StreamHash2 Hash Function

Michał Trojnara

Institute of Telecommunications Faculty of Electronics and Information Technology Warsaw University of Technology

26 May 2010



イロト イポト イヨト イヨト

Outline



Origins of StreamHash Family

- History
- Prior Cryptanalysis
- Hash Functions
 - Requirements
 - Traditional Design
- 3 StreamHash2
 - StreamHash2 Design
 - Properties





Hash Functions StreamHash2 Conclusion History Prior Cryptanalysis

Next Section





프 🖌 🖌 프

History Prior Cryptanalysis

History of StreamHash Family

Jan 2007 NIST published draft of requirements for the SHA-3 competition

Nov 2007 NIST requested submissions for new hash functions

Oct 2008 StreamHash function submitted for the SHA-3 competition

Dec 2008 StreamHash function published by NIST

Dec 2008 Published attacks against StreamHash function

2009-2010 Working on the successor - StreamHash2



ヘロト ヘアト ヘビト ヘビ

History Prior Cryptanalysis

History of StreamHash Family

Jan 2007 NIST published draft of requirements for the SHA-3 competition

Nov 2007 NIST requested submissions for new hash functions

Oct 2008 StreamHash function submitted for the SHA-3 competition

Dec 2008 StreamHash function published by NIST

Dec 2008 Published attacks against StreamHash function

2009-2010 Working on the successor – StreamHash2



ヘロト ヘアト ヘビト ヘビ

History Prior Cryptanalysis

History of StreamHash Family

Jan 2007 NIST published draft of requirements for the SHA-3 competition

Nov 2007 NIST requested submissions for new hash functions

Oct 2008 StreamHash function submitted for the SHA-3 competition

Dec 2008 StreamHash function published by NIST Dec 2008 Published attacks against StreamHash function 2009-2010 Working on the successor – StreamHash2



イロト イポト イヨト イヨ

History Prior Cryptanalysis

History of StreamHash Family

- Jan 2007 NIST published draft of requirements for the SHA-3 competition
- Nov 2007 NIST requested submissions for new hash functions
- Oct 2008 StreamHash function submitted for the SHA-3 competition
- Dec 2008 StreamHash function published by NIST

Dec 2008 Published attacks against StreamHash function 2009-2010 Working on the successor – StreamHash2



・ 同 ト ・ 三 ト ・

History Prior Cryptanalysis

History of StreamHash Family

- Jan 2007 NIST published draft of requirements for the SHA-3 competition
- Nov 2007 NIST requested submissions for new hash functions
- Oct 2008 StreamHash function submitted for the SHA-3 competition
- Dec 2008 StreamHash function published by NIST
- Dec 2008 Published attacks against StreamHash function

2009-2010 Working on the successor – StreamHash2



・ 戸 ・ ・ 三 ・ ・

History Prior Cryptanalysis

History of StreamHash Family

- Jan 2007 NIST published draft of requirements for the SHA-3 competition
- Nov 2007 NIST requested submissions for new hash functions
- Oct 2008 StreamHash function submitted for the SHA-3 competition
- Dec 2008 StreamHash function published by NIST
- Dec 2008 Published attacks against StreamHash function
- 2009-2010 Working on the successor StreamHash2



Hash Functions StreamHash2 Conclusion History Prior Cryptanalysis

Next Section



Origins of StreamHash Family

- History
- Prior Cryptanalysis
- 2 Hash Functions
 - Requirements
 - Traditional Design
- 3 StreamHash2
 - StreamHash2 Design
 - Properties

Conclusion



э

Hash Functions StreamHash2 Conclusion History Prior Cryptanalysis

Preimage Attack

Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg

- Multicollision Attack (Antoine Joux: Multicollisions in Iterated Hash Functions, CRYPTO 2004)
 - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
 - Complexity of $\frac{\overline{n}}{2} \cdot 2^{n/2}$ for finding preimages
- Issue addressed in StreamHash2 by introducing a counter

ヘロト ヘアト ヘヨト ヘ

Origins of StreamHash Family Hash Functions

StreamHash2

Conclusion

History Prior Cryptanalysis

Preimage Attack

- Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg
- Multicollision Attack (Antoine Joux: Multicollisions in Iterated Hash Functions, CRYPTO 2004)
 - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
 - Complexity of $\frac{\overline{n}}{2} \cdot 2^{n/2}$ for finding preimages

Issue addressed in StreamHash2 by introducing a counter

Origins of StreamHash Family Hash Functions

StreamHash2

Conclusion

History Prior Cryptanalysis

Preimage Attack

- Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg
- Multicollision Attack (Antoine Joux: Multicollisions in Iterated Hash Functions, CRYPTO 2004)
 - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
 - Complexity of $\frac{\overline{n}}{2} \cdot 2^{n/2}$ for finding preimages
- Issue addressed in StreamHash2 by introducing a counter

Hash Functions StreamHash2 Conclusion History Prior Cryptanalysis

Collision Attack

Tor E. Bjørstad, Department of Informatics, University of Bergen, Norway

- Internal state cycles
- The ⊕ operation of StreamHash did not propagate changes between the four bytes of the 32-byte state word
- Issue addressed by replacing \oplus operation with \boxplus



イロト イ理ト イヨト イヨト

Hash Functions History StreamHash2 Prior Cryptanalysis Conclusion

Collision Attack

- Tor E. Bjørstad, Department of Informatics, University of Bergen, Norway
- Internal state cycles
- The ⊕ operation of StreamHash did not propagate changes between the four bytes of the 32-byte state word
- Issue addressed by replacing \oplus operation with \boxplus



Hash Functions History StreamHash2 Prior Cryptanalysis Conclusion

Collision Attack

- Tor E. Bjørstad, Department of Informatics, University of Bergen, Norway
- Internal state cycles
- The ⊕ operation of StreamHash did not propagate changes between the four bytes of the 32-byte state word
- Issue addressed by replacing \oplus operation with \boxplus

Requirements Traditional Design

Next Section





э

Michał Trojnara StreamHash2 Hash Function

Requirements Traditional Design

Functional Requirements

Hash function h(m) is expected to meet the following requirements

- Input *m* can be of any length
- Output of h(m) has a predefined, fixed length
- h(m) is fast to compute for any given m



イロト イヨト イヨト イ

Requirements Traditional Design

Functional Requirements

Hash function h(m) is expected to meet the following requirements

- Input *m* can be of any length
- Output of *h*(*m*) has a predefined, fixed length
- h(m) is fast to compute for any given m



Requirements Traditional Design

Functional Requirements

Hash function h(m) is expected to meet the following requirements

- Input *m* can be of any length
- Output of *h*(*m*) has a predefined, fixed length
- *h*(*m*) is fast to compute for any given *m*



Requirements Traditional Design

Security Requirements

• **Preimage resistance** Practically infeasible for any given *h*(*m*) to compute *m*

• Second preimage resistance Practically infeasible for any given m_1 message it is infeasible to find another m_2 such that $h(m_1) = h(m_2)$

Collision resistance

Practically infeasible to find two different messages m_1 and m_2 such that $h(m_1) = h(m_2)$

<ロト < 回 > < 回 > < 三 > < 三 >

Requirements Traditional Design

Security Requirements

Preimage resistance

Practically infeasible for any given h(m) to compute m

Second preimage resistance

Practically infeasible for any given m_1 message it is infeasible to find another m_2 such that $h(m_1) = h(m_2)$

Collision resistance

Practically infeasible to find two different messages m_1 and m_2 such that $h(m_1) = h(m_2)$

・ロト ・厚ト ・ヨト ・ヨト

Requirements Traditional Design

Security Requirements

Preimage resistance

Practically infeasible for any given h(m) to compute m

Second preimage resistance

Practically infeasible for any given m_1 message it is infeasible to find another m_2 such that $h(m_1) = h(m_2)$

Collision resistance

Practically infeasible to find two different messages m_1 and m_2 such that $h(m_1) = h(m_2)$

・ 同 ト ・ ヨ ト ・

Requirements Traditional Design

Next Section





э

э.

Requirements Traditional Design

Merkle-Damgård Construction



Michał Trojnara StreamHash2 Hash Function

э

Requirements Traditional Design

Davies-Meyer Compression Function



$H_i \leftarrow E_{m_i}(H_{i-1}) \oplus H_{i-1}$



イロト イポト イヨト イヨト

StreamHash2 Design Properties

Next Section





э

StreamHash2 Design Properties

State Vector

State vector consists of 32-bit words

- 7 × 32 = 224 bits
- 8 × 32 = 256 bits
- 12 × 32 = 384 bits
- 16 × 32 = 512 bits



イロト イポト イヨト イヨト

StreamHash2 Design Properties

NLF Transformation

NLF is a non-linear transformation based on an S-BOX



StreamHash2 Design Properties

StreamHash Family Structure



StreamHash2 Design

Conclusion

NLF Implementation of StreamHash2 Function

 $state_{i+1} \leftarrow state_i \boxplus S$ -BOX[LSB($state_i$) $\oplus b \oplus i$] $\boxplus c$

, where:

- b processed byte value
- c processed byte index
- *i* state vector index
- S-BOX S-BOX table
 - state state vector



StreamHash2 Design Properties

Next Section





э

Properties

Conclusion

Streamhash2 Advantages – Simplicity

Clear and easy to analyze design



イロト イポト イヨト イヨト

StreamHash2 Design Properties

Streamhash2 Advantages – Simplicity

- Clear and easy to analyze design
- Minimal size of code
- Minimal size of variables
- Low size of static data
- Flexible hash value length



< 🗇 🕨

StreamHash2 Design Properties

Streamhash2 Advantages - Simplicity

- Clear and easy to analyze design
- Minimal size of code
- Minimal size of variables
- Low size of static data
- Flexible hash value length



StreamHash2 Design Properties

Streamhash2 Advantages - Simplicity

- Clear and easy to analyze design
- Minimal size of code
- Minimal size of variables
- Low size of static data
- Flexible hash value length



StreamHash2 Design Properties

Streamhash2 Advantages - Simplicity

- Clear and easy to analyze design
- Minimal size of code
- Minimal size of variables
- Low size of static data
- Flexible hash value length



Properties

Conclusion

Streamhash2 Advantages – Performance

Easy to parallelize internal structure ۲



・ロト ・厚ト ・ヨト ・ヨト

Properties

Conclusion

Streamhash2 Advantages – Performance

- Easy to parallelize internal structure
- Negligible performance impact of machine endianness



・ 通 ト ・ ヨ ト ・

Properties

Conclusion

Streamhash2 Advantages – Performance

- Easy to parallelize internal structure
- Negligible performance impact of machine endianness
- High performance on 8-bit and 16-bit architectures



・ 戸 ・ ・ 三 ・ ・

Properties

Conclusion

Streamhash2 Advantages – Performance

- Easy to parallelize internal structure
- Negligible performance impact of machine endianness
- High performance on 8-bit and 16-bit architectures
- Low latency
- High throughput for short messages



Properties

Conclusion

Streamhash2 Advantages – Performance

- Easy to parallelize internal structure
- Negligible performance impact of machine endianness
- High performance on 8-bit and 16-bit architectures
- Low latency
- High throughput for short messages

StreamHash2 Design Properties

Conclusion

StreamHash2 Disadvantages

• Expensive hardware implementation

- Side-channel attacks on S-BOX lookups
- Mathematical background not well studied in cryptographic applications



イロト イポト イヨト イヨト

Properties

Conclusion

StreamHash2 Disadvantages

- Expensive hardware implementation
- Side-channel attacks on S-BOX lookups



▲ 同 ▶ ▲ 臣 ▶ .

StreamHash2 Design Properties

StreamHash2 Disadvantages

- Expensive hardware implementation
- Side-channel attacks on S-BOX lookups
- Mathematical background not well studied in cryptographic applications





• A new family of cryptographic hash functions was proposed

 Security properties of this new family require some further analysis



イロト イポト イヨト イヨト

Conclusion

- A new family of cryptographic hash functions was proposed
- Security properties of this new family require some further analysis



ъ