Classification of the SHA-3 Candidates

Ewan Fleischmann[1], Christian Forler[1,2], and Michael Gorski[1]

[1] Bauhaus-University Weimar {Ewan.Fleischmann, Michael.Gorski}@uni-weimar.de
[2] Sirrix AG security technologies c.forler@sirrix.com

Version 0.90
April 19, 2009

Abstract. In this note we give an overview on the current state of the SHA-3 candidates. First, we classify all publicly known candidates and, second, we outline and summarize the performance data as given in the candidates documentation for 64-bit and 32-bit implementations. We define performance classes and classify the hash algorithms. Note, that this article will be updated as soon as new candidates arrive or new cryptanalytic results get published. Comments to the authors of this article are welcome.

Keywords: hash function, SHA-3, classification.

1 Introduction

The design of secure and practical hash functions is of great interest since most practical hash functions, like MD5 [82], SHA-0 [78] or SHA-1 [76] have been broken. Due to the SHA-3 competition [75], many new proposals for hash function primitives have been submitted to become the new SHA-3 algorithm.

This article is organized as follows: In Section 2 we define criteria that we will use to classify the SHA-3 candidate algorithms. In Section 3 we give an overview of the software performance claimed by the algorithm's authors.

2 Classification of the SHA-3 Candidates

We have defined in the following some attributes including characteristics that are used in our classification. Tables 1-2 show all SHA-3. The attribute characteristics "X" means that the SHA-3 candidate has the attribute and"-" that the SHA-3 candidate does not have it. The meaning of the other characteristics can be found in the following attribute description.

Feistel Network (FN)[88] A Feistel network is a general method of transforming any function (usually called an F-function) into a permutation. An F-function is always non-linear and almost always irreversible. The Feistel Network was invented by Horst Feistel. The FN attribut can have two characteristics.

Balanced Feistel Network (B)

A compression function is called a balanced feistel network, when

1. the internal state is divided into a left and right part of equal size n.
2. a message depended, nonlinear function F maps those parts to two output parts of the same length.

Unbalanced feistel network (U)

A compression function is called an unbalanced feistel network is based on a feistel network where the internal state is divided into more resp. less then two parts or into two parts of unequal size.

Wide Pipe design (WP) [58]

The internal state, i.e. chaining value, of the hash function is larger than the message digest.

Key Schedule (KEY)

The hash function has an explicit key schedule or a message expansion algorithm.

MDS Matrix (MDS) [85]

One or more Maximum Distance Separable (MDS) matrices are used as a building block of the compression function. A MDS matrix has strong diffusion properties that can be exploited in certain cryptographic primitives. The characteristics is the size of the MDS Matrix in bytes. *For exmaple, the characteristics "4×4" means that a "4×4" byte MDS matrix is used as a bulding block.*

Output Transformation (OUT)

Is a function with the"final" chaining value as input and the message digest as output. Trivial output transformation such as the identity or truncation does not count at all.

S-box (SBOX)

The hash function uses one or more substitution boxes. In general a S-box is a non linear function that maps m input bits to n output bits. Usually, a S-box is implemented as lookup table. The characteristics is the S-bix size in bits.

Feedback Shift Register (FSR)

The compression functions is/uses a (N)LFSRs. The input bits of a (non-)linear feedback shift register ((N)LFSR) are computed via a (non-)linear function from the previous state.

Addition Rotation XOR (ARX)

The compression function contains addition, rotation and XOR operations with the purpose to destroy linearity.

Boolean operations (BOOL)

The compression function contains basic operations of the boolean algebra. Characteristics are the boolean operation of the compression function.

Collision Attack

The best known collision attack that is better than the birthday attack. The characteristic E means that a practical example for a collision exists.

(Second) Preimage Attack

The best known (2nd) preimage attack that is better than then long second preimage attack [46]. The characteristic E means that a practical example for a (2nd) preimage exists.

Hash algorithm	**FN**	**WP**	**KEY**	**MDS**	**OUT**	**SBOX**	**FSR**	**ARX**	**BOOL**	**COL**	**PRE**
*Abacus[89]	-	X	-	4×4	X	8×8	X	-	-	2^{172} [99]	2^{172} [70]
ARIRANG [19]	U	X	X	$4 \times 4,\ 8 \times 8$	-	8×8	-	-	-	-	-
AURORA [44]	-	-	X	4×4	X	8×8	-	-	-	$2^{234.51}/2^{229.6}$[23]	$2^{291}/2^{31.5}$[23]
BLAKE [4]	U	-	X	-	-	-	-	X -	-	-	-
Blender [13]	-	X	-	-	-	-	-	X	-	$10 * 2^{n/4}$[51]	$10 * 2^{n/4}$[51]
BMW [30]	-	X	X	-	-	-	-	X	-	[91]†	-
*Boole [84]	-	-	-	-	X	-	X	-	$\vee$	2^{34} [25]	$2^{\frac{9n}{16}}$ [74]
Cheetah [29]	-	-	X	$4 \times 4,\ 8 \times 8$	-	8×8	-	-	-	-	-
Chi [35]	U	X	X	-	-	4×3	-	-	$\vee, \neg$	-	-
CRUNCH [32]	U	-	X	-	-	8×1016	-	-	-	-	-
CubeHash8/1 [8]	-	-	-	-	-	-	-	X	-	-	2^{509} [5]
*DHC [101]	-	-	X	-	-	8×8	-	-	-	2^{9} [55]	2^{9} [49]
DynamicSHA [103]	U	-	X	-	-	-	-	-	$\wedge, \vee, \neg$	2^{114}[41]	-
DynamicSHA2 [104]	U	-	X	-	-	-	-	X	$\wedge, \vee, \neg$	-	-
ECHO [7]	-	X	-	4×4	-	8×8	-	-	-	-	-
ECOH [15]	-	-	X	-	-	-	-	-	-	-	-
Edon-R [31]	-	X	X	-	-	-	-	X	-	-	$2^{2n/3}, 2^{2n/3}$ [50]
EnRUPT [79]	-	(X)	-	-	-	-	-	X	-	E, 2^{47} [39]	$2^{480}/2^{480}$[47]
Essence [61]	-	-	-	-	-	-	X	-	-	-	-
FSB [3]	-	X	-	-	X	-	-	-	-	-	-
Fugue [33]	-	X	-	4×4	X	8×8	-	-	-	-	-
Grøstl [28]	-	X	-	8×8	X	8×8	-	-	-	-	-
Hamsi [54]	-	-	X	-	-	4×4	-	-	-	-	-
JH [102]	B	X	-	1.5×1.5	-	4×4	-		-	-	$2^{510.3}/2^{510.3}$ [26]
Keccak [10]	-	X	-	-	-	-	-	-	$\wedge, \neg$	-	-
*Khichidi-1 [97]	-	-	X	-	-	-	X	-	-	$1/2^{33}$ [90]	E [68], $1/2^{33}$ [90]
LANE [40]	-	-	X	4×4	X	8×8	-	-	-	-	-
Lesamnta [37]	U	-	X	$2 \times 2,\ 4 \times 4$	X	8×8	-	-	-	-	-
Luffa [16]	-	-	-	-	X	4×4	-	-	-	-	-
Lux [72]	-	X	-	$4 \times 4\ ,8 \times 8$	X	8×8	-	-	-	-	-
MCSSHA-3 [62]	-	-	-	-	-	-	X	-	-	$2^{3n/8}$[6]	$2^{3n/4}$ [6]
MD6 [83]	-	X	-	-	-	-	X	-	$\wedge$	-	-
*MeshHash [22]	-	-	-	-	X	8×8	-	-	-	-	$2^{323.2}/2^{n/2}$ [92]

* Submitter has conceded that the algorithm is broken.
† Free-start near-collision.

Table 1. Attribute list of the first round candidates (A-M).

Hash algorithm	FN	WP	KEY	MDS	OUT	SBOX	FSR	ARX	BOOL	COL	PRE
NaSHA [60]	B	-	-	-	-	8×8	X	-	-	2^{128} [57]	$2^{n/2}$ [73]‡
SANDstorm [94]	-	-	X	-	-	8×8	-	-	$\wedge, \neg$	-	-
Sarmal [96]	U	-	-	8×8	-	8×8	-	-	-	$2^{n/3}/2^{n/3}$ [66]†	$2^{384}/2^{128}$ [71]
Sgàil [64]	-	X	X	$8 \times 8,\ 16 \times 16$	-	8×8	-	X	-	E [63]	-
Shabal [14]	-	-	X	-	-	-	X	-	$\wedge, \neg$	-	-
*SHAMATA [2]	B	X	X	4×4	-	8×8	-	-	-	$2^{40}/2^{29}$ [43]	$2^{451.7}/2^{452.7}$ [38]
SHAvite-3 [11]	B	-	X	4×4	-	8×8	X	-	-	-	-
SIMD [56]	U	X	X	TRSC+	-	-	-	-	$\wedge, \neg, \vee$	-	-
Skein [24]	B	(X)◇	X	-	X	-	-	X	-	-	-
Spectral Hash [86]	-	-	-	-	X	8×8	-	-	-	E [36]	-
*StreamHash [95]	-	-	-	-	-	8×8	-	-	-	E [12]	$\frac{n}{2} * 2^{n/2}$ [48]
SWIFFTX [1]	-	-	-	-	-	8×8	-	-	-	-	-
*Tangle [81]	-	(X)	X	-	-	8×8	-	X	$\wedge, \neg, \vee$	E, 2^{19} [93]	-
TIB3 [67]	U	-	X	-	-	3×3	-	-	-	-	-
Twister [21]	-	X	-	8×8	X	8×8	-	-	-	2^{252} [65]	$2^{448}/2^{64}$ [65]
Vortex [53]	-	-	-	4×4	X	8×8	-	-	-	$2^{122.5}/2^{122.5}$ [52]	$2^{3n/4}/2^{n/4}$ [52]
*WAMM [98]	-	X	-	-	X	8×8	-	-	-	E [100]	-
*Waterfall [34]	-	X	-	-	X	8×8	X	-	-	2^{70} [27]	-

◇ Specified for either narrow or wide pipe design.
+ Truncated Reed-Solomone codes.
* Submitter has conceded that the algorithm is broken.
† Collision with salt.
‡ Free-start preimage.

Table 2. Attribute list of the first round candidates (N-Z).

Hash algorithm	FN	WP	KEY	MDS	OUT	SBOX	FSR	ARX	BOOL	COL	PRE
Maraca [45]	-	X	X	-	-	-	-	-	-	$2^{237}/2^{230.5}$ [18]	E [42]
NKS2D [80]	-	-	-	-	-	-	-	-	-	E [17, 20]	-
Ponic [87]	-	X	-	-	X	8×8	X	-	-	-	$2^{265}/2^{256}$ [69]

Table 3. Attribute list of the SHA-3 candidates that are not accepted for the first round.

3 Software Speed of the SHA-3 Candidates

In this section we give an overview of the claimed software performance of the public known SHA-3 candidates. We compare each candidate for their 32 and 64 bit performance. Therefore, we define five speed classes, which are listed in Table 4.
Tables 5-8 compare the SHA-3 candidates and their speed classes. As a reference algorithm we add SHA-256/ 512 [77]. Since each SHA-2 version is in class C for the 32 bit performance and in class B for the 64 bit performance, we think that this can be seen as a benchmark for all algorithms submitted. Nevertheless, there is a tradeoff between speed and security.

Speed	**Classification**
$x < \frac{1}{2}$ SHA-2	AA
$\frac{1}{2}$ SHA-2 $\leq x < \frac{3}{4}$ SHA-2	A
$\frac{3}{4}$ SHA-2 $\leq x <$ SHA-2	B
SHA-2 $\leq x < \frac{5}{4}$ SHA-2	C
$\frac{5}{4}$ SHA-2 $\leq x \leq 2$ SHA-2	D
$x > 2$ SHA-2	E

Table 4. Speed classification table.

One can easily design a hash function with a high level of security which is very slow and therefore may be useless in practice. For practical interest algorithms that are in speed class D or E will have a disadvantage for practical purpose, but they could possibly face a strong design. On the other side if an algorithm is very fast, i.e. in speed class AA, this could be a hint that the security margin is not chosen so high. Recent breaks of very fast hash functions, i.e. EnRUPT [79] or Boole [84], have verified this conjecture.

4 Acknowledgements

The authors wish to thank Jason Martin, Danilo Gligoroski, Vlastimil Klima, Peter Schmidt-Nielsen, Shiho Moriai, Florian Mendel, Joan Daemen, Gilles Van Assche, Taizo Shirai, Orr Dunkelman, David Bauer, and Stefan Lucks for there useful comments and remarks.

Hash algorithm	Performance 32 Bit		Performance 64 Bit	
	cpb	class	cpb	class
SHA-256 [77]	29.3	C	20.1	C
SHA-512 [77]	55.2	C	13.1	C
*Abacus-256 [89]	37.7	D	37.7	D
*Abacus-512 [89]	68	C	68	E
ARIRANG-256 [19]	20	A	55.3	E
ARIRANG-512 [19]	14.9	AA	11.2	B
AURORA-256 [44]	24.3	B	15.4	B
AURORA-512 [44]	46.9	B	27.4	E
BLAKE-32[4]	28.3	B	16.7	B
BLAKE-64[4]	61.7	C	12.3	B
Blender[13]†	105.8	E	105.8	E
Blender[13]†	122.4	E	164.2	E
BMW-256 [30]	8.6	AA	7.85	AA
BMW-512 [30]	13.37	AA	4.06	AA
*Boole [84]	8.9	AA	6.1	AA
Cheetah-256 [29]	15.3	A	10.5	A
Cheetah-512 [29]	83.8	D	15.6	C
Chi-256 [35]	49	C	26	D
Chi-512 [35]	78	D	16	C
CRUNCH-256 [32]	29.9	C	16.9	B
CRUNCH-512 [32]	86.4	D	46.9	E
CubeHash8/1 [9]	200	E	148	E
*DHC [101]	230	E	160	E
DynamicSHA-256 [103]	27.9	B	27.9	D
DynamicSHA-512 [103]	47.2	B	47.2	E
DynamicSHA2-256 [104]	21.9	B	21.9	C
DynamicSHA2-512 [104]	67.3	C	67.1	E
ECHO-256 [7]	38	D	32	D
ECHO-256 [7]	83	D	66	E
ECOH [15]	-	-	-	-
Edon-R-256 [31]	9.1	AA	5.9	AA
Edon-R-512 [31]	13.7	AA	2.9	AA
EnRUPT-256 [79]	8.3	AA	8.3	A
EnRUPT-512 [79]	5.1	AA	5.1	AA
Essence-256 [61]	149.8	E	19.5	B
Essence-512 [61]	176.5	E	23.5	D
FSB-256 [3]	324	E	-	-
FSB-512 [3]	507	E	-	-
Fugue-256 [33]	36.2‡	C	61‡	E
Fugue-512 [33]	74.6‡	D	132.7‡	E

* Submitter has conceded that the algorithm is broken.
† Test platform is Pentium III.
‡ Test platform is Intel Family 6 Model 15 XEON 5150 for 32-bit and Intel Family 15 Model 4 Xeon for 64-bit performance tests. The cpb values are approximated from documented MB/sec.

Table 5. Claimed software speed list of first round candidates of the SHA-3 contest (A-F). Benchmarks are in cycles per byte (cpb) on NIST target platform (Intel Core 2 Duo).

Hash algorithm	**Performance 32 Bit**		**Performance 64 Bit**	
	cpb	class	cpb	class
SHA-256 [77]	29.3	C	20.1	C
SHA-512 [77]	55.2	C	13.1	C
Grøstl-256	22.9	B	22.4	D
Grøstl-512	37.5	A	30.1	E
Hamsi [54]	-	-	-	-
JH-256 [102]	21.3	B	16.8	B
JH-512 [102]	21.3	AA	16.8	D
*Khichidi-1-256 [97]†	74	E	74	E
*Khichidi-1-512 [97]†	148	E	148	E
Keccak-256 [10]	35.4	C	10.1	A
Keccak-512 [10]	68.9	C	20.3	D
LANE-256 [40]	40.4	D	25.6	D
LANE-512 [40]	152.2	E	145.3	E
Lesamnta-256 [37]	59.2	E	52.7	E
Lesamnta-512 [37]	54.5	B	51.2	E
Luffa-256 [16]	13.9	AA	13.4	A
Luffa-512 [16]	25.5	AA	23.2	D
Lux-256 [72]	16.7	A	28.2	D
Lux-512 [72]	14.9	AA	12.5	B
MCSSHA-3 [62]	-	-	-	-
MD6-256 [83]	68	E	28	D
MD6-512 [83]	106	D	44	E
*MeshHash-256 [22]	14.7	A	4.4	AA
*MeshHash-512 [22]	39.1	A	10.3	B
NaSHA-256 [59]	39	D	28.4	D
NaSHA-512 [59]	38.9	A	29.3	E

* Submitter has conceded that the algorithm is broken.

† Test platform: Intel Xeon 1.86 GHz.

Table 6. Claimed software speed list of first round candidates of the SHA-3 contest (G-P). Benchmarks are in cycles per byte (cpb) on NIST target platform (Intel Core 2 Duo).

Hash algorithm	Performance 32 Bit		Performance 64 Bit	
	cpb	class	cpb	class
SHA-256 [77]	29.3	C	20.1	C
SHA-512 [77]	55.2	C	13.1	C
SANDstorm-256 [94]	62.5	E	36.5	D
SANDstorm-512 [94]	296.8	E	95.3	E
Sarmal-256 [96]	19.2	A	10	A
Sarmal-512 [96]	23.3	AA	12.6	B
Sgàil [64]	-	-	61	E
Shabal-256 [14]	18.4‡	A	13.5‡	A
Shabal-512 [14]	18.4‡	AA	13.5‡	C
SHAvite-3_{256} [11]	35.3#	C	26.7#	C
SHAvite-3_{512} [11]	55	B	38.2	E
*SHAMATA-224/256 [2]	15	A	8	AA
*SHAMATA-384/512 [2]	22	AA	11	B
SIMD-256 [56]	12	AA	11	A
SIMD-512 [56]	118	E	85	E
Skein-256 [24]	21.6	A	7.6	AA
Skein-512 [24]	20.1	AA	6.1	AA
Spectral Hash [86]	454.6 †	E	454.6 †	E
*StreamHash [95]	-	-	-	-
SWIFFTX-256 [1]	57	D	-	-
SWIFFTX-512 [1]	57	C	-	-
*Tangle-256 [81]	9	AA	9.4	AA
*Tangle-512 [81]	12.3	AA	12.7	B
TIB3-256 [67]	12.9	AA	7.6	A
TIB3-512 [67]	17.5	AA	6.3	AA
Twister-256	35.8	C	15.8	B
Twister-512	39.6	A	17.5	D
Vortex-256 [53]	46.2	D	69.4	E
Vortex-512 [53]	56	C	90	E
*WAMM [98]	268 †	E	268 †	E
*Waterfall-256 [34]	16.3	A	-	-
*Waterfall-512 [34]	16.3	AA	-	-

* Submitter has conceded that the algorithm is broken.
# Test platform: AMD Sempron 3200+.
† Not specified whether on 32-bit or 64-bit tested, cpb values are approximated from documented MB/sec.
‡ Test platform: AMD Athlon 3200+ 2GHz. The cpb values are approximated from documented MB/sec.

Table 7. Claimed software speed list of first round candidates of the SHA-3 contest (Q-Z). Benchmarks are in cycles per byte (cpb) on NIST target platform (Intel Core 2 Duo).

Hash algorithm	Performance 32 Bit		Performance 64 Bit	
	cpb	class	cpb	class
Maraca [45]	5.5	AA	5.3⋄	AA
NKS2D-256 [80]	178+	E	117+	E
NKS2D-512 [80]	350+	E	243+	E
Ponic [87]	7250∩	E	3250∩	E

⋄ Test platform: Intel Dual E5320 Quad Core.
+ Test platform: AMD Phenom 9500 Quad Core.
∩ Test platform: AMD Athlon.

Table 8. Claimed software speed list of SHA-3 candidates that are not accepted for the first round. Benchmarks are in cycles per byte (cpb) on NIST target platform (Intel Core 2 Duo).

References

[1] Yuriy Arbitman, Gil Dogon, Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. SWIFFTX: A Proposal for the SHA-3 Standard. Submission to NIST, 2008.

[2] Adem Atalay, Orhun Kara, Ferhat Karakoc, and Cevat Manap. SHAMATA HASH FUNCTION ALGORITHM SPECIFICATIONS. Submission to NIST, 2008.

[3] Daniel Augot, Matthieu Finiasz, Philippe Gaborit, Stphane Manuel, and Nicolas Sendrier. SHA-3 proposal: FSB. Submission to NIST, 2008.

[4] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and Raphael C.-W. Phan. SHA-3 proposal BLAKE. Submission to NIST, 2008.

[5] Jean-Philippe Aumasson, Willi Meier, Mara Naya-Plasencia, and Thomas Peyrin. Inside the Hypercube. Cryptology ePrint Archive, Report 2008/486, 2008.

[6] Jean-Philippe Aumasson and Mara Naya-Plasencia. Second preimages on MCSSHA-3. Available online, 2008.

[7] Ryad Benadjila, Olivier Billet, Henri Gilbert, Gilles Macario-Rat, Thomas Peyrin, Matt Robshaw, and Yannick Seurin. SHA-3 Proposal: ECHO. Submission to NIST, 2008.

[8] Daniel J. Bernstein. CubeHash Specification (2.B.1). Submission to NIST, 2008.

[9] Daniel J. Bernstein. CubeHash8/1 Performance. Submission to NIST, 2008.

[10] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche.

[11] Eli Biham and Orr Dunkelman. The SHAvite-3 Hash Function. Submission to NIST, 2008.

[12] Tor E. Bjrstad. Collision for StreamHash. NIST mailing list (local link), 2008.

[13] Colin Bradbury. Blender: A proposed new family of cryptographic hash algorithms. Submission to NIST, 2008.

[14] Emmanuel Bresson, Anne Canteaut, Benot Chevallier-Mames, Christophe Clavier, Thomas Fuhr, Aline Gouget, Thomas Icart, Jean-Franois Misarsky, Mara Naya-Plasencia, Pascal Paillier, Thomas Pornin, Jean-Ren Reinhard, Cline Thuillet, and Marion Videau. Shabal, a Submission to NISTs Cryptographic Hash Algorithm Competition. Submission to NIST, 2008.

[15] Daniel R. L. Brown, Adrian Antipa, Matt Campagna, and Rene Struik. ECOH: the Elliptic Curve Only Hash. Submission to NIST, 2008.

[16] Christophe De Canniere, Hisayoshi Sato, and Dai Watanabe. Hash Function Luffa: Supporting Document. Submission to NIST, 2008.

[17] Christophe De Cannire. Collisions for NKS2D-224. NIST mailing list (local link), 2008.

[18] Anne Canteaut and Mara Naya-Plasencia. Internal collision attack on Maraca. Available online, 2008.

[19] Donghoon Chang, Seokhie Hong, Changheon Kang, Jinkeon Kang, Jongsung Kim, Changhoon Lee, Jesang Lee, Jongtae Lee, Sangjin Lee, Yuseop Lee, Jongin Lim, and Jaechul Sung. Arirang. Submission to NIST, 2008.

[20] Brandon Enright. Collisions for NKS2D-512. NIST mailing list (local link), 2008.

[21] Christian Forler Ewan Fleischmann and Michael Gorski. The Twister Hash Function Family. Submission to NIST, 2008.

[22] Bjrn Fay. MeshHash. Submission to NIST, 2008.

[23] Niels Ferguson and Stefan Lucks. Attacks on AURORA-512 and the Double-Mix Merkle-Damgaard Transform. Cryptology ePrint Archive, Report 2009/113, 2009. urlhttp://eprint.iacr.org/.

[24] Niels Ferguson, Stefan Lucks, Bruce Schneier, Doug Whiting, Mihir Bellare, Tadayoshi Kohno, Jon Callas, and Jesse Walker. The Skein Hash Function Family. Submission to NIST, 2008.

[25] Martin Schlffer Florian Mendel, Tomislav Nad. Collision Attack on Boole-n. NIST mailing list (local link), 2008.

[26] Sren S. Thomsen Florian Mendel. An Observation on JH-512. Available online, 2008.

[27] Scott Fluhrer. Collision Attack on the Waterfall Hash Function. Cryptology ePrint Archive, Report 2008/531, 2008.

[28] Praveen Gauravaram, Lars R. Knudsen, Krystian Matusiewicz, Florian Mendel, Christian Rechberger, Martin Schlffer, and Sren S. Thomsen. Grstl – a SHA-3 candidate. Submission to NIST, 2008.

[29] Danilo Gligoroski. Cheetah hash function is not resistant against length-extension attack. OFFICIAL COMMENT (local link), 2008.

[30] Danilo Gligoroski, Vlastimil Klima, Svein Johan Knapskog, Mohamed El-Hadedy, Jrn Amundsen, and Stig Frode Mjlsnes. Cryptographic Hash Function BLUE MIDNIGHT WISH. Submission to NIST, 2008.

[31] Danilo Gligoroski, Rune Steinsmo degrd, Marija Mihova, Svein Johan Knapskog, Ljupco Kocarev, and Ale Drpal. Cryptographic Hash Function EDON-R. Submission to NIST, 2008.

[32] Louis Goubin, Mickael Ivascot, William Jalby, Olivier Ly, Valerie Nachef, Jacques Patarin, Joana Treger, and Emmanuel Volte. CRUNCH. Submission to NIST, 2008.

[33] Shai Halevi, William E. Hall, and Charanjit S. Jutla. The Hash Function Fugue, 2008.

[34] Bob Hattersley. Waterfall Hash - Algorithm Specification and Analysis. Submission to NIST, 2008.

[35] Phil Hawkes and Cameron McDonald. Submission to the SHA-3 Competition: The CHI Family of Cryptographic Hash Algorithms. Submission to NIST, 2008.

[36] Ethan Heilman. Spectral Hash Collisions. NIST mailing list (local link), 2009.

[37] Shoichi Hirose, Hidenori Kuwakado, and Hirotaka Yoshida. SHA-3 Proposal: Lesamnta. Submission to NIST, 2008.

[38] Kota Ideguchi and Dai Watanabe. Second preimage attack on shamata-512. Available online, 2009.

[39] Sebastiaan Indesteege. Collisions for enrupt. Available online, 2008.

[40] Sebastiaan Indesteege. The LANE hash function. Submission to NIST, 2008.

[41] Sebastiaan Indesteege. Cryptanalysis of dynamic sha. FSE 2009 rump session, slides available online (local link), 2009.

[42] Sebastiaan Indesteege. Practical Preimages for Maraca. Available online, abstract only, 2009.

[43] Sebastiaan Indesteege, Florian Mendel, Martin Schlaeffer, and Christian Rechberger. Practical Collisions for SHAMATA. Available online, 2009.

[44] Tetsu Iwata, Kyoji Shibutani, Taizo Shirai, Shiho Moriai, and Toru Akishita. Aurora: A cryptographic hash algorithm family. Submission to NIST, 2008.

[45] Robert J. Jenkins Jr. [algorithm specification.

[46] John Kelsey and Bruce Schneier. Second Preimages on n-bit Hash Functions for Much Less than 2^n Work. Cryptology ePrint Archive, Report 2004/304, 2004. http://eprint.iacr.org/.

[47] Dmitry Khovratovich and Ivica Nikoli. Cryptanalysis of enrupt. Available online, 2008.

[48] Dmitry Khovratovich and Ivica Nikoli. Cryptanalysis of StreamHash. Available online, 2008.

[49] Dmitry Khovratovich and Ivica Nikoli. Cryptanalysis of DCH-n. Available online, 2008.

[50] Dmitry Khovratovich, Ivica Nikoli, and Ralf-Philipp Weinmann. Cryptanalysis of edon-r. Available online, 2008.

[51] Vlastimil Klima. Huge Multicollisions and Multipreimages of Hash Functions BLENDER-n. Cryptology ePrint Archive, Report 2009/006, 2009. urlhttp://eprint.iacr.org/2009/006.pdf.

[52] Lars R. Knudsen, Florian Mendel, Christian Rechberger, and Sren S. Thomsen. Collision and Preimage Attacks on Vortex as submitted to the SHA-3 competition. Available online, 2008.

[53] Michael Kounavis and Shay Gueron. Vortex: A New Family of One Way Hash Functions based on Rijndael Rounds and Carry-less Multiplication. Submission to NIST, 2008.

[54] Ozgul Kucuk. The Hash Function Hamsi. Submission to NIST, 2008.

[55] Mario Lamberger and Florian Mendel. Practical collision and preimage attack on dch-n. Available online, 2008.

[56] Gatan Leurent, Charles Bouillaguet, and Pierre-Alain Fouque. SIMD Is a Message Digest. Submission to NIST, 2008.

[57] Zhimin Li and Daofeng Li. Collision Attack on NaSHA-384/512. Cryptology ePrint Archive, Report 2009/026, 2009.

[58] Stefan Lucks. Design principles for iterated hash functions. Cryptology ePrint Archive, Report 2004/253, 2004. http://eprint.iacr.org/.

[59] Smile Markovski and Aleksandra Mileva. 2.B Algorithm Specifications and Supporting Documentations. Submission to NIST, 2008.

[60] Smile Markovski and Aleksandra Mileva. 2.B.1 Algorithm Specification. Submission to NIST, 2008.

[61] Jason Worth Martin. ESSENCE: A Candidate Hashing Algorithm for the NIST Competition. Submission to NIST, 2008.

[62] Mikhail Maslennikov. SECURE HASH ALGORITHM MCSSHA-3. Submission to NIST, 2008.

[63] Peter Maxwell. Aww, p*sh! Available online, 2008.

[64] Peter Maxwell. The Sgil Cryptographic Hash Function. Submission to NIST, 2008.

[65] Florian Mendel, Christian Rechberger, and Martin Schlffer. Cryptanalysis of Twister. Available online, 2008.

[66] Florian Mendel and Martin Schlffer. Collisions and Pseudo-Collisions for Sarmal. Available online, 2008.

[67] Miguel Montes and Daniel Penazzi. The TIB3 Hash. Submission to NIST, 2008.

[68] Nicky Mouha. Collision for Khichidi-1. NIST mailing list (local link), 2008.

[69] Mara Naya-Plasencia. Second preimage attack on Ponic. Available online, 2008.

[70] Ivica Nikolic and Dmitry Khovratovich. Second preimage attack on abacus. available online, 2008.

[71] Ivica Nikoli. Preimage attack on Sarmal-512. Available online, 2008.

[72] Ivica Nikoli, Alex Biryukov, , and Dmitry Khovratovich. Hash family LUX - Algorithm Specifications and Supporting Documentation. Submission to NIST, 2008.

[73] Ivica Nikoli and Dmitry Khovratovich. Free-start attacks on nasha. Available online, 2008.

[74] Ivica Nikoli. Preimage attack on Boole-n. Available online, 2008.

[75] National Institute of Standards and Technology. Cryptographic Hash Project. See http://csrc.nist.gov/groups/ST/hash/index.html.

[76] National Institute of Standards and Technology. FIPS 180-1: Secure Hash Standard. April 1995. See http://csrc.nist.gov.
[77] National Institute of Standards and Technology. FIPS 180-2: Secure Hash Standard. August 2002. See http://csrc.nist.gov.
[78] National Institute of Standards and Technology. FIPS 180: Secure Hash Standard. 1993. See http://csrc.nist.gov.
[79] Sean O'Neil, Karsten Nohl, and Luca Henzen. Enrupt hash function specification. Submission to NIST, 2008.
[80] Geoffrey Park. NKS 2D Cellular Automata Hash. Submission to NIST, 2008.
[81] Gary McGuire Rafael Alvarez and Antonio Zamora. The Tangle Hash Function. Submission to NIST, 2008.
[82] R. Rivest. The MD5 Message-Digest Algorithm, 1992.
[83] Ronald L. Rivest. The MD6 hash function – A proposal to NIST for SHA-3. Submission to NIST, 2008.
[84] Gregory G. Rose. Design and Primitive Specification for Boole. Submission to NIST, 2008.
[85] Joachim Rosenthal and Roxana Smarandache. Maximum distance separable convolutional codes. *Applicable Algebra in Engineering, Communication and Computing*, 10(1):15–32, 1999.
[86] Gokay Saldaml, Cevahir Demirkran, Megan Maguire, Carl Minden, Jacob Topper, Alex Troesch, Cody Walker, and etin Kaya Ko. Spectral Hash. Submission to NIST, 2008.
[87] Peter Schmidt-Nielsen. The Ponic Hash Function. Submission to NIST, 2008.
[88] Bruce Schneier and John Kelsey. Unbalanced feistel networks and block cipher design. In *Fast Software Encryption, 3rd International Workshop Proceedings*, pages 121–144. Springer-Verlag, 1996.
[89] Neil Sholer. Abacus: A candidate for sha-3. Submission to NIST, 2008.
[90] Prasanth Thandra. Huge 2ndpreimages and collisions of khichidi-1. Available online, 2009.
[91] Sren S. Thomsen. A near-collision attack on the Blue Midnight Wish compression function. Version 2.0, available online, 2008.
[92] Sren S. Thomsen. Second preimage attack on MeshHash. Available online, 2008.
[93] Sren S. Thomsen. Untangled. Available online, 2008.
[94] Mark Torgerson, Richard Schroeppel, Tim Draelos, Nathan Dautenhahn, Sean Malone, Andrea Walker, Michael Collins, and Hilarie Orman. The SANDstorm Hash. Submission to NIST, 2008.
[95] Michal Trojnara. StreamHash Algorithm Specifications and Supporting Documentation. Submission to NIST, 2008.
[96] Kerem Varc, Onur zen, and elebi Kocair. Sarmal: SHA-3 Proposal. Submission to NIST, 2008.
[97] Natarajan Vijayarangan. A NEW HASH ALGORITHM: Khichidi-1. Submission to NIST, 2008.
[98] John Washburn. WAMM: A CANDIDATE ALGORITHM FOR THE SHA-3 COMPETITION. Submission to NIST, 2008.
[99] David A. Wilson. Abacus: A second-preimage and collision attack. available online, 2008.
[100] David A. Wilson. Constructing Second Preimages in the WaMM Hash Algorithm. Available online, 2008.
[101] David A. Wilson. The DCH Hash Function. Submission to NIST, 2008.
[102] Hongjun Wu. The Hash Function JH. Submission to NIST, 2008.
[103] Zijie Xu. Dynamic SHA. Submission to NIST, 2008.
[104] Zijie Xu. Dynamic SHA2. Submission to NIST, 2008.