NaSHA
Cryptographic Hash Function

2.B Algorithm Specifications and Supporting Documentations
2.B.1 Algorithm Specifications

Designers:

Smile Markovski and Aleksandra Mileva

Implementation Contributors:

Simona Samardziska and Boro Jakimovski

Skopje and Štip, MACEDONIA, 2008

Abstract

We propose the NaSHA-(m, k, r) family of cryptographic hash functions, based on quasigroup transformations. We use huge quasigroups defined by extended Feistel networks from small bijections and a novel design principle: the quasigroup used in every iteration of the compression function is different and depends on the processed message block. We present in all details of the implementations of NaSHA-$(m, 2, 6)$ where $m \in \{224, 256, 384, 512\}$.

1 Introduction

In this part we give the algorithm specification of our NaSHA hash function, consisting of 5 sections: 2. Mathematical background, 3. The NaSHA-(m, k, r) hash algorithm, 4. Implementation of NaSHA-$(m, 2, 6)$ hash functions for $m \in \{224, 256, 384, 512\}$, 5. Design rationale and 6. Preliminary security analysis.

2 Mathematical background

2.1 Quasigroups

A quasigroup $(Q, *)$ is a groupoid, i.e., a set Q with a binary operation $*$, such that the equations $a * x = b$ and $y * a = b$ have unique solutions x and y in Q for each given $a, b \in Q$. Note that when Q is finite then the main body of the multiplication table of $(Q, *)$ is a Latin square, i.e., the rows and the columns are permutations of Q. Given a quasigroup $(Q, *)$, two adjoint operations $/$ and $\backslash$ can be defined by $x/y = z \iff x = z * y$ and $x \backslash y = z \iff x * z = y$. Then the groupoids $(Q, /)$ and $(Q, \backslash)$ are quasigroups too.

By a quasigroup of a good cryptographic quality we mean a finite quasigroup that is non-commutative, non-associative, non-idempotent, without right or left units and without a proper sub-quasigroups. That quasigroup $(Q, *)$ should not be linear, in the sense that no output bit of $a * b$ is a linear combination of the input bits of a and b, for each $a, b \in Q$. Also, the quasigroup should not satisfy identities of the kinds $\underbrace{x(\ldots * (x}_{l} * y)) = y$ and $y = ((y * \underbrace{x) * \ldots) * x}_{l}$ for some $l < 2n$, where n is the order of the quasigroup.

2.2 Quasigroup transformation used in NaSHA

For NaSHA hash family we use the following quasigroup transformations.

Definition 1 (**Quasigroup additive string transformation $\mathcal{A}_l : Q^t \to Q^t$ with leader** l) *Let t be a positive integer, let $(Q, *)$ be a quasigroup, $Q = \mathbb{Z}_{2^n}$, and $l, x_j, z_j \in Q$. The transformation $\mathcal{A}_l$ is defined by*

$$\mathcal{A}_l(x_1, \ldots, x_t) = (z_1, \ldots, z_t) \Leftrightarrow z_j = \begin{cases} (l + x_1) * x_1, \ j = 1 \\ (z_{j-1} + x_j) * x_j, \ 2 \leq j \leq t \end{cases} \tag{1}$$

where $+$ is addition modulo 2^n. The element l is said to be a leader of $\mathcal{A}$.

Definition 2 (**Quasigroup reverse additive string transformation $\mathcal{RA}_l : Q^t \to Q^t$ with leader** l) *Let t be a positive integer, let $(Q, *)$ be a quasigroup, $Q = \mathbb{Z}_{2^n}$, and $l, x_j, z_j \in Q$. The transformation $\mathcal{RA}_l$ is defined by*

$$\mathcal{RA}_l(x_1, \ldots, x_t) = (z_1, \ldots, z_t) \Leftrightarrow z_j = \begin{cases} x_j * (x_j + z_{j+1}), \ 1 \leq j \leq t-1 \\ x_t * (x_t + l), \ j = t \end{cases} \tag{2}$$

where $+$ is addition modulo 2^n. The element l is said to be a leader of $\mathcal{RA}$.

For an element $z \in \mathbb{Z}_{2^n}$ denote by $\rho(z, \lfloor \frac{n}{2} \rfloor)$ the element in $\mathbb{Z}_{2^n}$ obtained by rotating left for $\lfloor \frac{n}{2} \rfloor$ bits the n-bit representation of z. Given a string $Z = (z_1, \ldots, z_t) \in (\mathbb{Z}_{2^n})^t$, we denote by $\rho(Z)$ the string

$$\rho(Z) = \big(\rho(z_1, \lfloor \frac{n}{2} \rfloor), \ldots, \rho(z_t, \lfloor \frac{n}{2} \rfloor)\big) \in (\mathbb{Z}_{2^n})^t.$$

For a function $f = f(Z)$ we define a new function $\rho(f) = \rho(f)(Z)$ by $\rho(f)(Z) = f(\rho(Z))$.

Definition 3 (**Quasigroup main transformation** $\mathcal{MT} : Q^t \to Q^t$) *Let $Q = \mathbb{Z}_{2^n}$ and let t and k be positive integers, where k is even. (k is called the complexity of $\mathcal{MT}$.) The transformation $\mathcal{MT}$ is defined as composition of transformations of kind $\mathcal{A}_{l_i}$ followed by $\rho(\mathcal{RA}_{l_j})$, for suitable choices of the leaders l_i and l_j as functions depending on variables $x_1, x_2, \dots, x_t$, as follows. For every $x_\lambda \in Q$*

$$\mathcal{MT}(x_1, \dots, x_t) = \rho(\mathcal{RA}_{l_1})(\mathcal{A}_{l_2}(\dots(\rho(\mathcal{RA}_{l_{k-1}})(\mathcal{A}_{l_k}(x_1, \dots, x_t)))\dots)), \tag{3}$$

i.e., $\mathcal{MT} = \rho(\mathcal{RA}_{l_1}) \circ \mathcal{A}_{l_2} \circ \dots \circ \rho(\mathcal{RA}_{l_{k-1}}) \circ \mathcal{A}_{l_k}$, where $\circ$ denotes a composition of functions.

2.3 Left and right quasigroups

A groupoid $(G, \cdot)$ is said to be a left (a right) quasigroup if the equation $xa = b$ ($ay = b$) have a unique solution x (y) in G for every $a, b \in G$.

Proposition 1 *Let $(G, +)$ be a group and let $(G, *)$ be a quasigroup. Then the operation $\bullet$ defined by $x \bullet y = (x + y) * y$ defines a left quasigroup $(G, \bullet)$.*

Proof The solution $x = (b/a) - a$ of the equation $x \bullet a = b$ is unique, since $x \bullet y = x' \bullet y \Longrightarrow x = x'$. $\square$

Proposition 2 *Let $(G, +)$ be a group and let $(G, *)$ be a quasigroup. Then the operation $\diamond$ defined by $x \diamond y = x * (x + y)$ defines a right quasigroup $(G, \diamond)$.*

Proof The solution $y = -a + (a \backslash b)$ of the equation $a \diamond y = b$ is unique, since $x \diamond y = x \diamond y' \Longrightarrow y = y'$. $\square$

Given a groupoid $(G, \cdot)$, for each $a \in G$ the left and the right translations L_a and R_a are defined by $L_a(x) = xa$ and $R_a(x) = ax$ respectfully. If $(G, \cdot)$ is a left (right) quasigroup then its left (right) translation is a permutation, while the right (left) translation can be arbitrary mapping.

Considering the left and the right quasigroups defined as in Proposition 1 and Proposition 2, the situation is quite different in the case when $G = \mathbb{Z}_{2^n}$ and the group operation is addition modulo 2^n. Namely, the right translation of $(G,\bullet)$ and the left translation of $(G,\diamond)$ may not be permutations in that case either. However, the probability of that event is quite small, roughly speaking, around $2/|G|$. To show the last statement we consider the problem of finding solutions of the equation $x\diamond a=b$, i.e.,

$$x*(x+a)=b \tag{4}$$

where $a,b\in G$ are given, and x is unknown.

Proposition 3 *Let $G=\mathbb{Z}_{2^n}$ be with group operation addition modulo 2^n. Let a quasigroup operation $*$ on G be chosen randomly. Then the probability the right quasigroup $(G,\diamond)$ to have two different solutions $x_1\neq x_2$ of the equation (4) is less or equal to $\dfrac{2}{2^n-1}$.*

Proof Let x_1 and x_2 be two different solutions of the equation $x*(x+a)=b$. Then

$$\left\{\begin{array}{l} x_1*(x_1+a)=b\\ x_2*(x_2+a)=b\end{array}\right. \Rightarrow \left\{\begin{array}{l} x_1\setminus b-x_1=a\\ x_2\setminus b-x_2=a\end{array}\right. \Rightarrow x_1\backslash b-x_2\backslash b=x_1-x_2\neq 0.$$

At first, we find the probability a random quasigroup to satisfy the event $x_1\setminus b-x_2\setminus b=x_1-x_2\neq 0$.

The difference x_1-x_2 can take any value $r\in G$, where $r\neq 0$. Fix an $r\neq 0$. Then there are $\binom{2^n}{2}$ pairs of different elements of G, and exactly 2^n of them satisfy the equation $x_1-x_2=r$. Hence, we have this probability for any fixed $r\neq 0:\quad P_r\{x_1,x_2\in G, x_1-x_2=r\}=\frac{2}{2^n-1}$.

Consider now the equation $x_1\backslash b-x_2\backslash b=s$, where $s\neq 0\in G$ is given. Denote by K the set of all quasigroups on G and let fix a solution (x_1,x_2) of $x_1\backslash b-x_2\backslash b=s$. Denote by $K_s=K_s(x_1,x_2)$ the set of all quasigroups on G with the property $x_1\backslash b-x_2\backslash b=s$. Then $|K_s|=|K_t|$ for each s and t. Namely, if $(G,\backslash_1)\in K_s$, then we can construct a quasigroup $(G,\backslash_2)\in K_t$ as follows. At first choose $x_1\backslash_2 b$ and $x_2\backslash_2 b$ such that $x_1\backslash_2 b-x_2\backslash_2 b=t$ and let π be the permutation generated by the two transpositions $(x_1\backslash_1 b, x_1\backslash_2 b),(x_2\backslash_1 b, x_2\backslash_2 b)$. Then define the operation $\backslash_2$ for each $u,v\in G$ by $u\backslash_2 v=\pi(u\backslash_1 v)$. (Note that we have obtained $(G,\backslash_2)$ from $(G,\backslash_1)$ in such a way that we have only replaced in the multiplication table of $(G,\backslash_1)$ all appearances of $x_1\backslash_1 b$ $(x_2\backslash_1 b)$ by $x_1\backslash_2 b$ $(x_2\backslash_2 b)$.) Now, for given $x_1,x_2\in G$ and randomly chosen quasigroup $(Q, \backslash)$, we have the probability $P_s\{Q \in K, x_1\backslash b - x_2\backslash b = s$ is true in Q$\} = \frac{|K_s|}{|K|} = \frac{1}{2^n-1}$.

Consequently, the probability a random quasigroup $(G, *)$ to satisfy the event $x_1 \setminus b - x_2 \setminus b = x_1 - x_2 \neq 0$ is

$$P\{x_1 - x_2 = r, x_1\backslash b - x_2\backslash b = r, r > 0\} =$$
$$\sum_{r=1}^{q-1} P\{x_1 - x_2 = r, x_1\backslash b - x_2\backslash b = r\} =$$
$$\sum_{r=1}^{2^n-1} P\{x_1\backslash b - x_2\backslash b = r|\ x_1 - x_2 = r\} P\{x_1 - x_2 = r\} =$$
$$\sum_{r=1}^{2^n-1} P_s\{Q \in K, x_1\backslash b - x_2\backslash b = r\} P_r\{x_1, x_2 \in G, x_1 - x_2 = r\} = \frac{2}{2^n - 1}.$$

Finally, if we additionally take the condition $x_1\backslash b - x_1 = a$, we conclude that the probability a right quasigroup $(G, \diamond)$ to have two different solutions $x_1 \neq x_2$ of the equation (4) is less or equal to $\frac{2}{2^n-1}$.
□

In similar way one can prove the same property for left quasigroup $(G, \bullet)$.

Proposition 4 *Let $G = \mathbb{Z}_{2^n}$ be with group operation addition modulo 2^n. Let a quasigroup operation $*$ on G be chosen randomly. Then the probability the left quasigroup $(G, \bullet)$ to have two different solutions $x_1 \neq x_2$ of the equation*

$$(a + x) * x = b \tag{5}$$

is less or equal to $\dfrac{2}{2^n - 1}$. □

Remark 1 In the set of all 576 quasigroups of order 4, each equation of kind $x * (x + a) = b$ (or $(a + x) * x = b$) has two (or more) solutions in exactly 168 quasigroups.

2.4 The main transformation $\mathcal{MT}$ as a one-way function

Next we show that the transformation $\mathcal{MT} : Q^t \to Q^t$ can be considered as a one-way function when $Q = \mathbb{Z}_{2^n}$ is enough big.

Let us take $k = 2$ for simplicity, and let a quasigroup $(Q, *)$, leaders l_1, l_2 and elements $c_1, c_2, \ldots, c_t \in Q$ be given. Suppose that for some unknown $x_1, x_2, \ldots, x_t \in Q$ we have $(c_1, c_2, \ldots, c_t) = \mathcal{MT}(x_1, x_2, \ldots, x_t)$ $= \rho(\mathcal{RA}_{l_1})(\mathcal{A}_{l_2}(x_1, x_2, \ldots, x_t))$. Then there are unknown $y_1, y_2, \ldots, y_t \in Q$ such that

$$\mathcal{A}_{l_2}(x_1, x_2, \ldots, x_t) = (y_1, y_2, \ldots, y_t) \tag{6}$$

and

$$\mathcal{RA}_{l_1}(\rho(y_1, \lfloor\frac{n}{2}\rfloor), \rho(y_2, \lfloor\frac{n}{2}\rfloor), \ldots, \rho(y_t, \lfloor\frac{n}{2}\rfloor)) = (c_1, c_2, \ldots, c_t). \tag{7}$$

From the equations (6) and (7) we obtain the following system of $2t$ equations with $2t$ unknowns.

$$\left\{\begin{array}{l} (l_2 + x_1) * x_1 = y_1 \\ (y_1 + x_2) * x_2 = y_2 \\ \ldots \\ (y_{t-1} + x_t) * x_t = y_t \end{array}\right. \tag{8}$$

$$\left\{\begin{array}{l} \rho(y_t, \lfloor\frac{n}{2}\rfloor) * (\rho(y_t, \lfloor\frac{n}{2}\rfloor) + l_1) = c_t \\ \rho(y_{t-1}, \lfloor\frac{n}{2}\rfloor) * \rho((y_{t-1}, \lfloor\frac{n}{2}\rfloor) + c_t) = c_{t-1} \\ \ldots \\ \rho(y_1, \lfloor\frac{n}{2}\rfloor) * (\rho(y_1, \lfloor\frac{n}{2}\rfloor) + c_2) = c_1. \end{array}\right. \tag{9}$$

The subsystem (9) consists of t equations with t unknowns of kind $y * (y + a) = b$. As much as we know, there is no explicit formula to find the unknown y, so one has to check for each $y \in Q$ if the equation $y * (y + a) = b$ is satisfied. By Proposition 4 one has to make, roughly, $2^n - 1/2^n \approx 2^n$ checks, i.e., a solution can be found after 2^{n-1} checks on average. In the same way, by checking, solutions $x_1, x_2, \ldots, x_t$ can be found. Altogether, for finding a solution of the system consisting of (8) and (9) one has to make, on average, $2t2^{n-1} = 2^n t$ checks. Thus, we have the following properties.

Proposition 5 *The system of equations (8) and (9) can be solved after $2^n t$ checks on average.* □

Proposition 6 *If Q is sufficiently large and $(Q, *)$ is an arbitrary quasigroup, chosen uniformly at random, the problem of finding a preimage of the transformation $\mathcal{MT}$ is computationally infeasible.* □

2.5 Definition of quasigroups by extended Feistel networks

The algorithms for computing the $NaSHA-(m,2,6)$ hash functions, for $m \in \{224, 256, 384, 512\}$, use quasigroups of order 2^{64} and in the sequel we give an effective construction of quasigroups of such a big order. For that aim we use the extended Feistel network defined in [15] as a generalization of the Feistel network [8]. The complete proofs of all statements given in this subsection can be found in [15] as well.

Let $(G,+)$ be an Abelian group, let $f : G \to G$ be a mapping and let $a, b, c \in G$ be fixed elements. The **extended Feistel network** $F_{a,b,c} : G^2 \to G^2$ created by f is defined for every $l, r \in G$ by

$$F_{a,b,c}(l,r) = (r + a, l + b + f(r + c)).$$

The extended Feistel network $F_{a,b,c}$ is a bijection with inverse

$$F^{-1}_{a,b,c}(l,r) = (r - b - f(l + c - a), l - a).$$

An extended Feistel network became a Feistel network when $a = b = c = 0$.

A **complete mapping** of a group $(G,+)$ is a bijection $\theta : G \to G$ such that the mapping $\phi : G \to G$ defined by $\phi(x) = -x + \theta(x)$ is again a bijection of G.

Theorem 1 *Let $(G,+)$ be an Abelian group and $a, b, c \in G$. If $F_{a,b,c} : G^2 \to G^2$ is an extended Feistel network created by a bijection $f : G \to G$, then $F_{a,b,c}$ is a complete mapping of the group $(G^2,+)$.* □

Sade [18] proposed the following method for creating a quasigroup from a group with a complete mapping :

Proposition 7 *Let $(G,+)$ be a group with complete mapping θ. Define an operation $*$ on G by:*

$$x * y = \theta(x - y) + y \tag{10}$$

where $x, y \in G$. Then $(G,*)$ is a quasigroup derived by θ. □

Proposition 7 and Theorem 1 allow us to construct iteratively quasigroups on the sets $\{0,1\}^{2^n}$ as follows.

Take a bijection $f : \{0,1\}^{2^t} \to \{0,1\}^{2^t}$, where $t < n$ is a small positive integer ($t = 1, \dots, 8$). Denote by $G_i = (\{0,1\}^{2^{t+i}}, \oplus_{2^{t+i}})$, $i = 1, 2, 3, \dots$, the groups with carrier $\{0,1\}^{2^{t+i}}$ and bitwise XOR operation $\oplus_{2^{t+i}}$. Choose constants $a^{(i)}, b^{(i)}, c^{(i)} \in \{0,1\}^{2^{t+i}}$, $1 \le i \le n-t$, and construct iteratively the complete mappings $F_1, F_2, F_3, \dots$ on the groups $G_1, G_2, G_3, \dots$ respectively in the following way. $F_1 = F_{a^{(1)},b^{(1)},c^{(1)}}$ is the extended Feistel network created by f, and $F_{i+1} = F_{a^{(i+1)},b^{(i+1)},c^{(i+1)}}$ is the extended Feistel network created by F_i, $i = 1, 2, \dots, n-t-1$. Finally, define a quasigroup operation $*$ on $\{0,1\}^{2^n}$ by (10), derived by the complete mapping F_{n-t}. So,

$$x * y = F_{a^{(n-t)},b^{(n-t)},c^{(n-t)}}(x \oplus_{2^n} y) \oplus_{2^n} y$$

for every $x, y \in \{0,1\}^{2^n}$.

Note that we need only $n-t$ iterations for getting the complete mapping F_{n-t} on the group $(\{0,1\}^{2^n}, \oplus_{2^n})$ and a small amount of memory for storing the bijection f. Hence, the complexity of our algorithm for construction of quasigroups of order 2^{2^n} is $\mathcal{O}(n)$.

The algebraic degree of a transformation g of the set $\{0,1\}^n$ is defined to be equal to the maximal degree of the polynomials in the vector valued Boolean function presentation of g. Namely, $g : \{0,1\}^n \to \{0,1\}^n$ can be presented by $g(b_1, b_2, \dots, b_n) = (g_1(b_1, b_2, \dots, b_n), \dots$ $\dots, g_n(b_1, b_2, \dots, b_n))$, where $g_i(b_1, b_2, \dots, b_n) : \{0,1\}^n \to \{0,1\}$ are Boolean functions. The algebraic normal forms of g_i can be considered as multivariate polynomials on $GF(2)$, and the degree of g_i is defined to be the degree of the corresponding polynomial.

Proposition 8 *The algebraic degree of the extended Feistel network $F_{a,b,c}$ created by a mapping f is equal to the algebraic degree of f.* $\square$

The cryptographic qualities of the quasigroups $(Q, *)$ obtained by extended Feistel networks as above depend on the starting bijections f and on the constants a, b, c that are used for their definitions. Let $(Q, *)$ be a quasigroup derived by a complete mapping $F_{a,b,c}$ created by a bijection f on the set $Q = \{0,1\}^{2^n}$. The following proposition is true.

Proposition 9 (1) *The quasigroup* $(Q, *)$ *is non-idempotent iff* $f(c) \neq b$ *or* $a \neq 0_{2^n}$ *(where* 0_{2^n} *denotes the word consisting of zeros only).*

(2) *The quasigroup* $(Q, *)$ *does not have neither left nor right unit.*

(3) *The quasigroup* $(Q, *)$ *is non-commutative and, much more, no different elements of* Q *commutes.*

(4) *Let* $\phi(x) = F_{a,b,c}(x) \oplus_{2^n} x$. *If* $a \neq 0_{2^n}$, *or* $f(c) \neq b$, *or* $\phi \circ F_{a,b,c}(x) \neq F_{a,b,c} \circ \phi(x)$ *for some* $x \neq 0_{2^n} \in Q$, *then the quasigroup* $(Q, *)$ *is non-associative.*

(5) *The identity* $y = ((y * \underbrace{x) * \dots) * x}_{l}$ *holds true in* $(Q, *)$ *iff* $F^l_{a,b,c} = I$, *where* I *is the identity mapping.*

(6) *The identity* $\underbrace{x * (\dots * (x}_{l} * y)) = y$ *holds true in* $(Q, *)$ *iff* $(I \oplus_{2k} F_{a,b,c})^l = I$.

(7) $< 0 > = < \{F^i_{a,b,c}(0) | \; i = 1, 2, \dots\} >$, *where* $< A >$ *denotes the subquasigroup generated by the subset* A *of* Q. $\square$

2.6 Linear transformation

The algorithm of NaSHA hash functions uses also the linear transformations explained as below.

Denote by $LinTr_{512}$ and by $LinTr_{256}$ the transformations of the sets $\{0,1\}^{2028}$ and $\{0,1\}^{1024}$ respectively, defined by

$$LinTr_{512}(S_1||S_2||\dots||S_{31}||S_{32}) = (S_7 \oplus S_{15} \oplus S_{25} \oplus S_{32})||S_1||S_2||\dots||S_{31},$$

$$LinTr_{256}(S_1||S_2||\dots||S_{15}||S_{16}) = (S_4 \oplus S_7 \oplus S_{10} \oplus S_{16})||S_1||S_2||\dots||S_{15},$$

where S_i are 64-bits words, $\oplus$ denotes the operation XOR on 64-bits words, and the operation $||$ denotes the concatenation of words.

Note that $LinTr_{512}$ is in fact the LFSR obtained from the primitive polynomial $x^{32} + x^{25} + x^{15} + x^7 + 1$ over the Galois field GF(2), applied in parallel 64 times, while $LinTr_{256}$ is obtained in the same way from the primitive polynomial $x^{16} + x^{10} + x^7 + x^4 + 1$. As a consequence we have the following.

Proposition 10 *$LinTr_{512}$ is a permutation of the set $\{0,1\}^{2028}$ and $LinTr_{256}$ is a permutation of the set $\{0,1\}^{1024}$.* $\square$

3 The NaSHA-(m,k,r) hash algorithm

We define a family of hash functions NaSHA-(m,k,r) by using the transformations $LinTr_{2^s}$ and $\mathcal{MT}$ as it is explained by the algorithm that follows. The parameters m, k and r denote the length of the output hash result (the message digest), the complexity of $\mathcal{MT}$ and the order 2^{2^r} of used quasigroup respectively, so k is a positive even integer and m and r are positive integers.

NaSHA-(m,k,r) hash algorithm

Input *A positive even integer k and positive integers m and r such that $m > 2^r$, and an input message M.*

Output *A hash value (message digest)* NaSHA-$(m,k,r)(M)$ *of m bits.*

Step 1 *Denote by n the smallest integer such that $m \leq 2^n$. (For example, n=8 for m=224 and n=9 for m=384.)*

Step 2 *Pad the message M, so that the length of the padded message M' is a multiple of 2^{n+1}, $|M'| = 2^{n+1}N$ for some N. Separate M' in N 2^{n+1}-bit blocks, $M' = M_1||M_2||\dots||M_N$, $|M_i| = 2^{n+1}$.*

Step 3 *Initialize the initial value H_0, which is a 2^{n+1}-bit word.*

Step 4 *The first message block M_1 and the initial value H_0 separate to $q = 2^{n-r+1}$ 2^r-bits words:*

$$M_1 = S_1||S_3||S_5||\dots||S_{2q-3}||S_{2q-1}||, \quad H_0 = S_2||S_4||S_6||\dots||S_{2q-2}||S_{2q}||,$$

($|S_i| = 2^r$) and form the word

$$S^{(0)} = S_1||S_2||S_3||S_4||\dots||S_{2q-3}||S_{2q-2}||S_{2q-1}||S_{2q}.$$

Step 5 *Choose leaders l_i as functions that depend on $S_1, S_2, S_3, \dots, S_{2q}$, a quasigroup $(\{0,1\}^{2^r}, *)$ and a suitable linear transformation $LinTr_{2^{n+2}}$.*

Step 6 *Compute the string of bits $S^{(N-1)}$ as follows.*

```
**FOR** $i = 1$ **TO** $N-1$ **DO**
    $A_1||A_2||A_3||\dots||A_{2q} \leftarrow \mathcal{MT}(LinTr^{2q}_{2^{n+2}}(S^{(i-1)}))$,
    $B_1||B_2||B_3||\dots||B_{q-1}||B_q \leftarrow M_{i+1}$,
    $S^{(i)} := B_1||A_2||B_2||A_4||\dots||B_{q-1}||A_{2q-2}||B_q||A_{2q}$,
**NEXT** $i$
```

Here, A_i and B_i are 2^r-bit variables, and $S^{(i)}$ are 2^{n+2}-bit variables.

Step 7 *Compute $\mathcal{MT}(LinTr^{2q}_{2^{n+2}}(S^{(N-1)})) := A_1||A_2||A_3||\dots||A_{2q}$. Then*

$$\text{NaSHA-}(m,k,r)(M) = A_4||A_8||\dots||A_{2q-4}||A_{2q} \ (\texttt{mod}\ 2^m).$$

We emphasize that some steps (e.g., Step 5) need more detailed elaborations in concrete implementations.

The NaSHA-(m,k,r) hash algorithm allows each bit of an input message M to influence almost all bits of the resulting hash value. To verify this let represent $S^{(i)}$ as

$$S^{(i)} = S^{(i)}_1||S^{(i)}_2||S^{(i)}_3||\dots||S^{(i)}_{2q-2}||S^{(i)}_{2q-1}||S^{(i)}_{2q}.$$

We have that every bit from the bit string $S^{(i)}$ influences all blocks $S^{(i+1)}_j$ with even subindexes ($j = 2,4,6,\dots,2q$) of the bit string $S^{(i+1)}$. Namely, by Step 6 we apply the transformations $LinTr^{2q}_{2^{n+2}}$ and $\mathcal{MT}$ on $S^{(i)}$. The linear transformation besides diffusion spread out the influence of bits. The $\mathcal{MT}$ transformation is composition of $\mathcal{A}_l$ and $\rho(\mathcal{RA}_l)$ transformations. Now, if b is a bit from a block $S^{(i)}_j$ of $S^{(i)}$, then all blocks of $\mathcal{A}_l(S^{(i)})$ from the $j+1$-th until $2q$-th are influenced by b. After that, all blocks of $\mathcal{MT}(\mathcal{A}_l(S^{(i)}))$ will be influenced by b. So we have the following theorem.

Theorem 2 *Every bit from the input message M influences all blocks of the hash value NaSHA-$(m,k,r)(M)$.*

Proof. By the above considerations we have that each bit of M influences all blocks with even subindexes of $S^{(N)}$. Since NaSHA-$(m,k,r)(M) = A_4||A_8||\dots||A_{2q-4}||A_{2q}$, where $A_1||A_2||A_3||\dots||A_{2q} =$

$(LinTr^{2q}_{2^{n+2}}(S^{(N)}))$, all blocks of NaSHA-$(m,k,r)(M)$ are influenced by each bit of M. $\square$

Much more than Theorem 2 is stating, the internal structure of the quasigroup operation and the addition modulo 2^r allows us to conclude that almost all bits of the hash value are influenced by each bit of the input message.

4 Implementation of NaSHA-$(m,2,6)$ hash functions for $m \in \{224, 256, 384, 512\}$

Here we give a complete implementation of NaSHA-(m,k,r) algorithm when $k=2,\ r=6$ and $m \in \{224, 256, 384, 512\}$. The used quasigroup of order $2^6 = 64$ is derived by extended Feistel networks.

4.1 Padding

The padding consists of the standard Merkle-Damgård strengthening [16]. Denote by M the bit input message of length $s = |M| < 2^{128}$.

1. Denote by q the smallest nonnegative integer such that

$$s + q + 1 \equiv 384 \ (mod\ 512)$$

for $m = 224$ and $m = 256$, and

$$s + q + 1 \equiv 896 \ (mod\ 1024)$$

for $m = 384$ and $m = 512$.

2. Let 0_q denote the binary word consisting of q zeros, and let b_s be the binary presentation of s by 128 bits.

3. Append to the message M the words 1, 0_q and b_s.

The padding of M is the message $M' = M||1||0_q||b_s$ and for $m = 256$ is a multiple of 512 and for $m = 512$ is a multiple od 1024.

Note that this implementation of NaSHA hash algorithm accept messages of length up to $2^{128} - 1$ bits.

f	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	8c	90	d9	c1	46	63	53	f1	61	32	15	3e	26	9a	97	2e
1	d8	a0	99	9e	c0	95	67	b7	6d	e0	f3	28	20	86	b6	ef
2	4b	31	b5	d2	13	39	6c	a5	03	3f	4d	34	f9	ec	8e	17
3	c5	25	3c	89	c9	2b	3a	c2	6e	c6	aa	91	49	18	93	de
4	0d	6f	65	af	92	a7	f6	a6	40	b9	ed	b0	c3	d7	7d	7c
5	54	59	df	2f	da	a4	05	94	9b	72	01	74	a9	f7	81	e9
6	1f	b3	eb	cf	8	47	52	36	bc	16	29	76	12	fa	9c	8a
7	5b	a8	43	d1	79	85	42	82	c7	a1	78	4f	e2	35	ea	ad
8	dc	0e	d3	2d	6a	5a	44	ab	c8	e5	37	0a	6b	51	e3	14
9	cd	56	4a	d6	08	83	bb	33	e1	30	4e	24	5e	b4	00	48
a	5f	22	0b	50	3d	80	1a	bf	cc	ff	64	87	1b	c4	07	f8
b	0c	d4	ac	02	10	84	7e	69	70	60	55	2a	21	57	23	66
c	62	73	cb	41	58	71	77	1c	7b	8f	9f	9d	a3	b1	7f	5d
d	f4	06	ae	d5	e6	3b	ba	Fe	96	e7	0f	45	2c	f0	fc	bd
e	e4	98	fb	ca	11	f5	dd	7a	5c	fd	ce	88	d0	68	8d	4c
f	be	04	38	1d	1e	f2	27	19	b2	75	a2	ee	db	b8	09	8b

Table 1: The starting bijection $f = f(m||n)$

4.2 Starting bijection

As starting bijection $f : \mathbb{Z}_2^8 \to \mathbb{Z}_2^8$ for creating extended Feistel network we use improved AES S-box with the APA structure from Cui and Cao [3], given on Table 1 in hexadecimal notation.

4.3 Quasigroup operation via extended Feistel network

From the starting bijection f we define three extended Feistel networks $F_{a_1,b_1,c_1}, F_{a_2,b_2,c_2}, F_{a_3,b_3,c_3} : \mathbb{Z}_2^{16} \to \mathbb{Z}_2^{16}$ by

$$F_{a_i,b_i,c_i}(l_8||r_8) = (r_8 \oplus a_i)||(l_8 \oplus b_i \oplus f(r_8 \oplus c_i)),$$

where l_8 and r_8 are 8-bit variables, and $a_i,\ b_i,\ c_i$ are 8-bit words that are defined before each application of $\mathcal{MT}$, as it is explained in Subsection 4.5. Denote by f' the bijection $F_{a_1,b_1,c_1} \circ F_{a_2,b_2,c_2} \circ F_{a_3,b_3,c_3} : \mathbb{Z}_2^{16} \to \mathbb{Z}_2^{16}$.

By using the bijection f' we define a quasigroup operation on $\mathbb{Z}_2^{64}$ that will be used for the additive string transformation $\mathcal{A}$ as follows. Create the Feistel networks $F_{\alpha_1,\beta_1,\gamma_1} : \mathbb{Z}_2^{32} \to \mathbb{Z}_2^{32}$ and $F_{A_1,B_1,C_1} : \mathbb{Z}_2^{64} \to \mathbb{Z}_2^{64}$ by

$$F_{\alpha_1,\beta_1,\gamma_1}(l_{16}||r_{16}) = (r_{16} \oplus \alpha_1)||(l_{16} \oplus \beta_1 \oplus f'(r_{16} \oplus \gamma_1)),$$

$$F_{A_1,B_1,C_1}(l_{32}||r_{32}) = (r_{32} \oplus A_1)||(l_{32} \oplus B_1 \oplus F_{\alpha_1,\beta_1,\gamma_1}(r_{32} \oplus C_1)),$$

where l_{16}, r_{16} are 16-bit variables, $\alpha_1, \beta_1, \gamma_1$ are 16-bit words, l_{32}, r_{32} are 32-bit variables and A_1, B_1, C_1 are 32-bit words. The constant words will be defined in Subsection 4.5. The function F_{A_1,B_1,C_1} is a complete mapping in the group $(\mathbb{Z}_2^{64}, \oplus)$, and then the operation $*_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_1,\beta_1,\gamma_1,A_1,B_1,C_1}$ defined by

$$x *_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_1,\beta_1,\gamma_1,A_1,B_1,C_1} y = F_{A_1,B_1,C_1}(x \oplus y) \oplus y$$

is a quasigroup operation in $\mathbb{Z}_2^{64}$.

By using the bijection f' we define also a quasigroup operation in $\mathbb{Z}_2^{64}$ that will be used for the reverse additive string transformation $\mathcal{RA}$ as follows. Create the Feistel networks $F_{\alpha_2,\beta_2,\gamma_2} : \mathbb{Z}_2^{32} \to \mathbb{Z}_2^{32}$ and $F_{A_2,B_2,C_2} : \mathbb{Z}_2^{64} \to \mathbb{Z}_2^{64}$ by

$$F_{\alpha_2,\beta_2,\gamma_2}(l_{16}||r_{16}) = (r_{16} \oplus \alpha_2)||(l_{16} \oplus \beta_2 \oplus f'(r_{16} \oplus \gamma_2)),$$

$$F_{A_2,B_2,C_2}(l_{32}||r_{32}) = (r_{32} \oplus A_2)||(l_{32} \oplus B_2 \oplus F_{\alpha_2,\beta_2,\gamma_2}(r_{32} \oplus C_2)),$$

where l_{16}, r_{16} are 16-bit variables, $\alpha_2, \beta_2, \gamma_2$ are 16-bit words, l_{32}, r_{32} are 32-bit variables and A_2, B_2, C_2 are 32-bit words. The constant words will be defined in Subsection 4.5. The function F_{A_2,B_2,C_2} is a complete mapping in the group $(\mathbb{Z}_2^{64}, \oplus)$, and then the operation $*_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_2,\beta_2,\gamma_2,A_2,B_2,C_2}$ defined by

$$x *_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_2,\beta_2,\gamma_2,A_2,B_2,C_2} y = F_{A_2,B_2,C_2}(x \oplus y) \oplus y$$

is a quasigroup operation in $\mathbb{Z}_2^{64}$.

In such a way we achieve for each application of $\mathcal{MT}$ to use different quasigroup operations $*_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_1,\beta_1,\gamma_1,A_1,B_1,C_1}$ for the transformation $\mathcal{A}$ and $*_{a_1,b_1,c_1,a_2,b_2,c_2,a_3,b_3,c_3,\alpha_2,\beta_2,\gamma_2,A_2,B_2,C_2}$ for the transformation $\mathcal{RA}$.

4.4 Chaining initial vectors

The definition of NaSHA-(m, k, r) hash function includes one initial string H_0. The initial strings we are using are the following, represented in hexadecimal as concatenation of 64-bit chunks. (They were made accidently by the authors.)

1. $m = 224, H_0 =$
6a09e667f3bcc908, cbbb9d5dc1059ed8, bb67ae8584caa73b, 629a292a367cd507,

3c6ef372fe94f82b, 9159015a3070dd17, a54ff53a5f1d36f1, 152fecd8f70e5939

2. $m = 256, H_0 =$
510e527fade682d1, 67332667ffc00b31, 9b05688c2b3e6c1f, 8eb44a8768581511, 1f83d9abfb41bd6b, db0c2e0d64f98fa7, 5be0cd19137e2179, 47b5481dbefa4fa4

3. $m = 384, H_0 =$
6a09e667f3bcc908, cbbb9d5dc1059ed8, bb67ae8584caa73b, 629a292a367cd507, 3c6ef372fe94f82b, 9159015a3070dd17, a54ff53a5f1d36f1, 152fecd8f70e5939, 510e527fade682d1, 67332667ffc00b31, 9b05688c2b3e6c1f, 8eb44a8768581511, 1f83d9abfb41bd6b, db0c2e0d64f98fa7, 5be0cd19137e2179, 47b5481dbefa4fa4

4. $m = 512, H_0 =$
2dd8a09a3c4e3efb, e07688dc6f166b73, 061a77a060948dcd, 0c34aa2a315e01d5, 8a47ea1880559ce6, c785f4364a0b98f4, 9f22535b264607a8, 53a8c8ca56e1288c, 2547d84e9ccde59d, 3c1563a9317c57a1, 9486eb50c7d8037f, 77341edad21e9a40, c0f905d741c9cb74, d648813e45121dbb, ad0d1e41a985e51e, 4cf768fc7df11b00

4.5 Definition of the leaders and constants

Before every computation $\mathcal{MT}(S_1||S_2||S_3||\ldots||S_{2q-1}||S_{2q})$, where S_i are 64-bit words, we define the 64-bit leaders l_1 of $\mathcal{RA}$ and l_2 of $\mathcal{A}$, the 8-bit words $a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c_3$, the 16-bit words $\alpha_1, \beta_1, \gamma_1, \alpha_2, \beta_2$, γ_2 and the 32-bit words $A_1, B_1, C_1, A_2, B_2, C_2$, as follows:

$$l_1 = S_1 + S_2, \quad l_2 = S_3 + S_4,$$

$$a_1||b_1||c_1||a_2||b_2||c_2||a_3||b_3 = S_5 + S_6, \quad c_3 = a_1$$
$$\alpha_1||\beta_1||\gamma_1||\alpha_2 = S_7 + S_8,$$
$$\beta_2||\gamma_2 = (S_9 + S_{10})(\texttt{mod } 2^{32}),$$
$$A_1||B_1 = S_{11} + S_{12}, \quad C_1||A_2 = S_{13} + S_{14}, \quad B_2||C_2 = S_{15} + S_{16}.$$

Here, the addition $+$ is modulo 2^{64}.

5 Design rationales

THE CHOICE OF THE STARTING BIJECTION. As NaSHA starting bijection we wanted to use some publicly known function in order to prevent suspicious of possible "trap door" in the implementation. We considered several possibilities: AES S-box [1], improved AES S-box from Liu and all [12] and improved AES S-box with the APA structure from Cui and Cao [3]. All three runners have some proc and cons. The AES S-box is the most famous and the most investigated S-box in cryptology, with good differential and linear resistance and high algebraic degree. But it has simple algebraic structure with only 9 terms. The improved AES S-boxes has also good differential resistance with differential 4-uniformity and good linear resistance. They have the same algebraic degree as AES S-box, but they have much bigger algebraic complexity of 255 terms for the first, and 253 terms for the second, S-box. Their inverse S-box has high algebraic complexity of 255 terms as AES inverse S-box. But both are not enough studied from other authors. Our winner f is the third solution, because of its algebraic complexity and because it is a little bit more studied than the second solution. The function f also satisfies the condition $f(0) \neq 0$ that is needed our extended Feistel network to derive a non-idempotent and a non-associative quasigroup.

THE CHOICE OF THE LINEAR TRANSFORMATION. The linear transformation is used for obtaining suitable diffusion of the input 64-bit words. We use LFSRs for obtaining linear transformation that is a bijection and that can be easily computed. For that aim we use primitive polynomials over the Galois field GF(2), from the list [17]. The degree of the primitive polynomial for 224 and 256 hash need to be 16, and 32 for 384 and 512 hash. Since the algorithm applies the linear transformation 16 (i.e, 32) times, we take the primitive polynomials with 5 terms. Any other polynomial that fulfils these requirements is a good choice too.

THE CHOICE OF THE QUASIGROUP TRANSFORMATIONS. By our experience and some theoretical results we have obtained, quasigroup transformations are good nonlinear building blocks for designing different cryptographic primitives (e.g., the stream cipher Edon80 [6] is still unbroken). We use quasigroups of huge order 2^{64} and they are defined by extended Feistel networks, that are generalizations of

Feistel networks, defined by us in [15]. Our algorithm can also be implemented by quasigroups of order 2^{32}, 2^{128}, 2^{256} etc, but we found that the choice of order 2^{64} is optimal for obtaining tradeoff between security and speed.

THE CHOICE OF THE EXTENDED FEISTEL NETWORKS. It is not easy to define a workable quasigroup of huge order, like 2^{64}, having good cryptographic properties. Our choice were extended Feistel networks since they produce quasigroups of good cryptographic qualities, and allows to insert tunable parameters in their definition. We used that feature to obtain different quasigroups for every iteration of the compression function and, much more, the used quasigroups are functions of the processed message block.

We are using 9 8-bit words $a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c3$, 6 16-bit words $\alpha_1, \beta_1, \gamma_1, \alpha_2, \beta_2$, γ_2 and 6 32-bit words $A_1, B_1, C_1, A_2, B_2, C_2$ in every iteration of the compression function and pass them to extended Feistel networks. The way of their definition was leaded by the idea all bits of the processed input block to be included.

THE CHOICE OF THE COMPOSITE MAPPINGS IN THE MAIN TRANSFORMATION AND THE TUNABLE SECURITY PARAMETER k. In general, the main transformation $\mathcal{MT}$ can be defined as any composition of the transformations $\mathcal{A}$ and $\mathcal{RA}$. Having in mind the properties of the extended Feistel networks, where the starting bijection influences mostly the right half of the output result, we are using the transformation $\mathcal{RA}$ after rotating left for 32 bits the obtained 64-bit words from $\mathcal{A}$. In such a way, a homogeneous spreading of the starting bijection is obtained. Also, by the transformation $\mathcal{A}$ the influence of the input bits are spreading only in the right part of the output, that is why $\mathcal{RA}$ is defined in a reverse way of $\mathcal{A}$. At the end, we obtain every bit of an input block to influence almost all bits of the output blocks of $\mathcal{RA} \circ \mathcal{A}$.

The tunable security parameter of the NaSHA hash algorithm is the complexity k of the main transformation $\mathcal{MT}$, since we define $\mathcal{MT}$ as composition of k mappings of kind $\mathcal{RA}$ and $\mathcal{A}$, applied consecutively. The choice of higher values of k will give stronger security, but lower speed. Our choice, recommendation and low bound is $k = 2$ (there is no upper bound). We believe that the cryptanalysis will became practical if $k = 1$, that is when $\mathcal{MT} = \mathcal{A}$ or $\mathcal{MT} = \mathcal{RA}$.

6 Preliminary security analysis

NaSHA family of cryptographic hash function use Merkle-Damgård domain extender with standard Merkle-Damgård strengthening. It has incorporated also the wide-pipe design of Lucks [13, 14] and Coron's [2] suggestions. In every iterative step of the compression function, we use $2n$-bit message blocks and $2n$-bit chaining variable, so the strings of length $4n$ bits are mapped to strings of length $4n$ bits and then only $2n$ bits are kept for the next iterative step. And, the most important, *the length of any chaining variable is at least two times wider than the final digest value.* For the same reasons D. Gligorovski [7] stated, by this kind of design we gain resistance to some generic attacks like: Joux multicollision attack [9], length extension attack, Dean fixed point attack [5], Kelsey and Schneier long message 2^{nd} preimage attack [10], Kelsey and Kohno herding attack [11] and 2^{nd} collision attack.

6.1 Collision, preimage and 2^{nd} preimage resistance of NaSHA

The collision resistance of NaSHA-(m, k, r) depends mainly of Step 5 of the NaSHA-(m, k, r) hash algorithm. Namely, by suitable definition of the quasigroup operation one can get higher or lower collision security. For obtaining suitable security, we introduce several security parameters $a_1, b_1, c_1, a_2, b_2, c_2, a_3, b_3, c3,\ \alpha_1, \beta_1, \gamma_1, \alpha_2, \beta_2,\ \gamma_2 A_1, B_1, C_1, A_2, B_2,$ $C_2,\ l_1,\ l_2$. The security analysis of the $\mathcal{MT}$ reduces to solving system of equations, each equation of kind $x * (x + a) = b$ or $(x + a) * x = b$. Here $*$ is a quasigroup operation and $+$ is addition modulo 2^{64}. As well as we know, there is no other way of solving equations of above kind, except checking all the possible values of x (i.e., brute force attack). We show that these attacks are quite inefficient. So, the best collision, preimage and second preimage attacks are the generic attacks. All the proofs are given in part **2.B.4.** of this submission.

6.2 Avalanche effect

We tested the avalanche propagation of one bit differences in compression function of NaSHA-$(m, 2, 6)$, where $m \in \{224, 256, 384, 512\}$, in two cases: when the initial message consists of all zeros and when the initial message is randomly generated. We present in Tables 2 and 3 the obtained results for messages of length 8, 80, 800, 8000 and 80000 bits, where minimum, average and maximum different bits and standard deviation are given. Table 2 is for initial messages consisting of all zeros and Table 3 is for randomly generated initial message. One can see that in every case the Hamming distance is around $m/2$, or one bit difference of input bits produces about 50% different output bits, as it would be expected in theoretical models of ideal random functions.

n	8 bits	80 bits	800 bits	8000 bits	80000 bits
224	$min = 42\%$ $avg = 50.06\%$ $max = 56\%$ $sd = 4.44$	$min = 41\%$ $avg = 49.86\%$ $max = 57\%$ $sd = 3.48$	$min = 41\%$ $avg = 50.21\%$ $max = 60\%$ $sd = 3.39$	$min = 38\%$ $avg = 49.97\%$ $max = 63\%$ $sd = 3.40$	$min = 35\%$ $avg = 50.02\%$ $max = 63\%$ $sd = 3.41$
256	$min = 45\%$ $avg = 49.12\%$ $max = 55\%$ $sd = 2.91$	$min = 43\%$ $avg = 50.88\%$ $max = 58\%$ $sd = 3.35$	$min = 40\%$ $avg = 50.11\%$ $max = 58\%$ $sd = 3.20$	$min = 37\%$ $avg = 49.96\%$ $max = 60\%$ $sd = 3.14$	$min = 35\%$ $avg = 50.00\%$ $max = 62\%$ $sd = 3.16$
384	$min = 46\%$ $avg = 49.32\%$ $max = 53\%$ $sd = 1.96$	$min = 45\%$ $avg = 49.86\%$ $max = 54\%$ $sd = 2.49$	$min = 40\%$ $avg = 50.10\%$ $max = 59\%$ $sd = 2.52$	$min = 40\%$ $avg = 50.04\%$ $max = 59\%$ $sd = 2.60$	$min = 39\%$ $avg = 50.00\%$ $max = 60\%$ $sd = 2.61$
512	$min = 47\%$ $avg = 50.12\%$ $max = 51\%$ $sd = 1.41$	$min = 45\%$ $avg = 50.01\%$ $max = 55\%$ $sd = 2.11$	$min = 42\%$ $avg = 50.04\%$ $max = 58\%$ $sd = 2.35$	$min = 41\%$ $avg = 49.99\%$ $max = 58\%$ $sd = 2.25$	$min = 41\%$ $avg = 50.00\%$ $max = 58\%$ $sd = 2.25$

Table 2: Avalanche effect of input message with all zeros

n	8 bits	80 bits	800 bits	8000 bits	80000 bits
224	$min = 49\%$	$min = 41\%$	$min = 41\%$	$min = 37\%$	$min = 35\%$
	$avg = 52.68\%$	$avg = 50.38\%$	$avg = 50.14\%$	$avg = 49.99\%$	$avg = 50.00\%$
	$max = 56\%$	$max = 61\%$	$max = 62\%$	$max = 61\%$	$max = 63\%$
	$sd = 2.27$	$sd = 3.89$	$sd = 3.40$	$sd = 3.38$	$sd = 3.42$
256	$min = 42\%$	$min = 41\%$	$min = 41\%$	$min = 38\%$	$min = 36\%$
	$avg = 48.73\%$	$avg = 50.72\%$	$avg = 50.06\%$	$avg = 50.01\%$	$avg = 50.01\%$
	$max = 53\%$	$max = 60\%$	$max = 58\%$	$max = 61\%$	$max = 62\%$
	$sd = 3.80$	$sd = 3.46$	$sd = 3.14$	$sd = 3.18$	$sd = 3.18$
384	$min = 47\%$	$min = 43\%$	$min = 42\%$	$min = 40\%$	$min = 39\%$
	$avg = 50.29\%$	$avg = 49.95\%$	$avg = 49.87\%$	$avg = 49.98\%$	$avg = 50.00\%$
	$max = 54\%$	$max = 54\%$	$max = 57\%$	$max = 58\%$	$max = 59\%$
	$sd = 2.28$	$sd = 2.38$	$sd = 2.60$	$sd = 2.63$	$sd = 2.61$
512	$min = 49\%$	$min = 47\%$	$min = 43\%$	$min = 41\%$	$min = 40\%$
	$avg = 51.20\%$	$avg = 50.32\%$	$avg = 50.00\%$	$avg = 50.05\%$	$avg = 50.02\%$
	$max = 53\%$	$max = 55\%$	$max = 57\%$	$max = 58\%$	$max = 59\%$
	$sd = 1.28$	$sd = 1.95$	$sd = 2.26$	$sd = 2.25$	$sd = 2.26$

Table 3: Avalanche effect of a randomly generated input message

References

[1] J. Daemen and V. Rijmen, *The Design of Rindael: AES - The Advanced Encryption Standard*, Springer-Verlag, Berlin, 2002

[2] J.-S. Coron, Y. Dodis, C. Malinaud and P. Puniya, *Merkle-Damgård revisited: How to construct a hash function*, CRYPTO 2005, LNCS **3621**

[3] L. Cui and Y. Cao, *A new S-box structure named Affine-Power-Affine*, International Journal of Innovative Computing, Information and Control **3(3)**, (2007), pp. 751–759

[4] I. B. Damgård, *A Design Principle for Hash Functions*, Advances in Cryptology - CRYPTO 1989, LNCS **435** (1990), pp.416–427

[5] R. D. Dean, *Formal Aspects of Mobile Code Security*,Ph.D. dissertation, Princeton University, 1999

[6] D. Gligoroski, S. Markovski, L. Kocarev, and M. Gusev, *Edon80 Hardware Synchronous stream cipher*, SKEW 2005 - Symmetric Key Encryption Workshop, May 2005, Aarhus Denmark

[7] D. Gligoroski and S. Knapskog *Edon-R(224, 256, 384, 512)-an Efficient Implementation of Edon–$\mathcal{R}$ Family of Cryptographic Hash Functions*, ecrypt archive 2007/154

[8] H. Feistel, *Cryptography and computer privacy*, Scientific American, **228** (1973), No. 5, 15–23

[9] A. Joux, *Multi-collisions in Iterated Hash Functions. Applications to Cascades Constructions*, Advances in Cryptology - CRYPTO 2004, LNCS **3152** (2004), pp 306–316

[10] J. Kelsey and B. Schneier, *Second preimages on n-bit hash functions for much less than 2n work*, Advances in Cryptology - EUROCRYPT 2005, LNCS **3494** (2005), pp 474–490

[11] J. Kelsey and T. Kohno, *Herding Hash Functions and the Nostradamus Attack*, Advances in Cryptology - EUROCRYPT 2006, LNCS **4004** (2006), pp. 183–200

[12] J. Liu, B. Wei, X. Cheng and X. Wang, *Cryptanalysis of Rijndael S-box and improvement*, Applied Mathematics and Computation **170 (2)**, (2005), pp.958–975

[13] S. Lucks, *Design Principles for Iterated Hash Functions*, Cryptology ePrint Archive, Report 2004/253

[14] S. Lucks, *A Failure-Friendly Design Principle for Hash Functions*, ASIACRYPT 2005, LNCS **3788** (2005), pp. 474–494

[15] S. Markovski and A. Mileva, *Generating huge quasigroups from small non-linear bijections via extended Feistel network*

[16] R. C. Merkle, *One way hash functions and DES*, Advances in Cryptology - CRYPTO 1989, LNCS **435** (1990), pp.428–446

[17] J. Rajski and J. Tyszer, *Primitive Polynomials Over GF(2) of Degree up to 660 with Uniformly Distributed Coefficients*, Journal of Electronic Testing: Theory and Applications, Volume **19**, Issue 6 (2003), pp 645 – 657

[18] A. Sade, *Quasigroups automorphes par le groupe cyclique*, Canadian Journal of Mathematics **9**(1957), pp. 321–335