Cryptanalysis of `Streebog`, the new Russian hash function standard

Jérémy Jean[1]

joint work with:

Jian Guo[1] Gaëtan Leurent[2] Thomas Peyrin[1] Lei Wang[1]

[1]Nanyang Technological University, Singapore

[2]INRIA, France

ASK 2014 – December 21, 2014

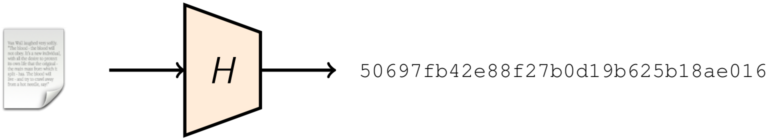
Outline

Outline

Cryptographic hash functions

A hash function is a function H in the mathematical sense:

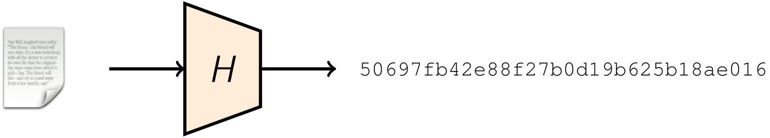
$$H : \{0,1\}^* \longrightarrow \{0,1\}^n.$$



A cryptographic hash function is a hash function securely reducing an input of arbitrary length to an output of fixed length.

Security notions for cryptographic hash functions

$$H : \{0,1\}^* \longrightarrow \{0,1\}^n.$$

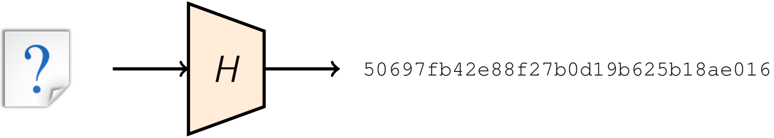


Expected behavior

- Public random oracle.
- Behave as a random function.
- Public function with no structural property.

Security notions for cryptographic hash functions

$$H : \{0,1\}^* \longrightarrow \{0,1\}^n.$$

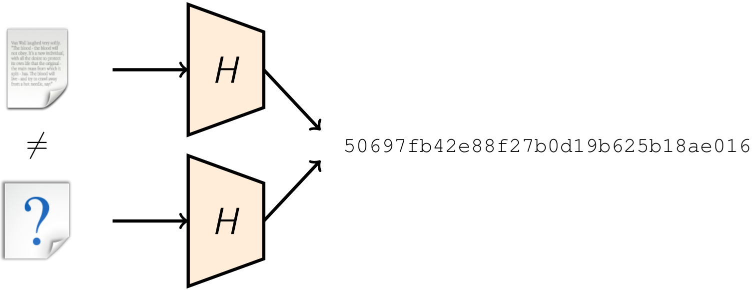


Security notions

- Preimage resistance 2^n
- Second-preimage resistance 2^n
- Collision resistance $2^{n/2}$

Security notions for cryptographic hash functions

$$H : \{0,1\}^* \longrightarrow \{0,1\}^n.$$

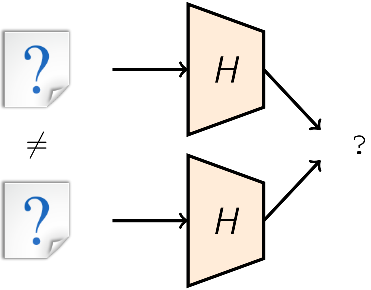


Security notions

▶ Preimage resistance	2^n
▶ Second-preimage resistance	2^n
▶ Collision resistance	$2^{n/2}$

Security notions for cryptographic hash functions

$$H : \{0,1\}^* \longrightarrow \{0,1\}^n.$$

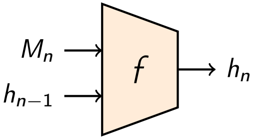


Security notions

- Preimage resistance 2^n
- Second-preimage resistance 2^n
- Collision resistance $2^{n/2}$

Iterated functions

- H takes input M of any length $\Rightarrow$ Difficult to handle.
- Use fixed-size input function called compression function
  $f : \{0,1\}^n \times \{0,1\}^b \longrightarrow \{0,1\}^n$.
- Mode of operation to handle messages longer (or shorter) than b bits.
- Chain the successive outputs h_i of f.
- First chaining value h_0 initialized to some IV (*initialization vector*).
- Split M into b-bit chunks $M = M_1||M_2||\cdots$.



Compression function $f : \{0,1\}^n \times \{0,1\}^b \longrightarrow \{0,1\}^n$.

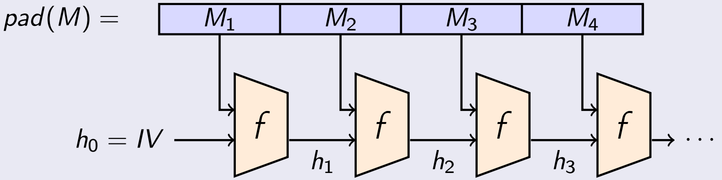
Merkle-Damgård mode: construction

- Construction of a hash function $H^f : \{0,1\}^* \longrightarrow \{0,1\}^n$.
- Domain extension (mode) from a given compression function
  $f : \{0,1\}^n \times \{0,1\}^b \longrightarrow \{0,1\}^n$.
- Initial message M is padded by `1`, as few `0` as possible and the length $|M|$ to reach $|pad(M)| = m \cdot b$ being a multiple of b. That is:

$$pad(M) = M \;||\; 1 \;||\; 0^* \;||\; |M|.$$

- Merkle-Damgård strengthening: $pad(M)$ ends with $|M|$.
- Set $h_0 = IV$, and for $k \in \{0, \ldots, m-1\}$, do $h_{k+1} = f(h_k, M_k)$.
- Output the last chaining value h_m as $H^f(M)$.

Domain extension by Merkle-Damgård



Merkle-Damgård mode: security

- **Construction to build a collision-resistant hash function from a collision-resistant compression function.**

Merkle-Damgård mode: security

- **Construction to build a collision-resistant hash function from a collision-resistant compression function.**
- Assume f is collision-resistant.
- Suppose we have a collision in the hash function: $H^f(M) = H^f(M')$.

Merkle-Damgård mode: security

- **Construction to build a collision-resistant hash function from a collision-resistant compression function.**
- Assume f is collision-resistant.
- Suppose we have a collision in the hash function: $H^f(M) = H^f(M')$.
- We can show that we can also find a collision in f.
  - If $|M| \neq |M'|$, the last block containing the length in the padding is different. So: $f(h_{|M|-1}, M_{|M|}) = f(h'_{|M'|-1}, M'_{|M'|})$.
  - If $|M| = |M'|$, we search for collision backwards in the chains until we find the collision.

Merkle-Damgård mode: security

- **Construction to build a collision-resistant hash function from a collision-resistant compression function.**
- Assume f is collision-resistant.
- Suppose we have a collision in the hash function: $H^f(M) = H^f(M')$.
- We can show that we can also find a collision in f.
  - If $|M| \neq |M'|$, the last block containing the length in the padding is different. So: $f(h_{|M|-1}, M_{|M|}) = f(h'_{|M'|-1}, M'_{|M'|})$.
  - If $|M| = |M'|$, we search for collision backwards in the chains until we find the collision.
- Therefore: **if** f is collision-resistant, **then** H^f is collision-resistant.

Multi-collision attack

- Technique by Antoine Joux (2004).
- What is the cost of constructing $\{M_1, \ldots, M_r\}$ such that

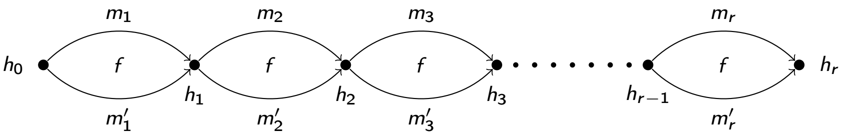
$$\forall i, j, \quad H^f(M_i) = H^f(M_j)?$$

Multi-collision attack

- Technique by Antoine Joux (2004).
- What is the cost of constructing $\{M_1, \ldots, M_r\}$ such that

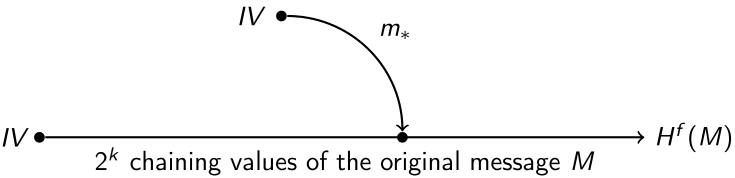
$$\forall i, j, \quad H^f(M_i) = H^f(M_j)?$$

- Random function: about $(r!)^{1/r} \cdot 2^{n(r-1)/r}$ function evaluations.
- MD hash function H^f: about $r \cdot 2^{n/2}$ function evaluations.
- Idea:
  - Rely on the iterated structure of Merkle-Damgård.
  - Construct internal f-collisions on the chaining variables.
  - Reach 2^r colliding messages from only r internal collisions.



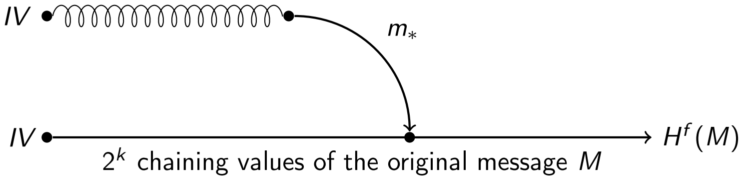
Generic second-preimage attack

- For very long message of 2^k blocks, one can construct a second preimage of $H^f(M)$ in 2^{n-k} computations ($k \leq n/2$).
- Indeed, it is sufficient to hit one of the intermediate chaining values in the MD chain to use the end of the original message to reach $H^f(M)$: $IV \rightarrow h_1 \rightarrow \cdots \rightarrow H^f(M)$.
- MD strengthening could prevent this



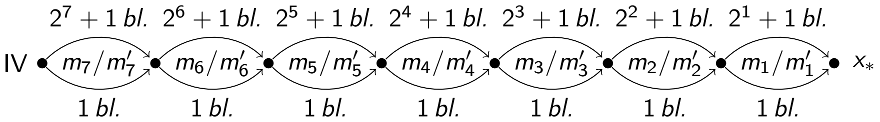
Generic second-preimage attack

- For very long message of 2^k blocks, one can construct a second preimage of $H^f(M)$ in 2^{n-k} computations ($k \leq n/2$).
- Indeed, it is sufficient to hit one of the intermediate chaining values in the MD chain to use the end of the original message to reach $H^f(M)$: $IV \to h_1 \to \cdots \to H^f(M)$.
- MD strengthening could prevent this, but Kelsey and Schneier have shown (2005) that we can actually construct expandable messages to *arbitrarily* choose the length of the prefix of the second message.



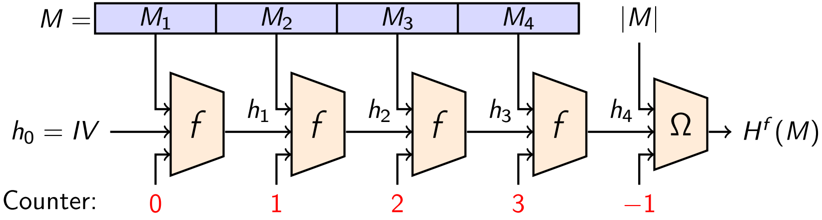
Expandable message

- Expandable messages due to [KS05]
- Multicollision with different lengths:
  - t pairs with lengths $(1, 2^k + 1)$, $0 \leq k < t$.
  - Set of 2^t messages with length in $[t, 2^t + t - 1]$.
  - All reach the same final chaining value x_*.
- Construction of a message m of length $t + L$ using the binary representation of L, that link IV to x_*.
- Second-preimage attack on MD:
  - Link x_* to original message using random blocks.
  - This gives the length to use in the expandable message.
  - HAIFA prevents using an expandable message with the counter input.



HAIFA framework

- To prevent this attack, we can introduce a counter in the compression function f inputs.
- Used in the **HAIFA framework** due to Eli Biham and Orr Dunkelman.
- The i-th call to f in H^f uses the value i.
- Output transformation Ω (prevent length-extension attack).
- The second-preimage attack for long messages does not work anymore.
- Provable 2^n security bound for second preimages when f is ideal (Bouillaguet, Fouque and Zimmer, 2010).



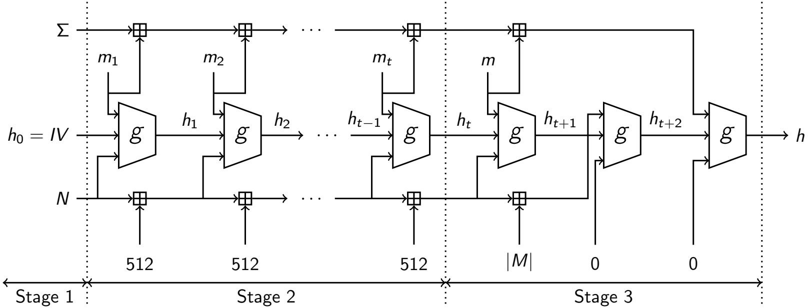
Outline

Streebog: new Russian hash function.

- New hash function standard in Russia.
- Standardized name: GOST R 34.11-2012
- Nickname of that function: Streebog.
- Previous standard: GOST R 34.11-94.
  - Theoretical weaknesses.
  - Rely on the GOST block cipher from the same standard.
  - This block cipher has also been weakened by third-party cryptanalysis.

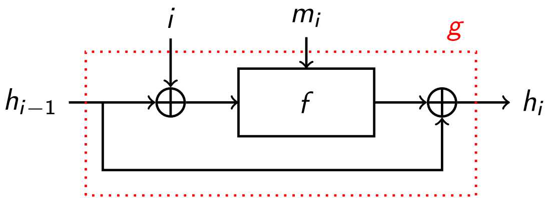
Specifications: domain extension.

- Two versions: Streebog-256 and Streebog-512.
- 10^* padding: $m_1||\cdots||m_t||m$ (blocks of 512 bits).
- Compression function: g.
- Checksum: Σ, over the message blocks m_i (addition modulo 2^{512}).
- Counter: N, HAIFA input to g over the number of processed bits.
- Three stages: initialization, message processing and finalization.



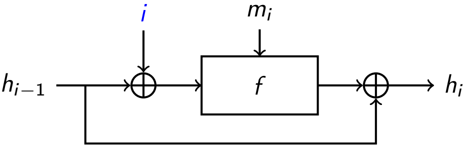
Specifications: compression function.

- Simplification: the counter counts #blocks, not #bits.
- g compresses (h_{i-1}, i, m_i) to h_i using: $h_i = f(h_{i-1} \oplus i, m_i) \oplus h_{i-1}$.
- Our attack is independent of the specifications of f (deterministic).



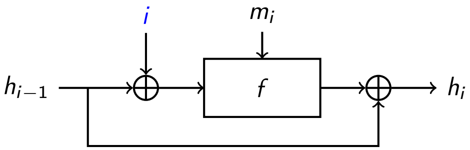
- g is one instantiation of a HAIFA compression function.
- The counter is simply XORed to the input of the f function.

Equivalent compression function.

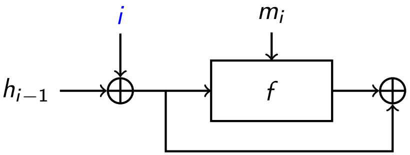


$$h_i = h_{i-1} \oplus f(h_{i-1} \oplus i, m_i) \quad \Longleftrightarrow$$

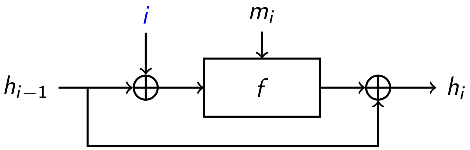
Equivalent compression function.



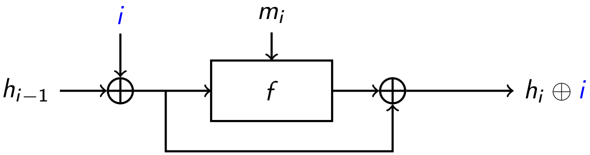
$$h_i = h_{i-1} \oplus f(h_{i-1} \oplus i, m_i) \quad \Longleftrightarrow$$



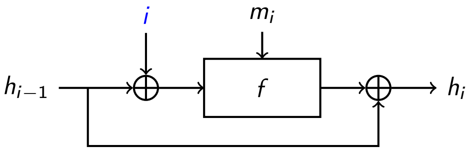
Equivalent compression function.



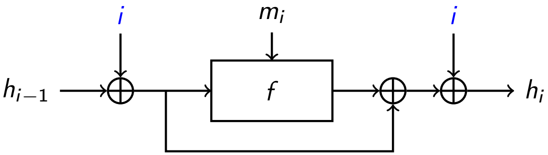
$$h_i = h_{i-1} \oplus f(h_{i-1} \oplus i, m_i) \quad \Longleftrightarrow$$



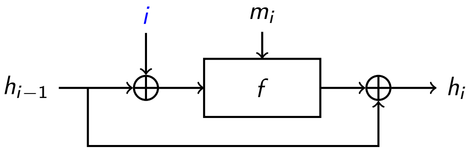
Equivalent compression function.



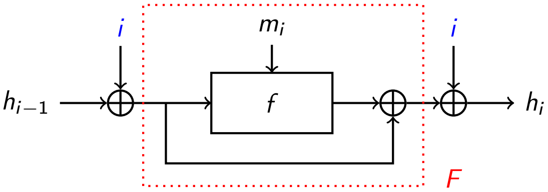
$$h_i = h_{i-1} \oplus f(h_{i-1} \oplus i, m_i) \quad \Longleftrightarrow$$



Equivalent compression function.



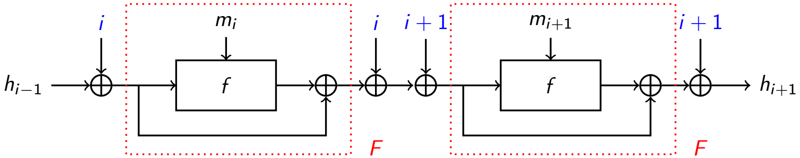
$$h_i = h_{i-1} \oplus f(h_{i-1} \oplus i, m_i) \iff \begin{cases} h_i = F(h_{i-1} \oplus i, m_i) \oplus i, \\ F(x, m_i) = f(x, m_i) \oplus x. \end{cases}$$



The function F is independent of the counter value!

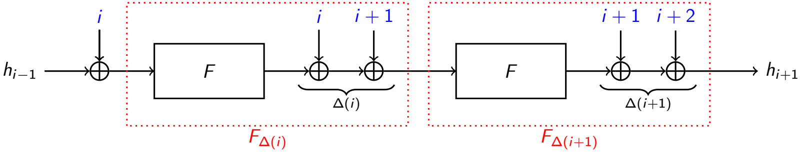
Iteration of the equivalent compression function.

- We have an equivalent representation of the compression function.
- Its iteration allows to combine the counter additions.



$$\Delta(i) \stackrel{\text{def}}{=} i \oplus (i+1),$$

$$F_{\Delta(i)}(X,Y) \stackrel{\text{def}}{=} F(X,Y) \oplus \Delta(i).$$



Relations between functions $F_{\Delta(i)}$ for $1 \leq i \leq t$ (1/2).

Recall that t is the number of full blocks $m_1 || \cdots || m_t || m$, $|m| < 512$.
We observe that:

- For all even i, $\Delta(i) = i \oplus (i+1) = 1$.
  $\Longrightarrow$ The same function F_1 is used every other time.
- Sequence of $\Delta(i)$ is very structured.

i:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
$\Delta(i)$:	1	3	1	7	1	3	1	15	1	3	1	7	1	3	1	31	1	3	1	7	1	3	1	15

Let $s > 0$, and denoting $\langle \mathtt{i} \rangle$ the s-bit binary representation of $i < 2^s - 1$:

$$\Delta(i + 2^s) = \Big(1 || \langle \mathtt{i} \rangle\Big) \oplus \Big(1 || \langle \mathtt{i} + 1 \rangle\Big) = \langle \mathtt{i} \rangle \oplus \langle \mathtt{i} + 1 \rangle = \Delta(i).$$

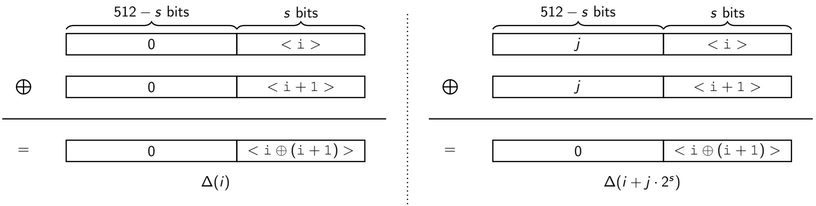
More generally: $F_{\Delta(i)} = F_{\Delta(i + j \cdot 2^s)}$ for all $0 \leq i \leq 2^s - 1$ and $j \geq 0$.

For example, with $s = 2$, F_1 and $F_{1+2^2} = F_5$ are equal.

Relations between functions $F_{\Delta(i)}$ for $1 \leq i \leq t$ (2/2).

Given an integer $s > 0$, we have:

$$\forall i \in \{0, \dots, 2^s - 2\}, \quad \forall j > 0 : \qquad F_{\Delta(i)} = F_{\Delta(j \cdot 2^s + i)}$$

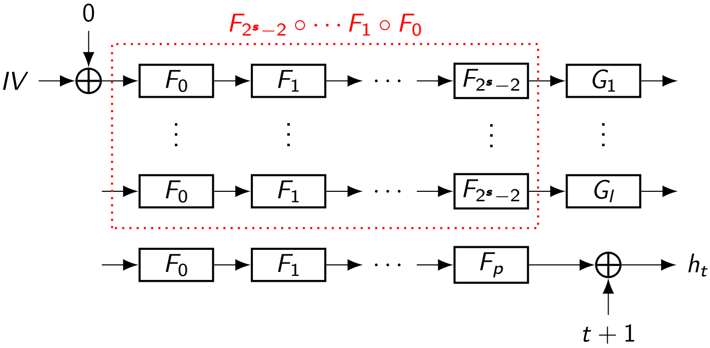


Consequently:

- The same sequence of $2^s - 1$ functions are used in the domain extension algorithm.
- This seems weaker than a true HAIFA mode.

Equivalent description of stage 2 of the domain extension.

- The last function differs in each 2^s-chunk.
  $\Longrightarrow$ We call it $G_j = F_{\Delta(j \times 2^s - 1)}$.
- We define l as the number of $(2^s - 1)$-chains of F functions: $l = \left\lfloor \frac{t}{2^s} \right\rfloor$.
  Moreover, let p be the remainder of t modulo 2^s.
- That is: the function $F_{2^s-2} \circ \cdots F_1 \circ F_0$ is reused l times.



Cryptographic consequences of the HAIFA instantiation.

Streebog is **one** choice of counter usage from the HAIFA framework.

Consequences of this choice:

- Counters at steps i and $i+1$ can be combined.
- Distinction of compression function calls in the HAIFA framework not achieved.
- Domain extension similar to a Merkle-Damgård scheme.
  $\Longrightarrow$ Possibility to apply existing known second-preimage attacks.

Cryptographic consequences of the HAIFA instantiation.

Streebog is **one** choice of counter usage from the HAIFA framework.

Consequences of this choice:

- Counters at steps i and $i+1$ can be combined.
- Distinction of compression function calls in the HAIFA framework not achieved.
- Domain extension similar to a Merkle-Damgård scheme.
  $\Longrightarrow$ Possibility to apply existing known second-preimage attacks.

Our second-preimage attacks on Streebog (security level: 2^{512}):

- Using a diamond structure:
  - Original message of at least 2^{179} blocks.
  - 2^{342} compression function evaluations.
- Using a expandable message:
  - Original message of at least 2^{259} blocks.
  - 2^{266} compression function evaluations.

Outline

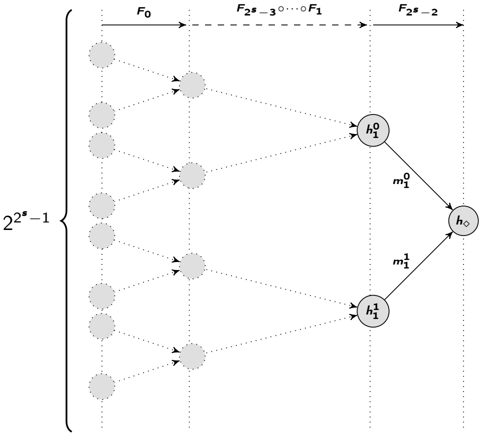
Diamond structure (1/2)

Diamond structure:

- Introduced in [KK06].
- Complete binary tree.
- Nodes: chaining values.
- Edges: 1-block n-bit messages.
- Depth d.

Construction:

- Levels constructed sequentially.
- Complexity: $2^{(n+d)/2}$ calls.
- Evaluation done in [KK13].



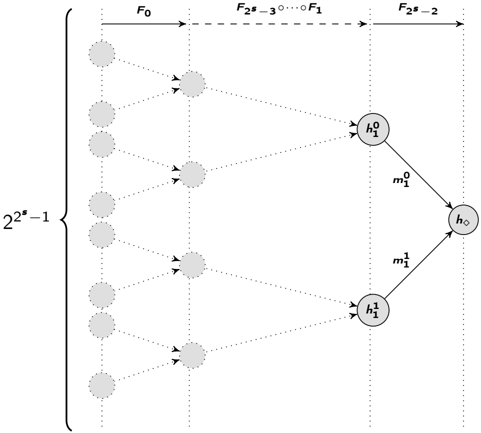
Diamond structure (2/2)

Diamond used in our attack:

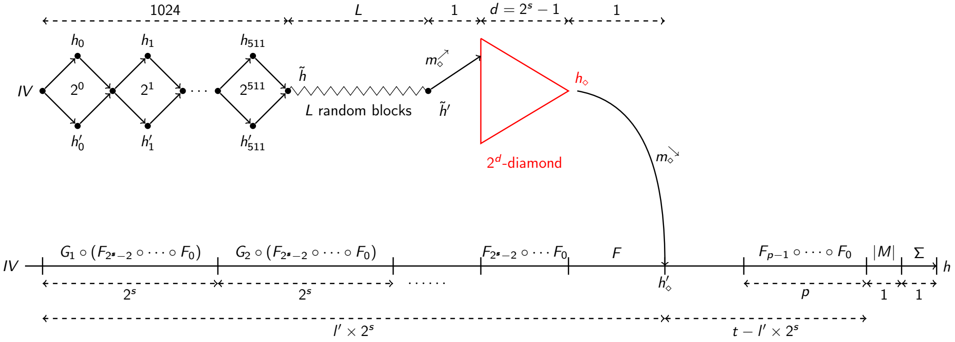
- Root $h_\diamond$.
- Depth $d = 2^s - 1$.
- F_i's used to join the levels.
- #leaves=2^{2^s-1}.

Remarks:

- Same function at each level in the original attack on Merkle-Damgård.
- Here, full control of the counter effect in the $(2^s - 1)$-chains with different functions F_i.

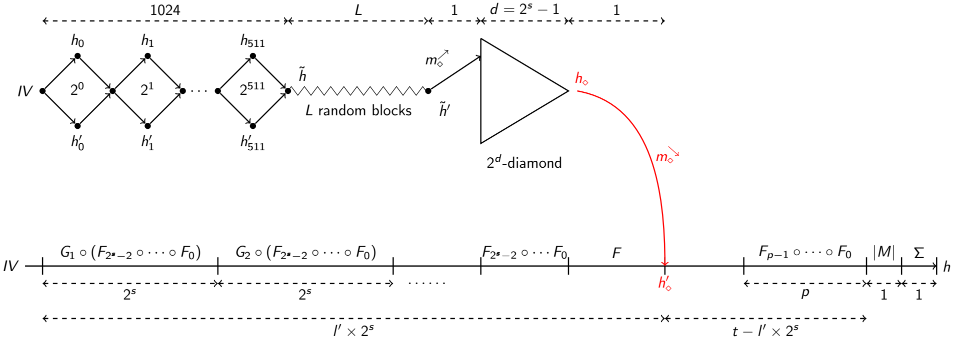


Overview of the diamond attack.



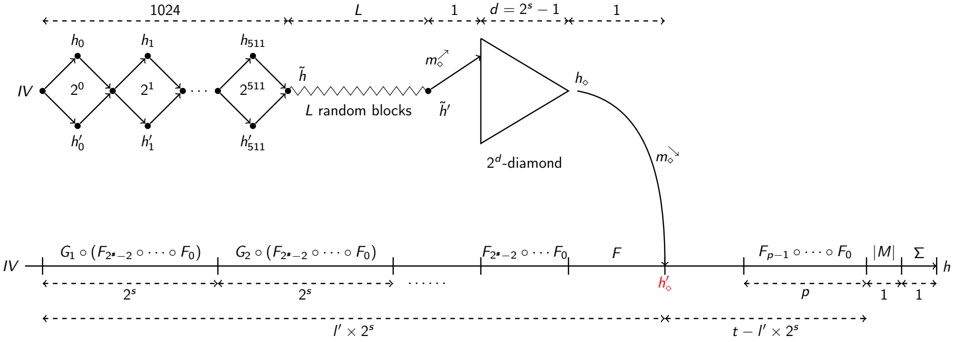
1. Construction of the diamond.
2. Randomize $m_\diamond^{\searrow}$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^{\nearrow}$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



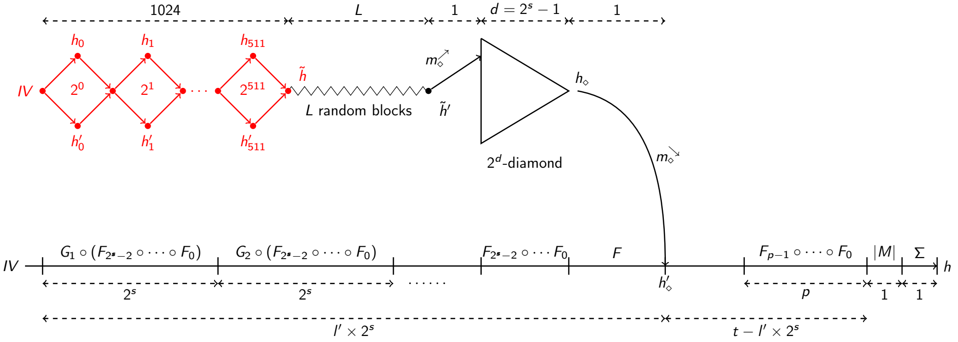
1. Construction of the diamond.
2. Randomize $m_\diamond^{\searrow}$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^{\nearrow}$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



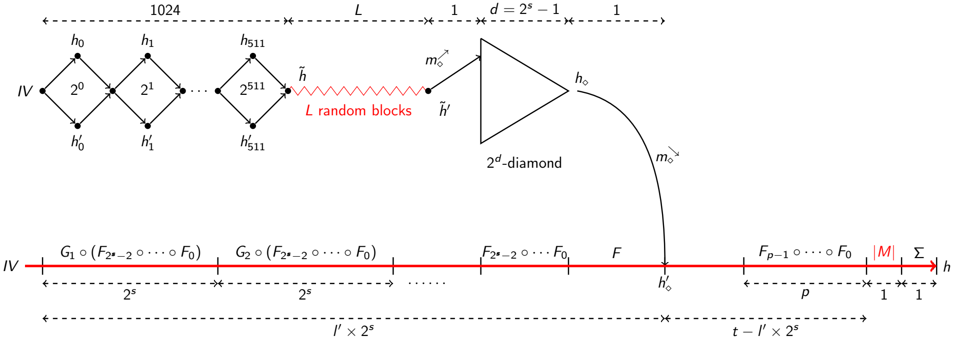
1. Construction of the diamond.
2. Randomize $m_\diamond^{\searrow}$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^{\nearrow}$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



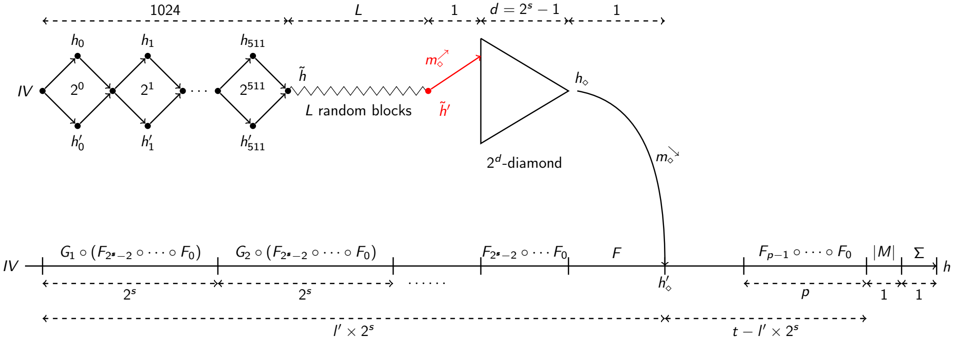
1. Construction of the diamond.
2. Randomize $m_\diamond^{\searrow}$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^{\nearrow}$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



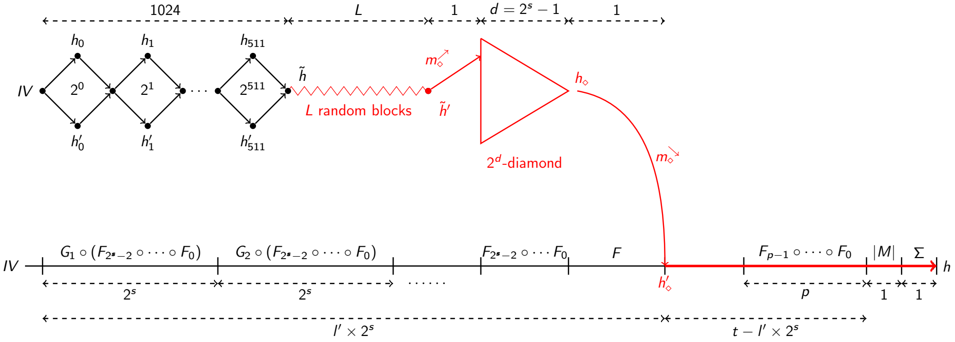
1. Construction of the diamond.
2. Randomize $m_\diamond^\searrow$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^\nearrow$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



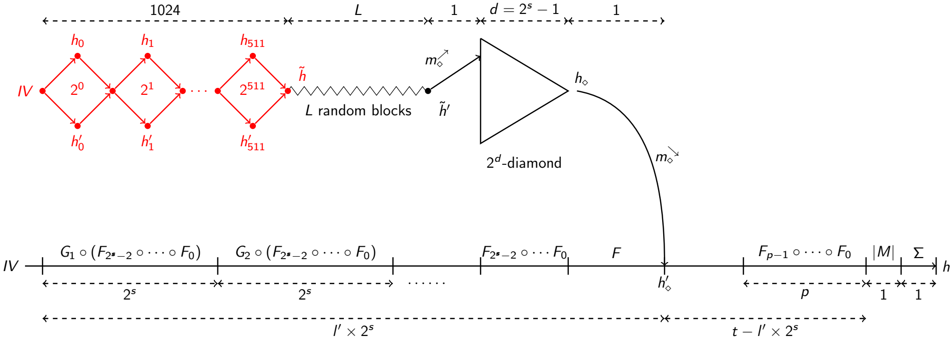
1. Construction of the diamond.
2. Randomize $m_\diamond^\searrow$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^\nearrow$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



1. Construction of the diamond.
2. Randomize $m_\diamond^{\searrow}$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^{\nearrow}$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Overview of the diamond attack.



1. Construction of the diamond.
2. Randomize $m_\diamond^\searrow$ to hit $h'_\diamond$.
3. Deduce the counter value N.
4. Construct 2^{512}-multicollision.
5. Randomize L blocks to match $|M|$.
6. Pick about 2^{n-d} $m_\diamond^\nearrow$ to hit the diamond.
7. Evaluate reduced checksum σ.
8. Use multicollision to match $\Sigma - \sigma$.

Complexity analysis of the diamond attack.

Time complexity T

$$T = 2^{(n+d)/2} + 512 \times 2^{n/2} + 2^{n-\log_2(l)} + 2^{n-d},$$

with:

- Construction of the diamond.
- Joux's multicollision using 512 two-block messages.
- Connect the root of the diamond to the original message.
- Connect the multicollision to one leaf of the diamond.

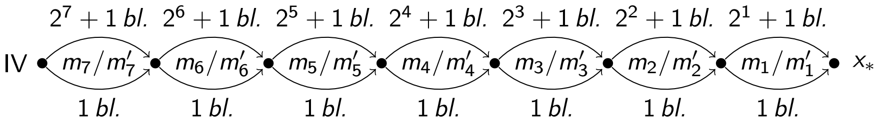
Minimize with:

- $d = n/3 = 2^s - 1$ the depth of the diamond, i.e. $s = \lceil \log_2(n/3) \rceil$.
- as long as $l = \left\lfloor \frac{t}{2^s} \right\rfloor$ is $l \geq 2^{n/3}$, i.e. $t \geq \lceil 2^{n/3+\log_2(n/3)} \rceil$.
- For `Streebog-512`: $T = 2^{342}$ for $|M| \geq 2^{179}$.

Outline

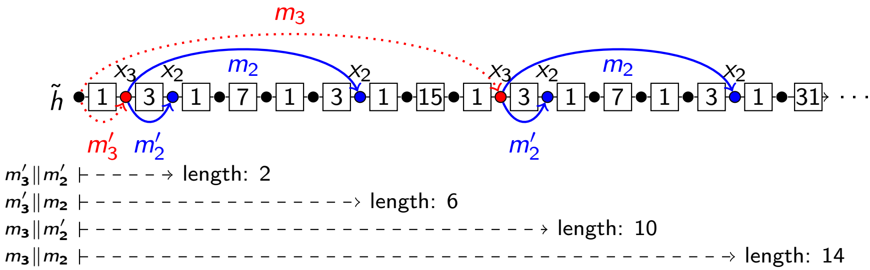
Recall: Expandable message

- Expandable messages due to [KS05]
- Multicollision with different lengths:
  - t pairs with lengths $(1, 2^k + 1)$, $0 \leq k < t$.
  - Set of 2^t messages with length in $[t, 2^t + t - 1]$.
  - All reach the same final chaining value x_*.
- Construction of a message m of length $t + L$ using the binary representation of L, that link IV to x_* (Figure: $t = 7$).
- Second-preimage attack on MD:
  - Link x_* to original message using random blocks.
  - This gives the length to use in the expandable message.
  - HAIFA prevents using an expandable message with the counter input.

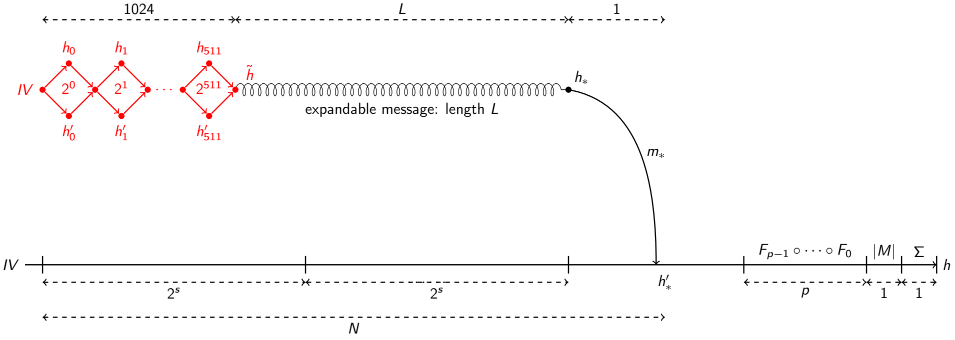


Expandable messages in Streebog

- Here, the counter input is weak.
- We can still apply the expandable message technique:
  - The functions $F_{\Delta(i)}$ are independent of the counter,
  - but the inner calls are not the same (HAIFA, not MD).
- Small example: 4 messages from $\tilde{h}$ to x_2.
  - Find (m_3', m_3) of lengths $(1, 2^3+1)$ colliding on x_3.
  - Find (m_2', m_2) of lengths $(1, 2^2+1)$ colliding on x_2.
  - The 4-message structure has lengths in $\{2, 6, 10, 14\}$.

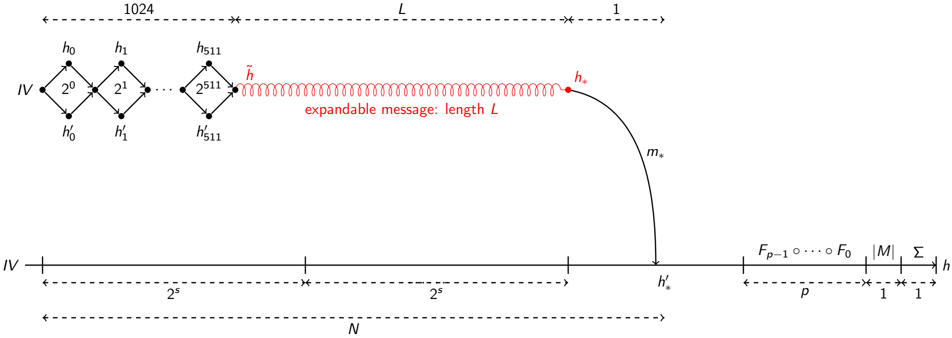


Overview of the attack using an expandable message.



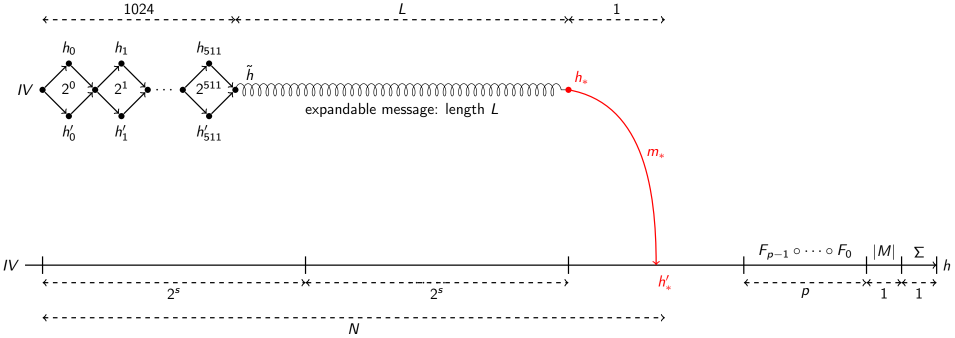
1. Construct the 2^{512}-multicollision.
2. Construct the expandable message.
3. Randomize m_* to hit h'_*.
4. Deduce the counter value.
5. Choose the valid length L and solve the checksum.

Overview of the attack using an expandable message.



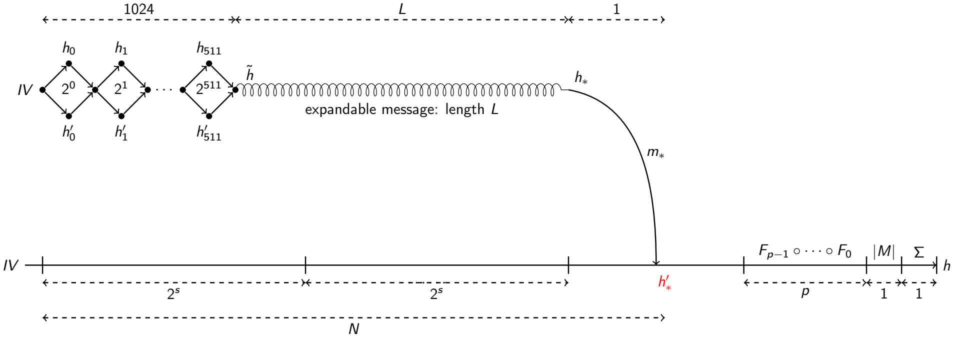
1. Construct the 2^{512}-multicollision.
2. Construct the expandable message.
3. Randomize m_* to hit h'_*.
4. Deduce the counter value.
5. Choose the valid length L and solve the checksum.

Overview of the attack using an expandable message.



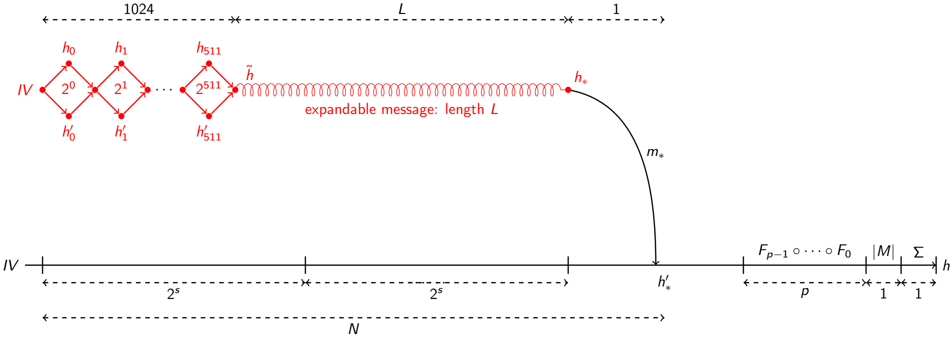
1. Construct the 2^{512}-multicollision.
2. Construct the expandable message.
3. Randomize m_* to hit h'_*.
4. Deduce the counter value.
5. Choose the valid length L and solve the checksum.

Overview of the attack using an expandable message.



1. Construct the 2^{512}-multicollision.
2. Construct the expandable message.
3. Randomize m_* to hit h'_*.
4. Deduce the counter value.
5. Choose the valid length L and solve the checksum.

Overview of the attack using an expandable message.



1. Construct the 2^{512}-multicollision.
2. Construct the expandable message.
3. Randomize m_* to hit h'_*.
4. Deduce the counter value.
5. Choose the valid length L and solve the checksum.

Complexity analysis.

Time complexity T

$$T = 512 \times 2^{n/2} + 256 \times 2^{n/2} + 2^{n-l},$$

with:

- Joux's multicollision using 512 two-block messages.
- Construction of the expandable message.
- Connect the expandable message to the challenge ($l = \lfloor \frac{t}{2^s} \rfloor$).

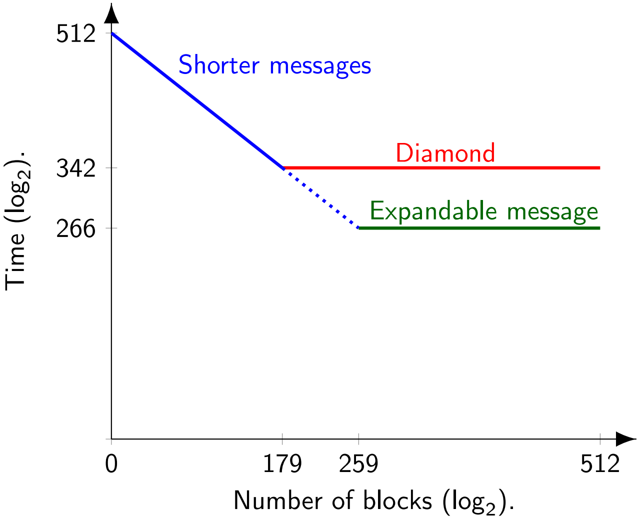
Minimize with:

- $l > 2^{n/2}/n$, i.e. more than 2^{259} blocks in the original message.
- T about $n \cdot 2^{n/2}$, i.e. 2^{266} CF evaluations ($s = 11$).

Outline

1. Introduction
2. Streebog
3. Diamond attack
4. Expandable message attack
5. Conclusion

Comparison of the two attacks on Streebog



Conclusion

- We study Streebog, the Russian hashing standard.
- The hash function instantiates the HAIFA framework.
- This works answer a public call from the Russian government.
  ⇒ http://www.streebog.info/

Conclusion

- We study Streebog, the Russian hashing standard.
- The hash function instantiates the HAIFA framework.
- This works answer a public call from the Russian government.
  ⇒ http://www.streebog.info/
- We propose an equivalent representation that hijack the counter effect of Streebog-512.
- Consequently, one can reuse previous second-preimage attack strategies:
  - using a diamond structure,
  - using an expandable message.
- The two attacks have time complexity T for message length $> L$:
  - $T = 2^{342}$ and $L = 2^{179}$,
  - $T = 2^{266}$ and $L = 2^{259}$.

Conclusion

- We study Streebog, the Russian hashing standard.
- The hash function instantiates the HAIFA framework.
- This works answer a public call from the Russian government.
  ⇒ http://www.streebog.info/
- We propose an equivalent representation that hijack the counter effect of Streebog-512.
- Consequently, one can reuse previous second-preimage attack strategies:
  - using a diamond structure,
  - using an expandable message.
- The two attacks have time complexity T for message length $> L$:
  - $T = 2^{342}$ and $L = 2^{179}$,
  - $T = 2^{266}$ and $L = 2^{259}$.

Thank you!