StreamHash2 Hash Function

Michał Trojnara

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

26 May 2010
Outline

1. Origins of StreamHash Family
   - History
   - Prior Cryptanalysis

2. Hash Functions
   - Requirements
   - Traditional Design

3. StreamHash2
   - StreamHash2 Design
   - Properties

4. Conclusion
1 Origins of StreamHash Family
   - History
   - Prior Cryptanalysis

2 Hash Functions
   - Requirements
   - Traditional Design

3 StreamHash2
   - StreamHash2 Design
   - Properties

4 Conclusion
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
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
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
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
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
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
Origins of StreamHash Family

1. Origins of StreamHash Family
   - History
   - Prior Cryptanalysis

2. Hash Functions
   - Requirements
   - Traditional Design

3. StreamHash2
   - StreamHash2 Design
   - Properties

4. Conclusion
Preimage Attack

- Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg

  - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
  - Complexity of $\frac{n}{2} \cdot 2^{n/2}$ for finding preimages

- Issue addressed in StreamHash2 by introducing a counter
Preimage Attack

- Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg
  - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
  - Complexity of $\frac{n}{2} \cdot 2^{n/2}$ for finding preimages
- Issue addressed in StreamHash2 by introducing a counter
Preimage Attack

- Dmitry Khovratovich and Ivica Nikolić, University of Luxembourg
  - Complexity of $\frac{n}{2} \cdot 2^{n/4}$ for finding collisions
  - Complexity of $\frac{n}{2} \cdot 2^{n/2}$ for finding preimages
- Issue addressed in StreamHash2 by introducing a counter
Tor E. Bjørstad, Department of Informatics, University of Bergen, Norway

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

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

History
Prior Cryptanalysis

Hash Functions

Requirements
Traditional Design

StreamHash2

StreamHash2 Design
Properties

Conclusion
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 $m$
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 $m$
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 $m$
**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 $h(m_1) = h(m_2)$
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 $h(m_1) = h(m_2)$
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 $h(m_1) = h(m_2)$
Origins of StreamHash Family

History

Prior Cryptanalysis

Hash Functions

Requirements

Traditional Design

StreamHash2

StreamHash2 Design

Properties

Conclusion
Merkle-Damgård Construction

IV → compression function → data block 1 → compression function → data block 2 → ... → compression function → data block n with padding → finalization → hash value
Davies-Meyer Compression Function

\[ H_i \leftarrow E_{m_i}(H_{i-1}) \oplus H_{i-1} \]
Origins of StreamHash Family
- History
- Prior Cryptanalysis

Hash Functions
- Requirements
- Traditional Design

StreamHash2
- StreamHash2 Design
- Properties

Conclusion
State Vector

State vector consists of 32-bit words

- \(7 \times 32 = 224\) bits
- \(8 \times 32 = 256\) bits
- \(12 \times 32 = 384\) bits
- \(16 \times 32 = 512\) bits
NLF Transformation

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

\[ \text{NLF} \]

\[ \text{state}_i \quad \text{bc} \quad \text{state}_{i+1} \]

\[ i, c \]
StreamHash Family Structure
\[ \text{state}_{i+1} \leftarrow \text{state}_i \oplus S-BOX[\text{LSB}(\text{state}_i) \oplus b \oplus i] \oplus c \]

, where:

- \( b \)  processed byte value
- \( c \) processed byte index
- \( i \) state vector index
- \( S-BOX \) S-BOX table
- \( \text{state} \) state vector
Origins of StreamHash Family

1. History
2. Prior Cryptanalysis

Hash Functions

3. Requirements
4. Traditional Design

StreamHash2

5. StreamHash2 Design
6. Properties

Conclusion
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 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 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 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 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 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 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 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 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 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 Disadvantages

- Expensive hardware implementation
- Side-channel attacks on S-BOX lookups
- Mathematical background not well studied in cryptographic applications
StreamHash2 Disadvantages

- Expensive hardware implementation
- Side-channel attacks on S-BOX lookups
- Mathematical background not well studied in cryptographic applications
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.
A new family of cryptographic hash functions was proposed. Security properties of this new family require some further analysis.