{n-k}$.

- Multi-collision attack.
  - Build a set of $2^k$ colliding messages with time $k \times 2^n$.

- Herding attack.
  - Commit to a value, and choose the message later.
  - Cost about $2^{2n/3}$.

- Solution: use a bigger state.
Other attacks against Merkle-Damgård

- Long message second-preimage attack.
  - Given a message of length $2^k$, a preimage costs $2^{n-k}$.

- Multi-collision attack.
  - Build a set of $2^k$ colliding messages with time $k \times 2^n$.

- Herding attack.
  - Commit to a value, and choose the message later.
  - Cost about $2^{2n/3}$.

- Solution: use a bigger state.
MD family design

- MD4 designed by Rivest in 1990
- MD5 designed by Rivest in 1991
- One of the first dedicated hash function

Based on a dedicated block-cipher in Davies-Meyer mode:

\[ H_i = CF(H_{i-1}, M) = E_M(H_{i-1}) \oplus H_{i-1} \]
Introduction

The SHA-3 competition

New attacks on SHA-3 candidates

---

**MD family design**

Input:

\[
M \leftarrow \text{Message} \quad (A, B, C, D) \leftarrow \text{Chaining value}
\]

Output:

\[
(A + A', B + B', C + C', D + D')
\]

- 32-bit registers
- Simple operations
- Message expansion: permutation based
**MD4 design**

$$Q_i = (Q_{i-4} \oplus m_i \oplus k_i \oplus \Phi_i(Q_{i-1}, Q_{i-2}, Q_{i-3})) \ll s_i$$

- 48 steps (16 message words)
- Boolean functions: IF, MAJ, XOR
MD5 design

\[
Q_i = (Q_{i-4} \oplus m_i \oplus k_i \oplus \Phi_i(Q_{i-1}, Q_{i-2}, Q_{i-3})) \ll s_i \oplus Q_{i-1}
\]

- 64 steps (16 message words)
- Boolean functions: IF, MAJ, XOR, ONX
SHA-1 design

- Successor to MD4/MD5
- Designed by NIST in 1993
- Bigger hash output / bigger state
- Stronger message expansion
  - Linear code
  - $m_i = (m_{i-3} \oplus m_{i-8} \oplus m_{i-14} \oplus m_{i-16}) \ll 1$
SHA-1 design

\[ Q_i = Q_{i-5} \ll 30 \oplus m_i \oplus k_i \oplus \Phi_i(Q_{i-2}, Q_{i-3} \ll 30, Q_{i-4} \ll 30) \oplus Q_{i-1} \]

- 80 steps (16 message words)
- Boolean functions: IF, MAJ, XOR
Wang et. al’s attacks

▶ In 2004, new attacks against MD4, MD5, SHA-1, RIPEMD-0

▶ Based on a differential attack:
  ▶ Consider a pair of message with a small difference
  ▶ Try to control the propagation of the differences

▶ New ideas:
  ▶ Use a signed difference
  ▶ Use a set of necessary conditions
  ▶ Some conditions are easy to satisfy: message modification

▶ A lot of work by hand to find differential characteristic.
Main mistakes

**MD4** Not enough rounds

**MD5** A difference in the MSB can stay in the MSB

(Den Boer and Bosselaers, 1993)

\[
Q'_i = Q_i \oplus 2^{31}
\]

\[
Q_i = (Q_{i-4} \boxplus m_i \boxplus k_i \boxplus \Phi_i(Q_{i-1}, Q_{i-2}, Q_{i-3})) \ll^{s_i} \boxplus Q_{i-1}
\]

**SHA-1** Message expansion is a cyclic code

It is possible to shift a difference pattern

Used to build local collisions
Outline

Introduction
- Hash functions
- The MD4 family

The SHA-3 competition
- New designs
- SIMD

New attacks on SHA-3 candidates
- Self-similarity attacks
- Cancellation cryptanalysis on generalized Feistels
The SHA-3 competition

- Similar to the AES competition
- Organized by NIST

- Submission dead-line was October 2008: 64 candidates
- 51 valid submissions

- 14 in the second round (July 2009)
- 5 finalists in September 2010?
- Winner in 2012?
New designs

- Take into consideration recent advances in cryptanalysis
- Somewhat higher expectation that SHA-2
- Second round candidates seem quite solid...
- Wide diversity of designs
Mode of operation

- Sequential

\[ M_0 \xrightarrow{f} H_0 \xrightarrow{f} H_1 \xrightarrow{f} H_2 \]

- Tree-based

\[ M_0 \xrightarrow{f} M_1 \xrightarrow{f} M_2 \xrightarrow{f} M_3 \]

- Using the sponge construction

\[ M_0 \xrightarrow{P} H_0 \xrightarrow{H_0 \oplus M_1} H_0 \]

\[ M_0 \xrightarrow{P} H_0 \xrightarrow{H_1 \oplus M_2} H_1 \]

G. Leurent (ENS) A Look at the SHA-3 Competition: Design and Analysis of Hash Functions 25 / 68
Construction of the compression function

- From a (supposedly) perfect primitive
  - Most block cipher based designs, Keccak
  - Security proofs
    - By reduction
    - Indifferentiability proof

- From a weak primitive with a large state and a small message block
  - CubeHash, RadioGatún, Grindhal
  - Security proof only rules out generic attack

- By reduction to a class of hard problem
  - Usually slow
  - Security proof will be asymptotic
Construction of the compression function

- From a block cipher
  \[ H_i = E_M(H_{i-1}) \oplus H_{i-1} \]
  \text{Davies-Meyer}

- From a permutation
  \[ H_i = E_{H_{i-1}}(M) \oplus M \]
  \text{Matyas-Meyer-Oseas}

- Something else...
  - Shabal, Grøstl, Luffa, ...

- Something broken...
Construction of the compression function

- From a block cipher

\[ H_i = E_M(H_{i-1}) \oplus H_{i-1} \]
Davies-Meyer

\[ H_i = E_{H_{i-1}}(M) \oplus M \]
Matyas-Meyer-Oseas

- From a permutation

\[ H_i = \text{Tr}(P(H_{i-1} || M)) \]

- Something else...
  - Shabal, Grøstl, Luffa, ...

- Something broken...
Inside the compression function

- Feistel or SPN

- ARX
  - Additions, Rotation, XOR
  - Sometimes Shifts, Boolean function

- AES-based or AES-inspired
  - Can take advantage of Intel AES instructions

- Bitsliced
The design of SIMD

- SHA-3 candidate selected in the second round
- Built on the MD/SHA legacy
- Secure against differential attacks

Gaëtan Leurent, Pierre-Alain Fouque, Charles Bouillaguet
SIMD Is a Message Digest
Submission to the NIST SHA-3 competition
Main Features of SIMD

- **Security**
  - Strong message expansion
  - Proof of security against differential cryptanalysis

- **Parallelism**
  - Small scale parallelism (inside the compression function):
    - good for hardware / software with SIMD instructions
  - Can use two cores: message expansion / compression

- **Performance**
  - Very good on high-end desktops: 11 cycles/byte on Core2
  - Good if SIMD instructions are available:
    - SSE on x86, Altivec on PowerPC, IwMMXt on ARM, VIS on SPARC...
  - Drawback: no portable efficient implementation.
What mode of operation?

- Iterate a compression function
  - Easier to analyse

- Double the size of the state
  - Avoid generic attacks

- Finalisation function takes the message size as input
How to build the compression function?

- **Davies-Meyer:**
  \[ H_i = E_M(H_{i-1}) \oplus H_{i-1} \]
  - differential attack on \( C \)
  \[ \leadsto \text{related key attack on } E \]
  - Two inputs: \( H_{i-1} \) hard to control / \( M \) easy to control.

- **Matyas-Meyer-Oseas**
  \[ H_i = E_{H_{i-1}}(M) \oplus M \]
  - differential attack on \( C \)
  \[ \leadsto \text{differential attacks } E \]

- With DM, message expansion can reduce control over \( M \)
How to build the compression function?

- Davies-Meyer:
  \[ H_i = E_M(H_{i-1}) \oplus H_{i-1} \]
  - differential attack on \( C \)
  - \( \rightsquigarrow \) related key attack on \( E \)
  - Two inputs: \( H_{i-1} \) hard to control / \( M \) easy to control.

- Matyas-Meyer-Oseas
  \[ H_i = E_{H_{i-1}}(M) \oplus M \]
  - differential attack on \( C \)
  - \( \rightsquigarrow \) differential attacks \( E \)

- With DM, message expansion can reduce control over \( M \)
The Message Expansion

<table>
<thead>
<tr>
<th></th>
<th>Message block</th>
<th>Expanded message</th>
<th>Minimal distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMD-256</td>
<td>512 bits</td>
<td>4096 bits</td>
<td>520 bits</td>
</tr>
<tr>
<td>SIMD-512</td>
<td>1024 bits</td>
<td>8192 bits</td>
<td>1032 bits</td>
</tr>
</tbody>
</table>

- Provides resistance to differential attack
- Based on (error correcting) codes with a good minimal distance
- Concatenated code:
  - outer code gives a high word distance
  - inner code gives a high bit distance
Outer Code

Reed-Solomon code

- Interpret the input (\(k\) words) as a polynomial of degree \(k - 1\) over some finite field
- Evaluate on \(n\) points (\(n > k\))
- **MDS code**: minimal distance \(n - k + 1\)

<table>
<thead>
<tr>
<th></th>
<th>(k)</th>
<th>(n)</th>
<th>(d)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMD-256</td>
<td>64</td>
<td>128</td>
<td>65</td>
</tr>
<tr>
<td>SIMD-512</td>
<td>128</td>
<td>256</td>
<td>129</td>
</tr>
</tbody>
</table>

- **Efficiency**:
  - Compute with an FFT algorithm
  - Use the field \(\mathbb{F}_{257}\)
- Add a constant part: affine code
**Inner code**

We encode the output words of the FFT twice, through two different inner codes.

Very efficient codes, with a single 16-bit multiplication.

\[ I_{185} : \mathbb{F}_{257} \leftrightarrow \mathbb{Z}_{2^{16}} \]
\[ x \rightarrow 185 \otimes \tilde{x} \quad \text{where } -128 \leq \tilde{x} \leq 128 \text{ and } \tilde{x} = x \, (\text{mod } 257) \]

\[ I_{233} : \mathbb{F}_{257} \leftrightarrow \mathbb{Z}_{2^{16}} \]
\[ x \rightarrow 233 \otimes \tilde{x} \quad \text{where } -128 \leq \tilde{x} \leq 128 \text{ and } \tilde{x} = x \, (\text{mod } 257) \]

The magic constants 185 and 233 give a minimal distance of 4 bits. (also for signed difference)
How to build the compressing part?

- Unbalanced Feistels with simple bit-wise functions
  - Follow the MD/SHA family

- Use parallel Feistel to allow a bigger state
Introduction

The SHA-3 competition

New attacks on SHA-3 candidates

---

The SHA-3 competition

New attacks on SHA-3 candidates

---

NTT

16 steps

16 steps

4 steps

---

M

185

P₁

W

233

P₂

W

---

Hᵢ₋₁

Hᵢ
Outline

Introduction
Hash functions
The MD4 family

The SHA-3 competition
New designs
SIMD

New attacks on SHA-3 candidates
Self-similarity attacks
Cancellation cryptanalysis on generalized Feistels
Self-similarity attacks

- Generalization of the complementation property of DES
- Applied to SHA-3 candidate Lesamnta

Charles Bouillaguet, Orr Dunkelman, Gaëtan Leurent, and Pierre-Alain Fouque
Another Look at Complementation Properties
**DES's Complementation Property**

- If the key is bitwise complemented, so are all the subkeys.

- If the input to the round function is also bitwise complemented, the complementation is canceled.

- In other words, the input to the S-boxes is the same.

- **DES's complementation property:**

$$\begin{align*}
L_i & \quad \Rightarrow \quad L_{i+1} \\
R_i & \quad \Rightarrow \quad R_{i+1}
\end{align*}$$
**DES’s Complementation Property**

- If the key is bitwise complemented, so are all the subkeys.
  \[ K \rightarrow K_1, K_2, \ldots, K_{16} \text{ and} \]
  \[ \overline{K} \rightarrow \overline{K_1}, \overline{K_2}, \ldots, \overline{K_{16}} \]

- If the input to the round function is also bitwise complemented, the complementation is canceled.

- In other words, the input to the S-boxes is the same.

- **DES’s complementation property:**
**DES’s Complementation Property**

- If the key is bitwise complemented, so are all the subkeys.
  \[ K \rightarrow K_1, K_2, \ldots, K_{16} \text{ and } \overline{K} \rightarrow \overline{K}_1, \overline{K}_2, \ldots, \overline{K}_{16} \]

- If the input to the round function is also bitwise complemented, the complementation is canceled.

- In other words, the input to the S-boxes is the same.

- DES’s complementation property:
**DES’s Complementation Property**

- If the key is bitwise complemented, so are all the subkeys.
  \[ K \rightarrow K_1, K_2, \ldots, K_{16} \text{ and } \bar{K} \rightarrow \bar{K}_1, \bar{K}_2, \ldots, \bar{K}_{16} \]

- If the input to the round function is also bitwise complemented, the complementation is canceled.

- In other words, the input to the S-boxes is the same.
  And the output of the S-boxes.

- **DES’s complementation property:**
**DES’s Complementation Property**

- If the key is bitwise complemented, so are all the subkeys.
  \[ K \rightarrow K_1, K_2, \ldots, K_{16} \text{ and } \overline{K} \rightarrow \overline{K_1}, \overline{K_2}, \ldots, \overline{K_{16}} \]

- If the input to the round function is also bitwise complemented, the complementation is canceled.

- In other words, the input to the S-boxes is the same. And the output of the S-boxes.

- **DES’s complementation property:**
  \[ DES_K(P) = \overline{DES_{\overline{K}}(\overline{P})} \]
Examples in hash functions

- In CHI:
  \[ CF(H, M) = CF(H, \overline{M}) \]
  - This property is a collision in the compression function.

- In MD5:
  \[ CF(H, M) = CF(H \oplus 2^{32}, M \oplus 2^{32}) \text{ with probability } 2^{-48} \]
  - Basic property used in many attacks

- Can we find more?
  - Look for simple transformations \( \phi, \psi \) and \( \theta \) such that:
    \[ \theta(CF(X, M)) = CF(\phi(X), \psi(M)) \]
Lesamnta

- Davies-Meyer with an MMO compression function
- Generalized Feistel
- Round function is AES-based

Shoichi Hirose, Hidenori Kuwakado, Hirotaka Yoshida
SHA-3 Proposal: Lesamnta
Submission to the NIST SHA-3 competition
Lesamnta (cont.)

\[ X_{i+4} = X_i \oplus F(X_{i+1} \oplus K_{i+3}) \]
\[ K_{i+4} = K_i \oplus G(K_{i+1} \oplus R_{i+3}). \]

- Message loaded to \( K_{-3}, K_{-2}, K_{-1}, K_0 \)
- Chaining value loaded to \( X_{-3}, X_{-2}, X_{-1}, X_0 \)
- \( F \) and \( G \) AES-based
Some Interesting Properties of AES [LSWD04]
Some Interesting Properties of AES [LSWD04]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>SB</th>
</tr>
</thead>
<tbody>
<tr>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
<td>i</td>
</tr>
<tr>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
<td>m</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>e</td>
<td>f</td>
<td>g</td>
<td>h</td>
<td>k</td>
</tr>
<tr>
<td>m</td>
<td>n</td>
<td>o</td>
<td>p</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>α</td>
</tr>
<tr>
<td>ε</td>
</tr>
<tr>
<td>i</td>
</tr>
<tr>
<td>ν</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>C</th>
<th>D</th>
<th>A</th>
<th>B</th>
<th>SB</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>H</td>
<td>E</td>
<td>F</td>
<td>k</td>
</tr>
<tr>
<td>K</td>
<td>L</td>
<td>I</td>
<td>J</td>
<td>o</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>c</th>
<th>d</th>
<th>a</th>
<th>b</th>
<th>SR</th>
</tr>
</thead>
<tbody>
<tr>
<td>g</td>
<td>h</td>
<td>e</td>
<td>f</td>
<td>i</td>
</tr>
<tr>
<td>n</td>
<td>o</td>
<td>p</td>
<td>m</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>γ</td>
</tr>
<tr>
<td>η</td>
</tr>
<tr>
<td>λ</td>
</tr>
<tr>
<td>ο</td>
</tr>
</tbody>
</table>
Some Interesting Properties of AES [LSWD04]

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A</th>
<th>B</th>
<th>SB</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>b</th>
<th>SR</th>
<th>a</th>
<th>b</th>
<th>a</th>
<th>b</th>
<th>MC</th>
<th>α</th>
<th>β</th>
<th>α</th>
<th>β</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>D</td>
<td>C</td>
<td>D</td>
<td></td>
<td>c</td>
<td>d</td>
<td>c</td>
<td>d</td>
<td></td>
<td>d</td>
<td>c</td>
<td>d</td>
<td>c</td>
<td></td>
<td>γ</td>
<td>δ</td>
<td>γ</td>
<td>δ</td>
</tr>
<tr>
<td>E</td>
<td>F</td>
<td>E</td>
<td>F</td>
<td></td>
<td>e</td>
<td>f</td>
<td>e</td>
<td>f</td>
<td></td>
<td>e</td>
<td>f</td>
<td>e</td>
<td>f</td>
<td></td>
<td>η</td>
<td>θ</td>
<td>η</td>
<td>θ</td>
</tr>
<tr>
<td>G</td>
<td>H</td>
<td>G</td>
<td>H</td>
<td></td>
<td>g</td>
<td>h</td>
<td>g</td>
<td>h</td>
<td></td>
<td>h</td>
<td>g</td>
<td>h</td>
<td>g</td>
<td></td>
<td>ε</td>
<td>ζ</td>
<td>ε</td>
<td>ζ</td>
</tr>
</tbody>
</table>
Some Interesting Properties of Lesamnta’s $F$ and $G$

- Lesamnta’s $F$ posses similar properties:
  \[
  F(X, Y) = (Z, W) \Rightarrow F(Y, X) = (W, Z).
  \]

- The same is true for $G$ as well:
  \[
  G(X, Y) = (Z, W) \Rightarrow G(Y, X) = (W, Z).
  \]

- Let \( \langle a, b \rangle = (a, b) \)
  
  - \( F(\langle x \rangle) = F(x) \)
  
  - \( G(\langle x \rangle) = G(x) \)
Complementation-like property in Lesamnta

- Can we use this in the key-schedule?

- No, because of the constants
- On the other hand, the constants are almost symmetric...
Complementation-like property in Lesamnta

▶ Can we use this in the key-schedule?

▶ No, because of the constants

▶ On the other hand, the constants are almost symmetric...
Complementation-like property in Lesamnta

- Can we use this in the key-schedule?

- No, because of the constants
- On the other hand, the constants are almost symmetric...
Lesamnta’s constants

- \( R_i = (2i, 2i + 1) \)
- \( R_i \oplus \overrightarrow{R_i} = (1, 1) \)
- Let \( \widehat{(a, b)} = (a, b) \oplus (1, 1) = (b \oplus 1, a \oplus 1) \)
- \( \tilde{R}_i = R_i \)
Lesamnta’s constants

- $R_i = (2i, 2i + 1)$
- $R_i \oplus \overrightarrow{R_i} = (1, 1)$
- Let $\widetilde{(a, b)} = (a, b) \oplus (1, 1) = (b \oplus 1, a \oplus 1)$
- $\widetilde{R_i} = R_i$
Complementation-like property in Lesamnta, part II

- Can we use this in the key-schedule?

\[ \tilde{K}_i \oplus R_{i+3} = K_{i+1} \oplus R_{i+3} \]

\[ G(\tilde{K}_{i+1} \oplus R_{i+3}) = G(K_{i+1} \oplus R_{i+3}) \]

\[ \tilde{K}_i \oplus G(\tilde{K}_{i+1} \oplus R_{i+3}) = K_i \oplus G(K_{i+1} \oplus R_{i+3}) = \tilde{K}_{i+4} \]
Complementation-like property in Lesamnta, part II

- Can we use this in the key-schedule?

\[ \tilde{K}_i \times \tilde{K}_{i+1} \times \tilde{K}_{i+2} \times \tilde{K}_{i+3} \]

\[ \tilde{R}_{i+3} \]

\[ G(\tilde{R}_{i+3} \oplus \tilde{K}_{i+1}) = G(\tilde{K}_{i+1} \oplus \tilde{R}_{i+3}) \]

\[ \tilde{K}_i \oplus G(\tilde{K}_{i+1} \oplus R_{i+3}) = K_i \oplus G(K_{i+1} \oplus R_{i+3}) = \tilde{K}_{i+4} \]
Complementation-like property in Lesamnta, part II

- Can we use this in the full compression function?

- $\tilde{K}_i \rightarrow \tilde{\tilde{K}}_i$

- $\tilde{X}_i \oplus \tilde{K}_{i+3} = \tilde{X}_{i+1} \oplus \tilde{K}_{i+3}$

- $F(\tilde{X}_{i+1} \oplus \tilde{K}_{i+3}) = F(\tilde{X}_{i+1} \oplus \tilde{K}_{i+3})$

- $\tilde{X}_i \oplus F(\tilde{X}_{i+1} \oplus \tilde{K}_{i+3}) = X_i \oplus F(X_{i+1} \oplus K_{i+3}) = \tilde{X}_{i+4}$
Complementation-like property in Lesamnta, part II

- Can we use this in the full compression function?

\[ R_{i+3} \]
\[ G \]
\[ \tilde{K}_i \]
\[ K_{i+1} \]
\[ K_{i+2} \]
\[ K_{i+3} \]
\[ K_{i+4} \]
\[ \tilde{X}_i \]
\[ X_{i+1} \]
\[ X_{i+2} \]
\[ X_{i+3} \]
\[ X_{i+4} \]

- \( K_i \rightarrow \tilde{K}_i \)
- \( \tilde{X}_{i+1} \oplus \tilde{K}_{i+3} = X_{i+1} \oplus K_{i+3} \)
- \( F(\tilde{X}_{i+1} \oplus \tilde{K}_{i+3}) = F(X_{i+1} \oplus K_{i+3}) \)
- \( \tilde{X}_i \oplus F(\tilde{X}_{i+1} \oplus \tilde{K}_{i+3}) = X_i \oplus F(X_{i+1} \oplus K_{i+3}) = \tilde{X}_{i+4} \)
Some Really Interesting Property of Lesamnta

- $\text{CF}(\tilde{X}, \tilde{K}) = \text{CF}(X, K)$

- If $\tilde{X} = X$ and $\tilde{K} = K$, then $\text{CF}(X, K) = \text{CF}(X, K)$
  - The output is in a subspace of size $2^{n/2}$.

- Collision in the compression function in time $2^{n/4}$

- Second-preimage on weak messages

- Improved herding attack
Some Really Interesting Property of Lesamnta

- \( CF(\tilde{X}, \tilde{K}) = CF(X, K) \)

- If \( \tilde{X} = X \) and \( \tilde{K} = K \), then \( CF(X, K) = CF(X, K) \)
  - The output is in a subspace of size \( 2^{n/2} \).

- Collision in the compression function in time \( 2^{n/4} \)

- Second-preimage on weak messages

- Improved herding attack
Some Really Interesting Property of Lesamnta

- \( \text{CF}(\tilde{X}, \tilde{K}) = \text{CF}(X, K) \)

- If \( \tilde{X} = X \) and \( \tilde{K} = K \), then \( \text{CF}(X, K) = \text{CF}(X, K) \)
  - The output is in a subspace of size \( 2^{n/2} \).

- Collision in the compression function in time \( 2^{n/4} \)

- Second-preimage on weak messages

- Improved herding attack
Self-similarity property

- Sometimes, simple relation can go through a function

- The constant are used to avoid this...
  - But sometimes the constants are weak
Cancellation cryptanalysis on generalized Feistels

- Cancel the effect of the non-linear components
  Using twice the same input pairs

- Fix some parts of the state to reduce the diffusion

Charles Bouillaguet, Orr Dunkelman, Gaëtan Leurent and Pierre-Alain Fouque
Attacks on Hash Functions based on Generalized Feistel
Application to Reduced-Round Lesamnta and SHA\text{vite-3}_{512}

Praveen Gauravaram, Gaëtan Leurent, Florian Mendel, María Naya-Plasencia, Thomas Peyrin, Christian Rechberger, and Martin Schläffer
Cryptanalysis of the 10-Round Hash and Full Compression Function of SHA\text{vite-3}_{512}
Cancellation cryptanalysis

- Generalized Feistel with slow diffusion

\[
\begin{align*}
S_i & \quad T_i & \quad U_i & \quad V_i \\
S_{i+1} & \quad T_{i+1} & \quad U_{i+1} & \quad V_{i+1}
\end{align*}
\]

Lesamnta

\[
\begin{align*}
S_i & \quad T_i & \quad U_i & \quad V_i & \quad K_i \\
S_{i+1} & \quad T_{i+1} & \quad U_{i+1} & \quad V_{i+1}
\end{align*}
\]

\(F_i(x) = F(k_i \oplus x)\)
- Can sometimes deal with more keys (see SHA\textit{v}ite-\textit{3512})

- Hash function setting
  - Some results apply to block ciphers.
Cancellation cryptanalysis

- Generalized Feistel with slow diffusion

\[
S_i \quad T_i \quad U_i \quad V_i
\]

\[
S_{i+1} \quad T_{i+1} \quad U_{i+1} \quad V_{i+1}
\]

Lesamnta

\[
\text{SHA}vite-3_{512}
\]

\[F_i(x) = F(k_i \oplus x)\]

- Can sometimes deal with more keys (see SHA\textit{vite-3}_{512})

- Hash function setting

- Some results apply to block ciphers.
Cancellation cryptanalysis

- Generalized Feistel with slow diffusion

\[
\begin{align*}
S_i & \quad T_i & \quad U_i & \quad V_i \\
\downarrow & & \downarrow & \downarrow & \downarrow \\
S_{i+1} & \quad T_{i+1} & \quad U_{i+1} & \quad V_{i+1} \\
K_i & \\
\end{align*}
\]

Lesamnta

\[
\begin{align*}
\begin{array}{c}
F_i(x) = F(k_i \oplus x) \\
\end{array}
\end{align*}
\]

- Can sometimes deal with more keys (see SHA\textit{vite-3}_{512})

- Hash function setting
  - Some results apply to block ciphers.
Feistel design

- Ideal: each $F_i$ is an independent ideal function/permutation

- In practice: $F_i(x) = F(k_i \oplus x)$ with a fixed $F$

Properties of $F_i(x) = F(k_i \oplus x)$

(i) $\exists c_{i,j} : \forall x, F_i(x \oplus c_{i,j}) = F_j(x)$.

(ii) $\forall \alpha, \# \{x : F_i(x) \oplus F_j(x) = \alpha\}$ is even

(iii) $\bigoplus_x F_k(F_i(x) \oplus F_j(x)) = 0$

- $c_{ij} = k_i \oplus k_j$
Feistel design

- Ideal: each $F_i$ is an independent ideal function/permutation
- In practice: $F_i(x) = F(k_i \oplus x)$ with a fixed $F$

Properties of $F_i(x) = F(k_i \oplus x)$

1. $\exists c_{i,j} : \forall x, F_i(x \oplus c_{i,j}) = F_j(x)$.
2. $\forall \alpha, \# \{ x : F_i(x) \oplus F_j(x) = \alpha \}$ is even
3. $\bigoplus_x F_k(F_i(x) \oplus F_j(x)) = 0$

- $c_{ij} = k_i \oplus k_j$
Feistel design

- Ideal: each $F_i$ is an independent ideal function/permutation
- In practice: $F_i(x) = F(k_i \oplus x)$ with a fixed $F$

**Properties of $F_i(x) = F(k_i \oplus x)$**

(i) $\exists c_{i,j} : \forall x$, $F_i(x \oplus c_{i,j}) = F_j(x)$.

(ii) $\forall \alpha$, $\# \{x : F_i(x) \oplus F_j(x) = \alpha\}$ is even

(iii) $\bigoplus_x F_k(F_i(x) \oplus F_j(x)) = 0$

- $c_{i,j} = k_i \oplus k_j$
Feistel design

- Ideal: each $F_i$ is an independent ideal function/permutation
- In practice: $F_i(x) = F(k_i \oplus x)$ with a fixed $F$

### Properties of $F_i(x) = F(k_i \oplus x)$

(i) $\exists c_{i,j} : \forall x, F_i(x \oplus c_{i,j}) = F_j(x)$.

(ii) $\forall \alpha, \# \{x : F_i(x) \oplus F_j(x) = \alpha\}$ is even

(iii) $\bigoplus_x F_k(F_i(x) \oplus F_j(x)) = 0$

- $c_{ij} = k_i \oplus k_j$
**The cancellation property**

<table>
<thead>
<tr>
<th>$i$</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$a$</td>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
</tr>
<tr>
<td>1</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
<td>$b$</td>
<td>$c$</td>
</tr>
<tr>
<td>2</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
<td>$b$</td>
</tr>
<tr>
<td>3</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
</tr>
<tr>
<td>4</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
</tr>
<tr>
<td>5</td>
<td>$F_4(F_1(b) \oplus c) \oplus F_0(c) \oplus d$</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
</tr>
</tbody>
</table>

**Round 5**

$F_4(F_1(b) \oplus c) \oplus F_0(c)$

Cancel if $F_1(b) = K_0 \oplus K_4$

$\Rightarrow b \triangleq F_1^{-1}(K_0 \oplus K_4)$

- If $b$ is fixed to the right value, simple expressions.
- Easy in hash function.
## The cancellation property

<table>
<thead>
<tr>
<th>( i )</th>
<th>( S_i )</th>
<th>( T_i )</th>
<th>( U_i )</th>
<th>( V_i )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( a )</td>
<td>( b )</td>
<td>( c )</td>
<td>( d )</td>
</tr>
<tr>
<td>1</td>
<td>( F_0(c) \oplus d )</td>
<td>( a )</td>
<td>( b )</td>
<td>( c )</td>
</tr>
<tr>
<td>2</td>
<td>( F_1(b) \oplus c )</td>
<td>( F_0(c) \oplus d )</td>
<td>( a )</td>
<td>( b )</td>
</tr>
<tr>
<td>3</td>
<td>( F_2(a) \oplus b )</td>
<td>( F_1(b) \oplus c )</td>
<td>( F_0(c) \oplus d )</td>
<td>( a )</td>
</tr>
<tr>
<td>4</td>
<td>( F_3(F_0(c) \oplus d) \oplus a )</td>
<td>( F_2(a) \oplus b )</td>
<td>( F_1(b) \oplus c )</td>
<td>( F_0(c) \oplus d )</td>
</tr>
<tr>
<td>5</td>
<td>( F_4(F_1(b) \oplus c) \oplus F_3(c) \oplus d )</td>
<td>( F_3(F_0(c) \oplus d) \oplus a )</td>
<td>( F_2(a) \oplus b )</td>
<td>( F_1(b) \oplus c )</td>
</tr>
</tbody>
</table>

### round 5

\[
F_4(F_1(b) \oplus c) \oplus F_0(c)
\]

Cancel if \( F_1(b) = K_0 \oplus K_4 \)

\[
\Rightarrow b \overset{\triangle}{=} F_1^{-1}(K_0 \oplus K_4)
\]

- If \( b \) is fixed to the right value, simple expressions.
- Easy in hash function.
The cancellation property

<table>
<thead>
<tr>
<th>i</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
</tr>
<tr>
<td>1</td>
<td>$F_0(c) \oplus d$</td>
<td>a</td>
<td>b</td>
<td>c</td>
</tr>
<tr>
<td>2</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>a</td>
<td>b</td>
</tr>
<tr>
<td>3</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>a</td>
</tr>
<tr>
<td>4</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
</tr>
<tr>
<td>5</td>
<td>$F_4(F_1(b) \oplus c) \oplus F_0(c)$</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
</tr>
</tbody>
</table>

**round 5**  
$F_4(F_1(b) \oplus c) \oplus F_0(c)$  
Cancel if $F_1(b) = K_0 \oplus K_4$  
$\Rightarrow b \overset{\triangle}{=} F_1^{-1}(K_0 \oplus K_4)$

- If $b$ is fixed to the right value, simple expressions.
- Easy in hash function.
The cancellation property

<table>
<thead>
<tr>
<th>i</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$a$</td>
<td>$b$</td>
<td>$c$</td>
<td>$d$</td>
</tr>
<tr>
<td>1</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
<td>$b$</td>
<td>$c$</td>
</tr>
<tr>
<td>2</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
<td>$b$</td>
</tr>
<tr>
<td>3</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
<td>$a$</td>
</tr>
<tr>
<td>4</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
<td>$F_0(c) \oplus d$</td>
</tr>
<tr>
<td>5</td>
<td>$F_4(F_1(b) \oplus c) \oplus F_0(c)$</td>
<td>$F_3(F_0(c) \oplus d) \oplus a$</td>
<td>$F_2(a) \oplus b$</td>
<td>$F_1(b) \oplus c$</td>
</tr>
</tbody>
</table>

**round 5** $F_4(F_1(b) \oplus c) \oplus F_0(c)$

Cancel if $F_1(b) = K_0 \oplus K_4$

$\Rightarrow b = F_1^{-1}(K_0 \oplus K_4)$

- If $b$ is fixed to the right value, simple expressions.
- Easy in hash function.
Attack Overview

- Choose one part of the output
  - Preimage and collision attacks.

- Mostly generic in the round function.

Basic algorithm

- Start from a state in the middle
- Fix some parts of the state to satisfy the cancellation conditions.
- One output word will have a relatively simple expression.
- Invert the expression to choose one word of the output.
Result overview

- Attacks on reduced \textit{Lesamnta}
  - 24 rounds out of 32: collision and preimage
  - previous attacks: 16 rounds

- Attack on reduced \textit{SHAvite-3}_{512}
  - 10 rounds out of 14: preimage
  - previous attacks: 8 rounds

- Pseudo-attack on full \textit{SHAvite-3}_{512} compression function
  - chosen-salt chosen-counter preimage
Result overview

- Attacks on reduced Lesamnta
  - 24 rounds out of 32: collision and preimage
  - previous attacks: 16 rounds

- Attack on reduced SHA\textit{vite}-3\textsubscript{512}
  - 10 rounds out of 14: preimage
  - previous attacks: 8 rounds

- Pseudo-attack on full SHA\textit{vite}-3\textsubscript{512} compression function
  - chosen-salt chosen-counter preimage
Result overview

- Attacks on reduced Lesamnta
  - 24 rounds out of 32: collision and preimage
  - previous attacks: 16 rounds

- Attack on reduced SHA\textit{v}ite-3_{512}
  - 10 rounds out of 14: preimage
  - previous attacks: 8 rounds

- Pseudo-attack on full SHA\textit{v}ite-3_{512} compression function
  - chosen-salt chosen-counter preimage
SHAvite-3\textsubscript{512}

\[ S_i \rightarrow T_i \rightarrow U_i \rightarrow V_i \rightarrow K_i \]

\[ F \]

\[ S_{i+1} \rightarrow T_{i+1} \rightarrow U_{i+1} \rightarrow V_{i+1} \]

- 14 rounds
- Davies-Meyer (message is the key)
- \[ F_i(x) = AES(AES(AES(x \oplus k_i^0) \oplus k_i^1) \oplus k_i^2) \oplus k_i^3 \]

Eli Biham and Orr Dunkelman
The SHAvite-3 Hash Function
Submission to the NIST SHA-3 competition
Cancellation differential path: SHA\textit{v}i\textit{t}e-3\textsubscript{512}

<table>
<thead>
<tr>
<th>$i$</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>?</td>
<td>$x$</td>
<td>?</td>
<td>$x_2$</td>
</tr>
<tr>
<td>-3</td>
<td>$x_2$</td>
<td>$x_1$</td>
<td>$x$</td>
<td>-</td>
</tr>
<tr>
<td>-2</td>
<td>-</td>
<td>-</td>
<td>$x_1$</td>
<td>$x$</td>
</tr>
<tr>
<td>-1</td>
<td>$x$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>$x$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>$y$</td>
<td>$x$</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>$z$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>3</td>
<td>$x$</td>
<td>$w$</td>
<td>$z$</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>?</td>
<td>$w$</td>
<td>$z$</td>
</tr>
<tr>
<td>5</td>
<td>$z$</td>
<td>?</td>
<td>?</td>
<td>-</td>
</tr>
<tr>
<td>FF</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>$x_2$</td>
</tr>
</tbody>
</table>

$x \rightarrow y$

$x \rightarrow y, z \rightarrow w$

$z \rightarrow w$

- Same attack as previously
- But...
- $F$ has many keys...
### Cancellation differential path: SHA\textit{vite}-3\textsubscript{512}

<table>
<thead>
<tr>
<th>$i$</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>?</td>
<td>$x$</td>
<td>?</td>
<td>$x_2$</td>
</tr>
<tr>
<td>-3</td>
<td>$x_2$</td>
<td>$x_1$</td>
<td>$x$</td>
<td>-</td>
</tr>
<tr>
<td>-2</td>
<td>-</td>
<td>-</td>
<td>$x_1$</td>
<td>$x$</td>
</tr>
<tr>
<td>-1</td>
<td>$x$</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>$x$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>$y$</td>
<td>$x$</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>$z$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>3</td>
<td>$x$</td>
<td>$w$</td>
<td>$z$</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>?</td>
<td>$w$</td>
<td>$z$</td>
</tr>
<tr>
<td>5</td>
<td>$z$</td>
<td>?</td>
<td>?</td>
<td>-</td>
</tr>
<tr>
<td>FF</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>$x_2$</td>
</tr>
</tbody>
</table>

- $x \rightarrow y$
- $x \rightarrow y, z \rightarrow w$
- $z \rightarrow w$

- Same attack as previously
- But...
- $F$ has many keys...
Cancellation differential path: SHA\text{vite}-3_{512}

<table>
<thead>
<tr>
<th>$i$</th>
<th>$S_i$</th>
<th>$T_i$</th>
<th>$U_i$</th>
<th>$V_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-4</td>
<td>?</td>
<td>x</td>
<td>?</td>
<td>$x_2$</td>
</tr>
<tr>
<td>-3</td>
<td>$x_2$</td>
<td>$x_1$</td>
<td>x</td>
<td>-</td>
</tr>
<tr>
<td>-2</td>
<td>-</td>
<td>-</td>
<td>$x_1$</td>
<td>x</td>
</tr>
<tr>
<td>-1</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>$x$</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1</td>
<td>-</td>
<td>$y$</td>
<td>$x$</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>$z$</td>
<td>$y$</td>
<td>$x$</td>
</tr>
<tr>
<td>3</td>
<td>$x$</td>
<td>$w$</td>
<td>$z$</td>
<td>-</td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>?</td>
<td>$w$</td>
<td>$z$</td>
</tr>
<tr>
<td>5</td>
<td>$z$</td>
<td>?</td>
<td>?</td>
<td>-</td>
</tr>
<tr>
<td>FF</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>$x_2$</td>
</tr>
</tbody>
</table>

$x \rightarrow y$

$x \rightarrow y, z \rightarrow w$

$z \rightarrow w$

- Same attack as previously
- But...
- $F$ has many keys...
## Cancellation path values: SHA\textit{v}ite-3\textsubscript{512}

<table>
<thead>
<tr>
<th>Round</th>
<th>$X_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X_0$</td>
<td>$d \oplus F_3(a) \oplus F'_1(a \oplus F_2(b \oplus F'_3(c)))$</td>
</tr>
<tr>
<td>$Y_0$</td>
<td>$b \oplus F_3(c) \oplus F_1(c \oplus F'_2(d \oplus F_3(a)))$</td>
</tr>
<tr>
<td>$X_1$</td>
<td>$c \oplus F'_2(d \oplus F_3(a))$</td>
</tr>
<tr>
<td>$Y_1$</td>
<td>$a \oplus F_2(b \oplus F'_3(c))$</td>
</tr>
<tr>
<td>$X_2$</td>
<td>$b \oplus F'_3(c)$</td>
</tr>
<tr>
<td>$Y_2$</td>
<td>$d \oplus F_3(a)$</td>
</tr>
<tr>
<td>$X_3$</td>
<td>$a$</td>
</tr>
<tr>
<td>$Y_3$</td>
<td>$c$</td>
</tr>
<tr>
<td>$X_4$</td>
<td>$d$</td>
</tr>
<tr>
<td>$Y_4$</td>
<td>$b$</td>
</tr>
<tr>
<td>$X_5$</td>
<td>$c \oplus F_4(d)$</td>
</tr>
<tr>
<td>$Y_5$</td>
<td>$a \oplus F'_4(b)$</td>
</tr>
<tr>
<td>$X_6$</td>
<td>$b \oplus F_5(c \oplus F_4(d))$</td>
</tr>
<tr>
<td>$Y_6$</td>
<td>$d \oplus F'_5(a \oplus F'_4(b))$</td>
</tr>
<tr>
<td>$X_7$</td>
<td>$a \oplus F'_4(b) \oplus F_6(b \oplus F_5(c \oplus F_4(d)))$</td>
</tr>
<tr>
<td>$Y_7$</td>
<td>$c \oplus F_4(d) \oplus F'_6(d \oplus F_5(a \oplus F'_4(b)))$</td>
</tr>
<tr>
<td>$X_8$</td>
<td>$d \oplus F'_5(a \oplus F'_4(b)) \oplus F_7(a)$</td>
</tr>
<tr>
<td>$X_9$</td>
<td>$c \oplus F_4(d) \oplus F'_6(d \oplus F_5(a \oplus F'_4(b))) \oplus F_8(d \oplus F'_5(a \oplus F'_4(b)) \oplus F_7(a)$</td>
</tr>
</tbody>
</table>
**Message conditions: SHA\text{v}i\text{t}e-3_{512}**

**Round 7**

\[ F'_4(b) \oplus F_6(b \oplus F_5(c \oplus F_4(d))) \]

Cancel if \( F_5(c \oplus F_4(d)) = k^0_{1,4} \oplus k^0_{0,6} \)
and \( (k^1_{1,4}, k^2_{1,4}, k^3_{1,4}) = (k^1_{0,6}, k^2_{0,6}, k^3_{0,6}) \).

**Round 9**

\[ F'_6(d \oplus F'_5(a \oplus F'_4(b))) \oplus F_8(d \oplus F'_5(a \oplus F'_4(b)) \oplus F_7(a)) \]

Cancel if \( F_7(a) = k^0_{1,6} \oplus k^0_{0,8} \)
and \( (k^1_{1,6}, k^2_{1,6}, k^3_{1,6}) = (k^1_{0,8}, k^2_{0,8}, k^3_{0,8}) \).
Message conditions: SHA\textit{v}ite-3\textsubscript{512}

**Round 7**
\[
F'_4(b) \oplus F_6(b \oplus F_5(c \oplus F_4(d)))
\]
Cancel if \(F_5(c \oplus F_4(d)) = k_{1,4}^0 \oplus k_{0,6}^0\)
and \((k_{1,4}^1, k_{1,4}^2, k_{1,4}^3) = (k_{0,6}^1, k_{0,6}^2, k_{0,6}^3)\).

**Round 9**
\[
F'_6(d \oplus F'_5(a \oplus F'_4(b))) \oplus F_8(d \oplus F'_5(a \oplus F'_4(b)) \oplus F_7(a))
\]
Cancel if \(F_7(a) = k_{1,6}^0 \oplus k_{0,8}^0\)
and \((k_{1,6}^1, k_{1,6}^2, k_{1,6}^3) = (k_{0,8}^1, k_{0,8}^2, k_{0,8}^3)\).
Attacking the key schedule

- We can build a chaining value satisfying the 6 conditions with cost $2^{224}$.

- Each chaining value can be used $2^{128}$ times to fix 128 bits of the output.

- Attacks on 9-round \texttt{SHA\textit{v}i}te-3\textsubscript{512}:
  - Free-start preimage with complexity $2^{480}$
  - Preimage with complexity $2^{497}$. 
## Adding more rounds

<table>
<thead>
<tr>
<th>$i$</th>
<th>$A_i$</th>
<th>$B_i$</th>
<th>$C_i$</th>
<th>$D_i$</th>
<th>Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>?</td>
<td>$B_3$</td>
<td>?</td>
<td>?</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>?</td>
<td>?</td>
<td>$B_3$</td>
<td>$D_4$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>$D_4$</td>
<td>$B_5$</td>
<td>?</td>
<td>$B_3 + F'_4(D_4)$</td>
<td>$F_5(B_5) = 0$</td>
</tr>
<tr>
<td>6</td>
<td>$B_5 + F'_4(D_4)$</td>
<td>$D_4$</td>
<td>$B_3$</td>
<td>$D_6$</td>
<td>$R K_6 = R K'_4$</td>
</tr>
<tr>
<td>7</td>
<td>$D_6$</td>
<td>$B_3$</td>
<td>$D_4$</td>
<td>$B_5 + F'_6(D_6)$</td>
<td>$F_7(B_3) = 0$</td>
</tr>
<tr>
<td>8</td>
<td>$B_3 + F'_6(D_6)$</td>
<td>$D_6$</td>
<td>$B_3$</td>
<td>$D_8$</td>
<td>$R K_8 = R K'_6$</td>
</tr>
<tr>
<td>9</td>
<td>$D_8$</td>
<td>$B_5$</td>
<td>$D_6$</td>
<td>$B_3 + F'_8(D_8)$</td>
<td>$R K_9 = R K'_5$</td>
</tr>
<tr>
<td>10</td>
<td>$B_5 + F'_8(D_8)$</td>
<td>$D_8$</td>
<td>$B_5$</td>
<td>$D_{10}$</td>
<td>$R K_{10} = R K'_8$</td>
</tr>
<tr>
<td>11</td>
<td>$D_{10}$</td>
<td>$B_3$</td>
<td>$D_8$</td>
<td>$B_5 + F'<em>{10}(D</em>{10})$</td>
<td>$R K_{11} = R K'_7$</td>
</tr>
</tbody>
</table>

- Only two conditions on the state
- Many conditions on the key
Weak salt for Round-1 SHAvite-3_{512} (Peyrin)

- **RK₀**
  - AES (salt)
- **RK₁**
  - LFSR
- **RK₂**
  - LFSR
- **RK₃**
  - AES (salt)
- **RK₄**
  - LFSR
- **RK₅**
  - LFSR

- **cnt**
  - ⊕

- Take the zero counter;
- Take the salt that sends zero to zero;
- Use the zero message: all the subkeys are zero.
Weak salt for Round-1 SHAvite-3_{512} (Peyrin)

- Take the zero counter;
- Take the salt that sends zero to zero;
- Use the zero message: all the subkeys are zero.
Weak salt for Round-1 SHA\textit{v}ite-3\textsubscript{512} (Peyrin)

Take the zero counter;
Take the salt that sends zero to zero;
Use the zero message: all the subkeys are zero.
Weak salt for Round-1 SHAvite-3\textsubscript{512} (Peyrin)

Take the zero counter;
Take the salt that sends zero to zero;
Use the zero message: all the subkeys are zero.
Weak salt for Round-2 SHA\textit{vite}-3$_{512}$

- $R_{K_0}$
- AES (salt)
- $R_{K'_0}$
- LFSR
- $R_{K_1}$
- LFSR
- $R_{K'_1}$
- AES (salt)
- $R_{K_2}$
- LFSR
- $R_{K'_2}$
- AES (salt)
- $R_{K_3}$
- LFSR
- $R_{K'_3}$
- AES (salt)
- $R_{K_4}$
- LFSR
- $R_{K'_4}$
- AES (salt)
- $R_{K_5}$
- LFSR
- $R_{K'_5}$

- Cancel one counter in the middle;
- Take the salt that sends zero to zero;
- Use the zero subkey in the middle.

G. Leurent (ENS)  A Look at the SHA-3 Competition: Design and Analysis of Hash Functions 65 / 68
Weak salt for Round-2 SHAvite-3\textsubscript{512}

\begin{itemize}
  \item Cancel one counter in the middle;
  \item Take the salt that sends zero to zero;
  \item Use the zero subkey in the middle.
\end{itemize}
Weak salt for Round-2 SHAvite-3<sub>512</sub>

- Cancel one counter in the middle;
- Take the salt that sends zero to zero;
- Use the zero subkey in the middle.
Weak salt for Round-2 SHAvite-3_{512}

\[ \begin{align*}
\text{cnt} \oplus f00 & \quad \rightarrow \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
& \quad \quad \quad \oplus \\
\end{align*} \]

- Cancel one counter in the middle;
- Take the salt that sends zero to zero;
- Use the zero subkey in the middle.
### Weak salt for Round-2 SHAvite-3512

<table>
<thead>
<tr>
<th>$i$</th>
<th>$k^0_{0,i}$</th>
<th>$k^1_{0,i}$</th>
<th>$k^2_{0,i}$</th>
<th>$k^3_{0,i}$</th>
<th>$k^0_{1,i}$</th>
<th>$k^1_{1,i}$</th>
<th>$k^2_{1,i}$</th>
<th>$k^3_{1,i}$</th>
<th>$r$</th>
<th>$M$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>?</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0**</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>?**</td>
<td>13</td>
</tr>
</tbody>
</table>
14-round attack

**Input:** Target value H  
**Output:** message, chaining value, salt, counter

1. repeat
2. Take a random weak salt, and the corresponding message
3. Compute $2^{128}$ states with 128 chosen output bits
4. until a full preimage is found ($2^{256}$ iterations)

- Pseudo-preimage attack: complexity $2^{384}$ and $2^{128}$ memory
- Pseudo-preimage attack: complexity $2^{448}$ without memory
- Pseudo-collision attack: complexity $2^{192}$ and $2^{128}$ memory.
14-round attack

Input: Target value H
Output: message, chaining value, salt, counter

1: repeat
2: Take a random weak salt, and the corresponding message
3: Compute $2^{128}$ states with 128 chosen output bits
4: until a full preimage is found ($2^{256}$ iterations)

- Pseudo-preimage attack: complexity $2^{384}$ and $2^{128}$ memory
- Pseudo-preimage attack: complexity $2^{448}$ without memory
- Pseudo-collision attack: complexity $2^{192}$ and $2^{128}$ memory.
Questions?

Thank you for your attention!