StreamHash2 Hash Function

Michał Trojnara

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

26 May 2010

Outline

Next Section

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

Next Section

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 $\frac{n}{2} \cdot 2^{n/2}$ for finding preimages
- Issue addressed in StreamHash2 by introducing a counter

Preimage Attack

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

Preimage Attack

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

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 $\boxplus$

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 $\boxplus$

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 $\boxplus$

Next Section

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

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

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

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

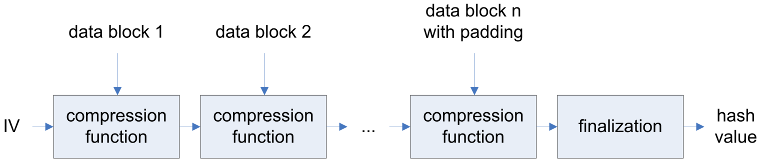
- **Preimage resistance**
  Practically infeasible for any given $h(m)$ to compute m
- **Second preimage resistance**
  Practically infeasible for any given m_1 message it is infeasible to find another m_2 such that $h(m_1) = h(m_2)$
- **Collision resistance**
  Practically infeasible to find two different messages m_1 and m_2 such that $h(m_1) = h(m_2)$

Security Requirements

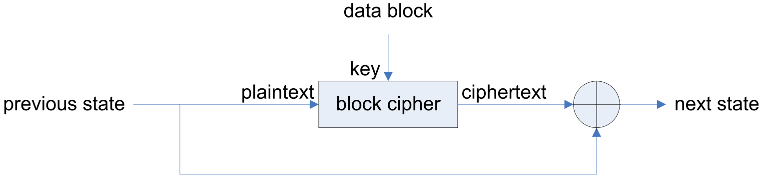
- **Preimage resistance**
  Practically infeasible for any given $h(m)$ to compute m
- **Second preimage resistance**
  Practically infeasible for any given m_1 message it is infeasible to find another m_2 such that $h(m_1) = h(m_2)$
- **Collision resistance**
  Practically infeasible to find two different messages m_1 and m_2 such that $h(m_1) = h(m_2)$

Next Section

Merkle-Damgård Construction



Davies-Meyer Compression Function



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

Next Section

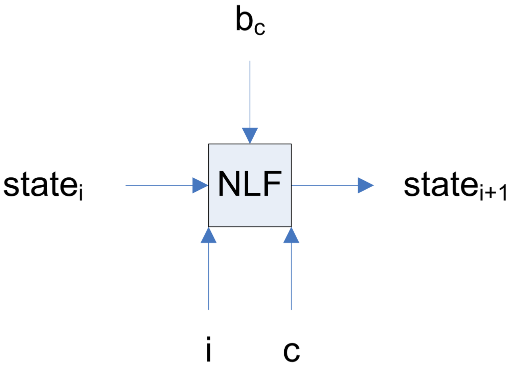
State Vector

State vector consists of 32-bit words

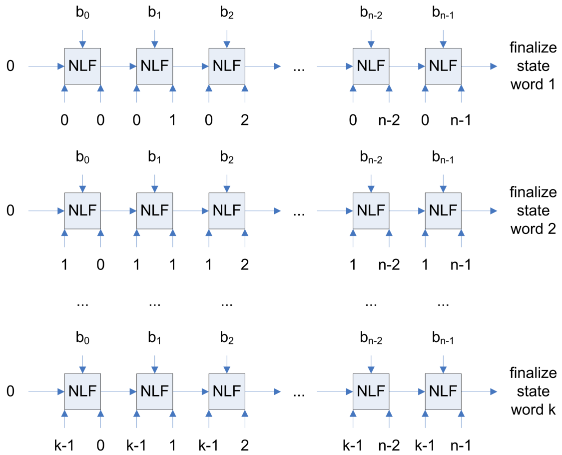
- $7 \times 32 = 224$ bits
- $8 \times 32 = 256$ bits
- $12 \times 32 = 384$ bits
- $16 \times 32 = 512$ bits

NLF Transformation

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



StreamHash Family Structure



NLF Implementation of StreamHash2 Function

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

, where:

- b processed byte value
- c processed byte index
- i state vector index
- $S\text{-}BOX$ S-BOX table
- $state$ state vector

Next Section

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

Conclusion

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

Conclusion

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