

NaSHA **family of cryptographic** **hash functions**

Smile Markovski
University "Ss. Cyril and Methodius"- Skopje

Aleksandra Mileva
University "Goce Delčev" - Štip

REPUBLIC OF MACEDONIA

The First SHA-3 Candidate Conference
Leuven, 2009

Contributors to implementations

- Simona Samardziski
- Boro Jakimovski

Outline

- Why NaSHA is special?
- Domain extender
- Compression function
- Software Implementation
- Security Analysis
- Cryptanalysis

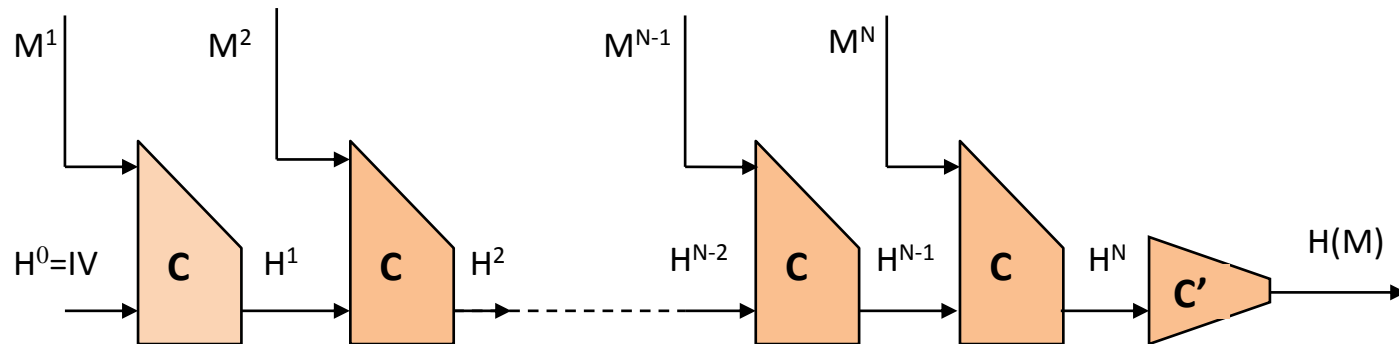
Why NaSHA is special?

- NaSHA use new quasigroup transformations
- NaSHA use novel design principle: the quasigroups used in every iteration in compression function are different and depend on the processed message block. Even in one iteration, different quasigroups are used for two quasigroup transformations. This is also what sets NaSHA apart from the playing field.
- Security of NaSHA can be changed with increasing the security parameter k

Why NaSHA is special?

- NaSHA uses known and well examined starting bijection, to exclude the fear of a trapdoor function.
- The definition of NaSHA allows the sizes of a message digest to be any number between 125 and 512.
- NaSHA can use quasigroups of different order, such as 2^{64} , 2^{128} , 2^{256} , etc.
- NaSHA is efficient

Wide-pipe domain extender



The message M is padded by standard MD-strengthening and then it is divided into N blocks $M = M^1 || M^2 || \dots || M^N$.

The message block M^i and the chaining variable H^i are divided into q 2^r -bits words. $M^i = m_1 || m_2 || \dots || m_q$ $H^i = H_1 || H_2 || \dots || H_q$

The internal state is formed alternatively from m_i and H_i .

$$S = m_1 || H_1 || m_2 || H_2 || \dots || m_q || H_q = S_1 || S_2 || \dots || S_{2q}$$

NaSHA-(m,k,r) algorithm

Input: Message M and message digest length m

Output: Message digest $H(M)$

For $i=1$ to N do

$$H^{i+1} = C(H^i, M^i) = \left(\mathcal{M}7 \left(\text{LinTr}_s^{2q}(S) \right) \right)_{2,4,\dots,2q} = Z_2 \parallel Z_4 \parallel \dots \parallel Z_{2q}$$

$$H(M) = C'(H^N) = Z_4 \parallel Z_8 \parallel \dots \parallel Z_{2q-4} \parallel Z_{2q} \pmod{2^m}$$

$$2^{2^r}$$

NaSHA-(m,k,r) algorithm

Input: Message M and message digest length m

Output: Message digest $H(M)$

For $i=1$ to N do

m – length of message digest

Z_{2q}

k – number of composite quasigroup transformations in \mathcal{M}

$H(M)$

r – order 2^{2^r} of used quasigroup

Compression function

$$MC = \rho(\mathcal{RA}_{l_1}) \circ \mathcal{A}_{l_2} \circ \dots \circ \rho(\mathcal{RA}_{l_{k-1}}) \circ \mathcal{A}_{l_k}$$

$$\mathcal{A}_{l_i}(X_1 \parallel X_2 \parallel \dots \parallel X_{2q}) = (z_1 \parallel z_2 \parallel \dots \parallel z_{2q}) \Leftrightarrow z_j = \begin{cases} (l_i + X_1) *_{1} X_1 & j=1 \\ (z_{j-1} + X_j) *_{1} X_j & 2 \leq j \leq 2q \end{cases}$$

$$\mathcal{RA}_{l_i}(X_1 \parallel X_2 \parallel \dots \parallel X_{2q}) = (Z_1 \parallel Z_2 \parallel \dots \parallel Z_{2q}) \Leftrightarrow Z_j = \begin{cases} X_j *_{2} (X_j + Z_{j+1}) & 1 \leq j \leq 2q-1 \\ X_{2q} *_{2} (X_{2q} + l_i) & j=2q \end{cases}$$

where $+$ is addition modulo 2^{2^r} , $*_{1}$ and $*_{2}$ are quasigroup operations for quasigroups of order 2^{2^r} and ρ is rotation to the left of every word, for half of its bits.

Compression function

$$M7 = \rho(\mathcal{RA}_{l_1}) \circ A_{l_2} \circ \dots \circ \rho(\mathcal{RA}_{l_{k-1}}) \circ A_{l_k}$$

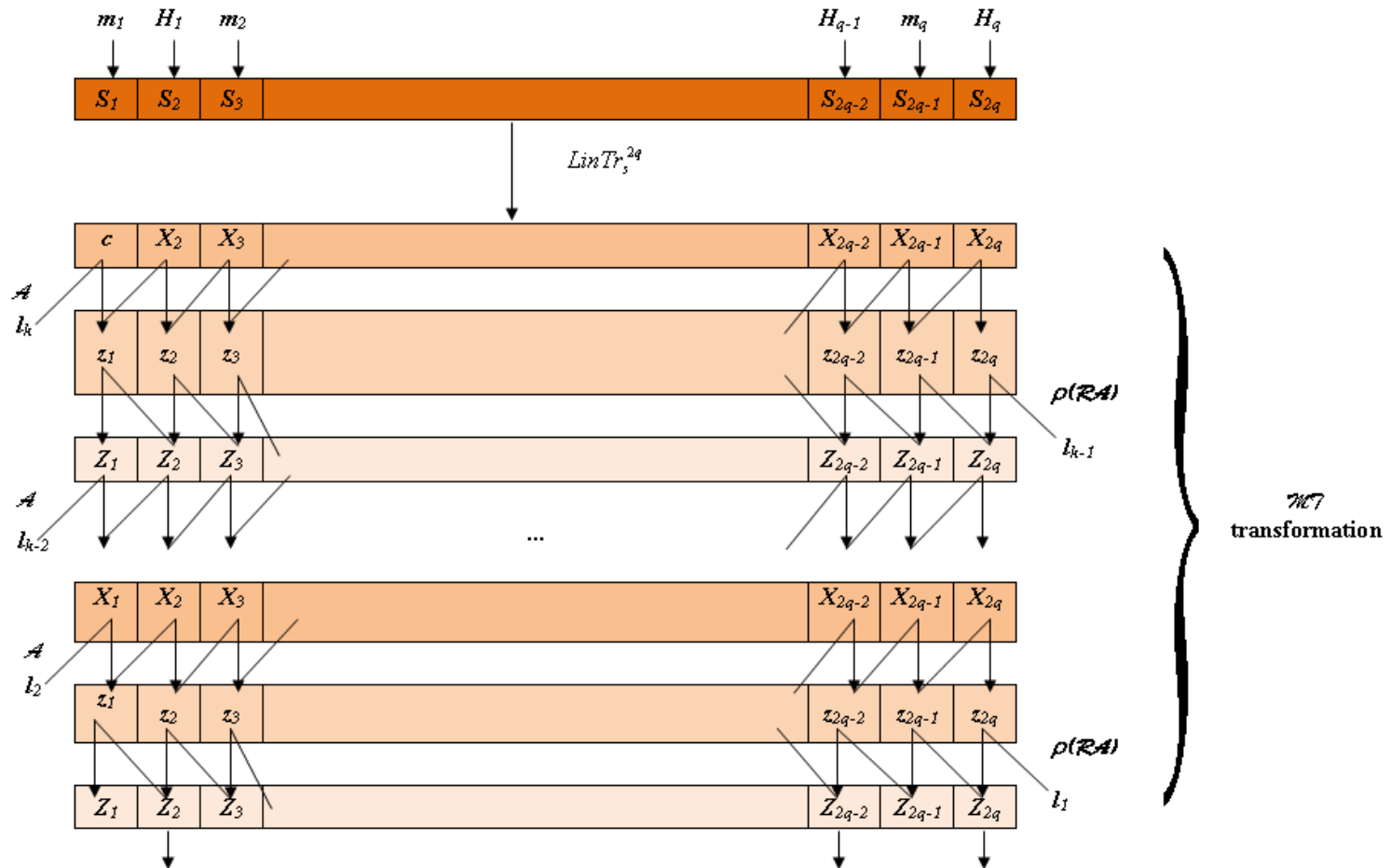
$M7$ consists of k quasigroup transformations
 A and \mathcal{RA} alternate.

$$\mathcal{RA}_{l_i}(X_1 \parallel X_2 \parallel \dots \parallel X_{2q}) = (Z_1 \parallel Z_2 \parallel \dots \parallel Z_{2q}) \Leftrightarrow Z_j = \begin{cases} X_j *_2 (X_j + Z_{j+1}) & 1 \leq j \leq 2q-1 \\ X_{2q} *_2 (X_{2q} + l_i) & j = 2q \end{cases}$$

where $+$ is addition modulo 2^{2^r} , $*_1$ and $*_2$ are quasigroup operations for quasigroups of order 2^{2^r} and ρ is rotation to the left of every word, for half of its bits.

Compression function

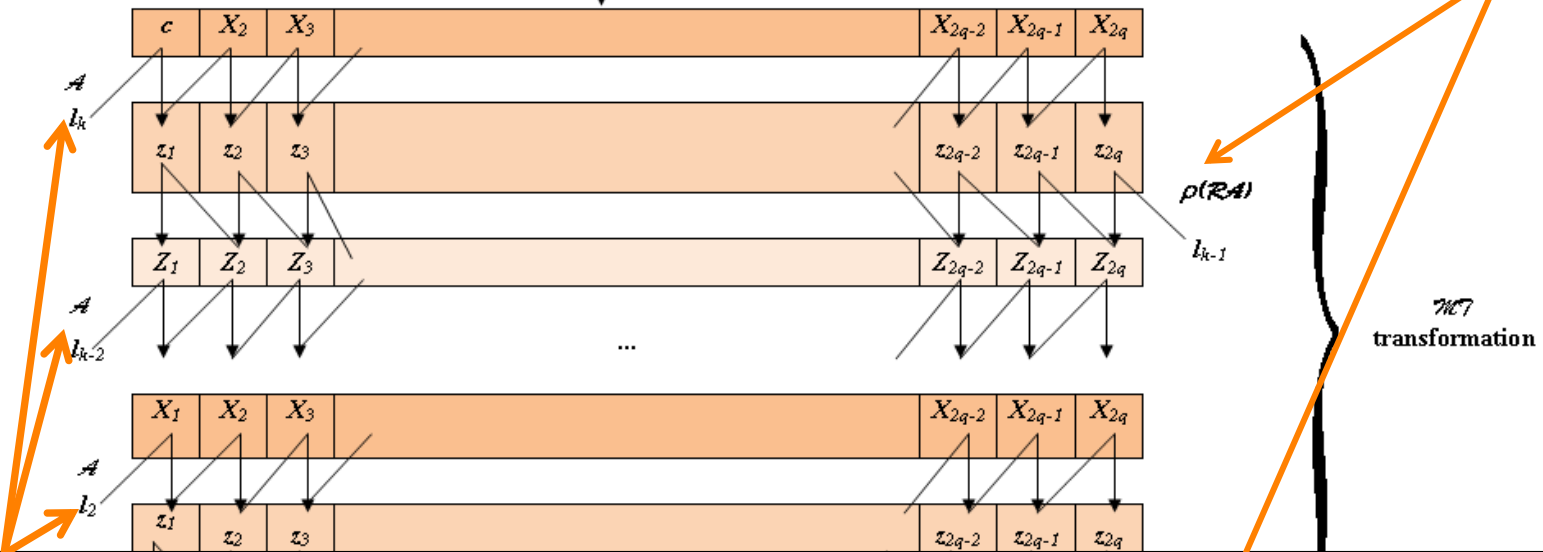
Quasigroup transformations



Compression function

Quasigroup transformations

RA transforms the state S from the end to the beginning



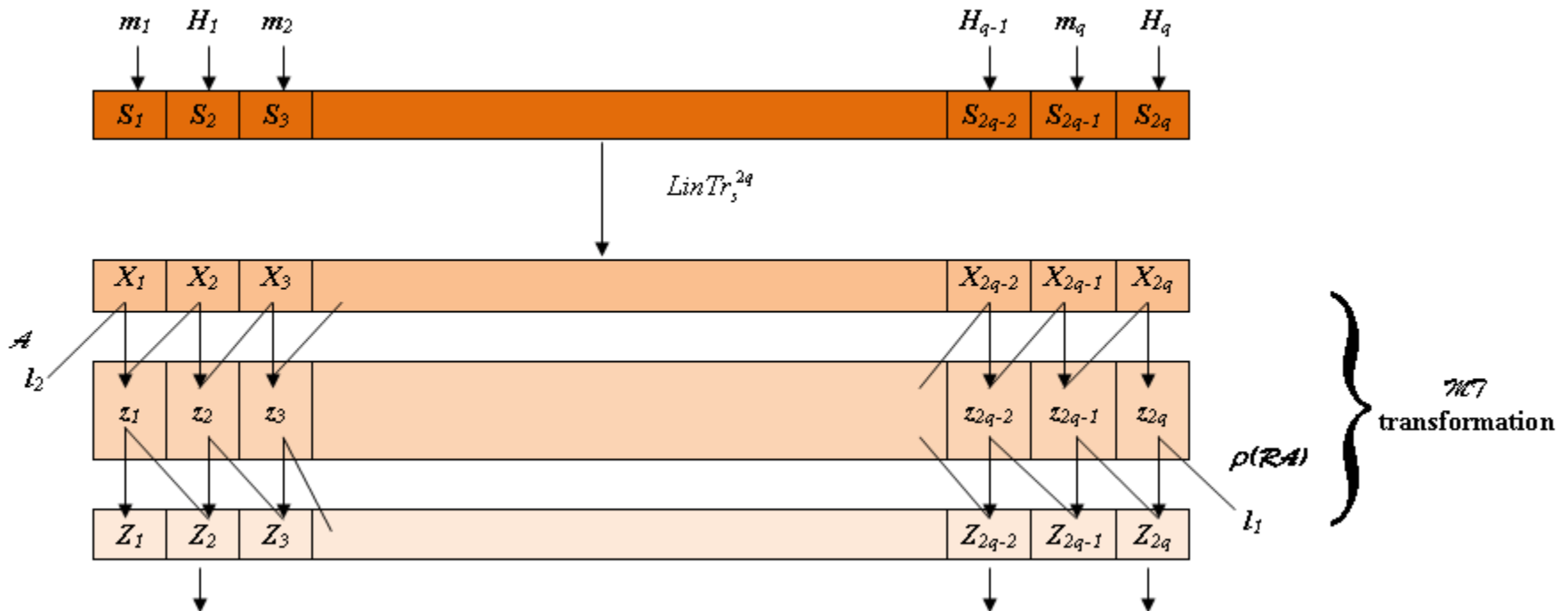
A transforms the state S from the beginning to the end

Software implementation Properties

Algorithm NaSHA-(m, k, r)	Word size (bits)	Message size (bits)	Block size (bits)	Internal state size (bits)	Message Digest size (bits)	LinTr parameter s	Message word number q
NaSHA-(224,2,6)	64	$<2^{128}$	512	1024	224	256	8
NaSHA-(256,2,6)	64	$<2^{128}$	512	1024	256	256	8
NaSHA-(384,2,6)	64	$<2^{128}$	1024	2048	384	512	16
NaSHA-(512,2,6)	64	$<2^{128}$	1024	2048	512	512	16

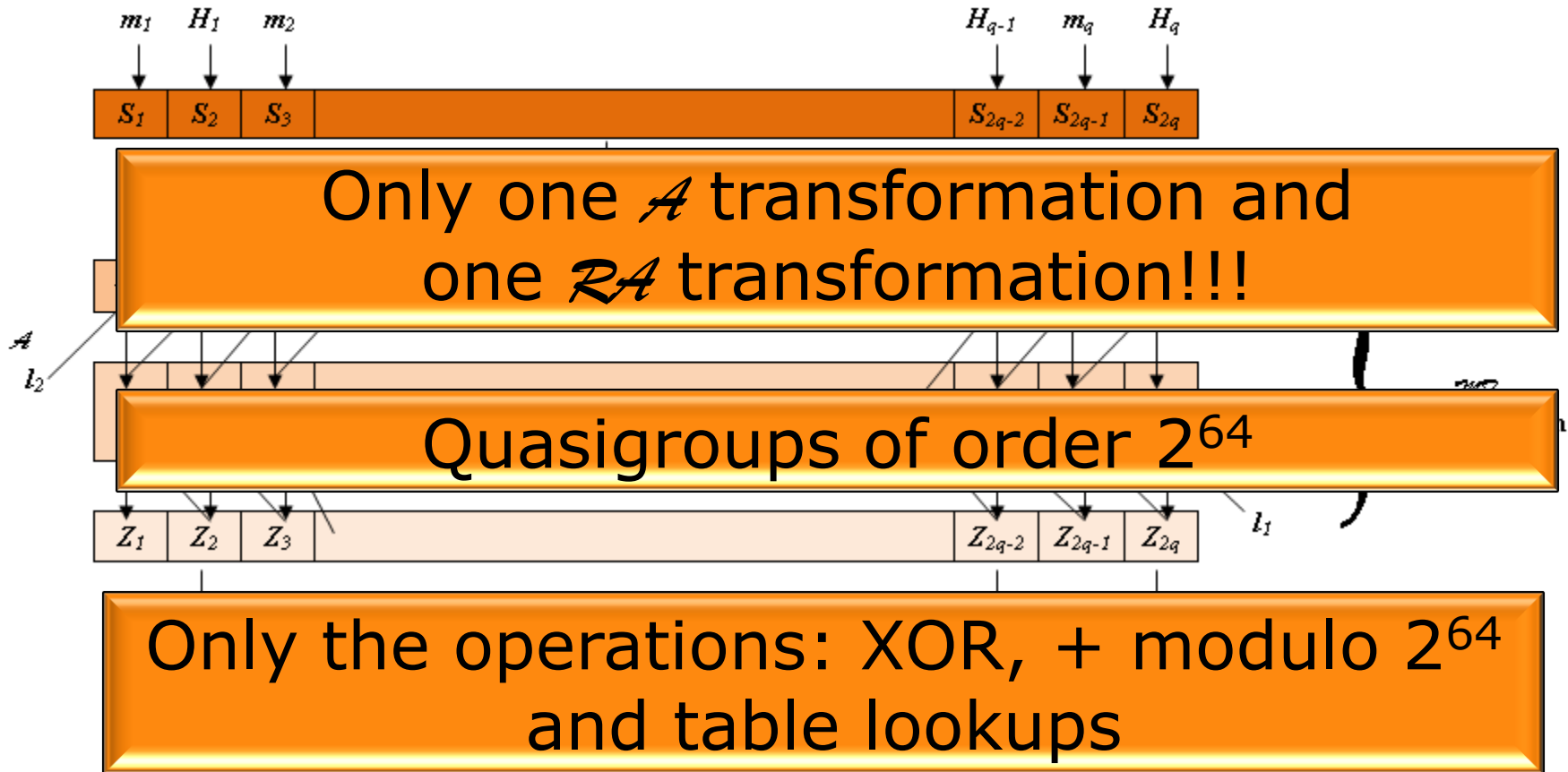
Software implementation

Quasigroup transformations



Software implementation

Quasigroup transformations



Software implementation

Linear transformations

$$\mathit{LinTr}_{256}(S_1 \parallel S_2 \parallel \dots \parallel S_{16}) = (S_4 \oplus S_7 \oplus S_{10} \oplus S_{16}) \parallel S_1 \parallel S_2 \parallel \dots \parallel S_{15}$$

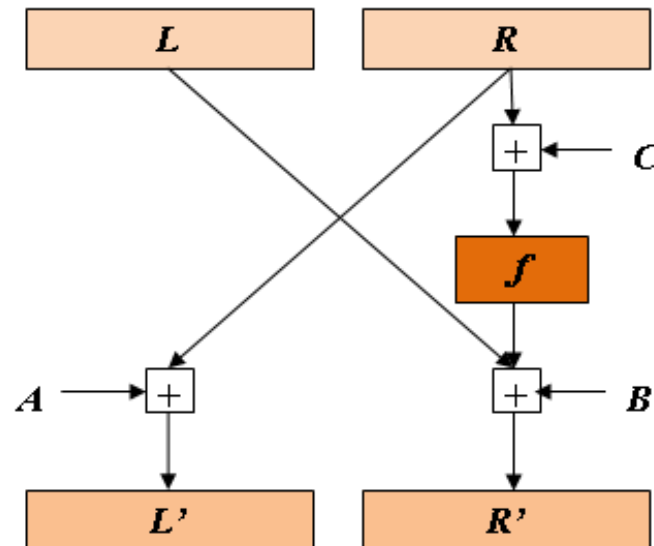
$$\mathit{LinTr}_{512}(S_1 \parallel S_2 \parallel \dots \parallel S_{32}) = (S_7 \oplus S_{15} \oplus S_{25} \oplus S_{32}) \parallel S_1 \parallel S_2 \parallel \dots \parallel S_{31}$$

Software implementation

Extended Feistel Network - EFN

Let $(G, +)$ be an Abelian group, let $f : G \rightarrow G$ be a mapping and let $A, B, C \in G$ be fixed elements. The Extended Feistel Network $F_{A,B,C} : G \rightarrow G$ created by f is defined for every $L, R \in G$ by

$$F_{A,B,C}(L \parallel R) = (R + A) \parallel (L + B + f(R + C))$$



Software implementation

Quasigroup operation via EFN

$$F_{a_i, b_i, c_i}(l_8 \parallel r_8) = (r_8 \oplus a_i) \parallel (l_8 \oplus b_i \oplus \text{sbox}(r_8 \oplus c_i)), i = 1, 2, 3$$

$$f' = F_{a_1, b_1, c_1} \circ F_{a_2, b_2, c_2} \circ F_{a_3, b_3, c_3}$$

$$F_{\alpha_i, \beta_i, \gamma_i}(l_{16} \parallel r_{16}) = (r_{16} \oplus \alpha_i) \parallel (l_{16} \oplus \beta_i \oplus f'(r_{16} \oplus \gamma_i)), i = 1, 2$$

$$F_{A_i, B_i, C_i}(l_{32} \parallel r_{32}) = (r_{32} \oplus A_i) \parallel (l_{32} \oplus B_i \oplus F_{\alpha_i, \beta_i, \gamma_i}(r_{32} \oplus C_i)), i = 1, 2$$

Definition of quasigroup operation for \mathcal{A} transformation

$$X *_1 Y = F_{A_1, B_1, C_1}(X \oplus Y) \oplus Y$$

Definition of quasigroup operation for \mathcal{RA} transformation

$$X *_2 Y = F_{A_2, B_2, C_2}(X \oplus Y) \oplus Y$$

Software implementation

Quasigroup operation via EFN

$$F_{a_i, b_i, c_i}(l_8 \parallel r_8) = (r_8 \oplus a_i) \parallel (l_8 \oplus b_i \oplus sbox(r_8 \oplus c_i)), i = 1, 2, 3$$

$$f' = F_{a_1, b_1, c_1} \circ F_{a_2, b_2, c_2} \circ F_{a_3, b_3, c_3}$$

Quasigroups of order 2^{64}
 defined by Extended Feistel Network
 as complete mappings by Sade's
 diagonal method.

$$X *_1 Y = F_{A_1, B_1, C_1}(X \oplus Y) \oplus Y$$

Definition of quasigroup operation for \mathcal{RA} transformation

$$X *_2 Y = F_{A_2, B_2, C_2}(X \oplus Y) \oplus Y$$

Software implementation

Parameters for quasigroup transformations

In implementations we use the Abelian groups (\mathbb{Z}_2^p, \oplus) , where p is 8, 16, 32. The definitions of the parameters of the quasigroup operations use addition modulo 2^p :

$$l_1 = S_1 + S_2 \qquad l_2 = S_3 + S_4$$

$$a_1 \parallel b_1 \parallel c_1 \parallel a_2 \parallel b_2 \parallel c_2 \parallel a_3 \parallel b_3 = S_5 + S_6, c_3 = a_1$$

$$\alpha_1 \parallel \beta_1 \parallel \gamma_1 \parallel \alpha_2 = S_7 + S_8$$

$$\beta_2 \parallel \gamma_2 = (S_9 + S_{10}) \pmod{2^{32}}$$

$$A_1 \parallel B_1 = S_{11} + S_{12}$$

$$C_1 \parallel A_2 = S_{13} + S_{14}$$

$$B_2 \parallel C_2 = S_{15} + S_{16}$$

Software implementation

Parameters for quasigroup transformations

In implementations we use the Abelian groups (\mathbb{Z}_2^p, \oplus) , where p is 8, 16, 32. The definitions of the parameters of the quasigroup operations use addition modulo 2^p .

Parameters of
quasigroup operations
depend on input message blocks
and are different for every iteration
of compression function and for
every used quasigroup
transformation!!!

NaSHA Performances

Platform 1 NIST SHA-3 Reference Platform

Platform 2 same, only compiled by Intel C++ v11.0.066

NaSHA performance in Cycles/Byte with message length of 100000B

Algorithm	Platform 1		Platform 2	
	32-bit	64-bit	32-bit	64-bit
NaSHA-(224,2,6)	35.74 (38.97)	26.38 (28.31)	27.30	26.53
NaSHA-(256,2,6)	35.78 (39.01)	26.38 (28.43)	27.32	26.53
NaSHA-(384,2,6)	34.30 (37.64)	27.23 (29.39)	30.54	27.01
NaSHA-(512,2,6)	34.30 (38.97)	27.08 (29.39)	30.55	26.79

NaSHA Performances

Memory requirements
only **0,625KB**

NaSHA performance in Cycles/Byte with message length of 100000B

Algorithm	Platform 1		Platform 2	
	32-bit	64-bit	32-bit	64-bit
NaSHA-(224,2,6)	35.74 (38.97)	26.38 (28.31)	27.30	26.53
NaSHA-(256,2,6)	35.78 (39.01)	26.38 (28.43)	27.32	26.53
NaSHA-(384,2,6)	34.30 (37.64)	27.23 (29.39)	30.54	27.01
NaSHA-(512,2,6)	34.30 (38.97)	27.08 (29.39)	30.55	26.79

Security analysis

Wide pipe domain extender provide resistance to some generic attacks like:

- ❖ **Joux multicollision attack**
- ❖ **length extension attack**
- ❖ **Dean fixed point attack**
- ❖ **Kelsey and Schneier long message 2nd preimage attack**
- ❖ **Kelsey and Kohno herding attack**
- ❖ **2nd collision attack**

Cryptanalysis

1. *Li Ji, Xu Liangyu and Guan Xu* – **Free start collisions for all versions of NaSHA with examples. Collision attack on NaSHA-512 with claimed complexity of the attack is 2^{192} .**

(Cryptology ePrint Archive, Report 2008/519, 2008)

S. Markovski, A. Mileva, V. Dimitrova and D. Gligoroski confirmed that this attack has unknown probability, because attackers use a system of three quasigroup equations with five variables. Their claim will be true if this kind of systems always has a solution. But this is not true. There are examples of these kinds of systems with no solutions for quasigroups of order 4.

(Cryptology ePrint Archive, Report 2009/034, 2009)

Cryptanalysis

2. *Zhimin Li, Daofeng Li* –
Collision attack on NaSHA-384/512 with claimed complexity of the attack is 2^{128} with probability $\left(1 - \frac{2}{2^{64}-1}\right)^2 \left(>> \frac{1}{2}\right)$

(Cryptology ePrint Archive, Report 2009/026, 2009)

S. Markovski, A. Mileva and V. Dimitrova confirmed that this attack is a variation of the previous one. The attackers' claiming will be true if a system of two quasigroup equations with five variables always has a solution. This is not generally true, we give an example of a system of two quasigroup equations with 4 unknown that has no solution in a quasigroup of order 8.

(http://inf.ugd.edu.mk/images/stories/file/Mileva/nasha_hf.html)

Cryptanalysis

- 3.** *Ivica Nikolić, Dmitry Khovratovich* –
Free-start collision attacks on NaSHA with complexity of 2^{32}
Free-start preimage attack on NaSHA-n with complexity of $2^{n/2}$

(http://e-hash.iaik.tugraz.at/uploads/3/33/Free-start_attacks_on_Nasha.pdf)

Cryptanalysis

Minor improvements of NaSHA

All three attacks can be fixed if a slight modification of the quasigroup parameters is made, so every parameter to depend on two more other words. For example:

$$\begin{aligned}l_1 &= S_1 + S_2 + S_{28} + S_{30} & l_2 &= S_3 + S_4 + S_{29} + S_{31} \\a_1 \parallel b_1 \parallel c_1 \parallel a_2 \parallel b_2 \parallel c_2 \parallel a_3 \parallel b_3 &= S_5 + S_6 + S_{17} + S_{18}, \\ \alpha_1 \parallel \beta_1 \parallel \gamma_1 \parallel \alpha_2 &= S_7 + S_8 + S_{19} + S_{20} & c_3 &= a_1 \\ \beta_2 \parallel \gamma_2 &= (S_9 + S_{10} + S_{21} + S_{22}) \pmod{2^{32}} \\ A_1 \parallel B_1 &= S_{11} + S_{12} + S_{23} + S_{27} \\ C_1 \parallel A_2 &= S_{13} + S_{14} + S_{24} + S_{26} \\ B_2 \parallel C_2 &= S_{15} + S_{16} + S_{25} + S_{32}\end{aligned}$$

**Thanks for yours
attention!!!**

The First SHA-3 Candidate Conference
Leuven, 2009