Fourth International Conference
Modelling and Development of Intelligent Systems
October 28 - November 1, 2015
"Lucian Blaga" University
Sibiu - Romania

Evaluation of the computational complexity of some hash functions

Olga Korol, Mykhailo Dorokhov

Abstract

The paper analyzes the computational complexity of the software implementation of cryptographic algorithms used in communication systems. A method of forming a cascade control codes of integrity and authenticity of data based on the algorithm UMAC with the final stage cryptographically strong hash function strictly on the basis of universal modular transformations algorithms MASH-1 and MFSH-2 has been proposed. Improved algorithm allows provide high collisional properties of strictly universal hashing, low computational complexity in the processing of large volumes of data and provide high safety performance at the level of modern means of provable resistance cryptographic protection. The resulting estimates of the specific computational complexity of forming MAC show that with increasing length of the processed information data for a fixed level of security specific computational complexity is reduced. For a high level of resistance (128 bits), the same result occurs for data blocks of 2^{15} bytes. For check the reliability of hashing algorithms be used the test suite NIST STS on a particular method study the statistical properties of hash functions. The analysis of the test results showed that the proposed algorithm can provide high-level security of modern provable resistance cryptographic protection.

1 Introduction

Studies have shown that using of hash key multilayer circuits allows you to build effective mechanisms for monitoring the integrity and authenticity of information in telecommunication systems and networks. However, the known multilayer structures (for example, the algorithm UMAC) together with the high speed and the cryptographic strength of the layer by applying a cryptographic transformation (using symmetric block cipher) lose properties of universal hashing, which leads to deterioration of the properties of the collision generated message authentication codes. Proposed in [1, 4, 5] universal hashing method using modular transformations algorithms MASH-1 and MASH-2 allows formation of authenticators (hashes) to provide the required parameter of stability.

The aim is to analyze the computational complexity of some hashing algorithms used in communication systems based on an assessment of time and speed performance processor bandwidth, a comparative evaluation of the computational complexity of the improved algorithm, UMAC, using as substrate pseudorandom algorithms modular transformations MASH-1 and MASH-2 and the statistical based security package NIST STS.

2 Analysis of computational complexity of some hashing algorithms.

For comparison of key hashing schemes in terms of durability and performance is customary to use the unit cpb, where cpb (cycles per byte) - specifies the number of processor cycles required for processing one byte of input information. The complexity of the algorithm is calculated according to the formula

$$Per = Utl * CPU\_clock / Rate \quad (1)$$

where *Utl* - utilization of the processor's core, CPU utilization averaged over the time interval – on each segment, where Idle Thread is not running, the processor is considered employed by any real load. This counter – the sum of the CPU utilization by user, system, and during periods of inactivity (Idle + User + System utilization, the name may be different on different platforms). According to the fact that on most platforms there is a separate idle counter CPU, it is recommended to use the following formula to calculate the consumption of CPU

$$CPU\ Consumption = 100 - Idle\ CPU\ (\%) \quad (2)$$

Rate - possibility of carrying algorithm (byte / sec). To assess the capacity, measured in machine cycles per byte for processing messages of different length we should use cycle per byte, because it gives the opportunity to compare the efficiency between processors running at different speeds. To convert to bytes per second, you need to divide the processor cycles per second (Hz) for the transmitted frames per byte. For example, a processor with a clock speed of 1 GHz at 2.0 cycles per byte performs 1e9 / 2,0 = 0.5e9 bytes per second (500 MB / sec).
Example of the values in terms of durability and performance of one of the algorithms contestants of NIST competition to a national standard hash algorithm SHA-3 is shown in Table. 1. Bits given by security categories of block cipher resistance: 80, 112, 128, 192, 256 [7].

Table 1 Durability and performance parameters of hashing algorithm Blake

Hashing algorithms	Security level *Sec* [біт]	Utilization of kernel, *Utl* (%)	Throughput *Rate* (байт/сек)	Complexity of the algorithm, *Per* (cpb)
Hash (Blake-224)	80, 112, 128, 192	56	19286741	61,5
Hash (Blake-256)	80, 112, 128, 192, 256	56	19192519	62,0
Hash (Blake-384)	80, 112, 128, 192, 256	56	15610497	76,0

Analysis of the table shows that the increase in resistance of the algorithm leads to increasing in complexity.

3 Comparative evaluation of the computational complexity of the improved algorithm UMAC

In developed an improved method of forming control codes of integrity and authenticity of data the first layers is proposed to implement the conversion to traditional high-speed algorithm, UMAC, but cryptographically weak universal hashing schemes, the last layer is proposed to implement using the developed safe (cryptographically strong) strictly universal hashing scheme based on modular transformations.
Formally, the proposed scheme of cascade formation control codes of integrity and authenticity of data presented in Fig. 1.

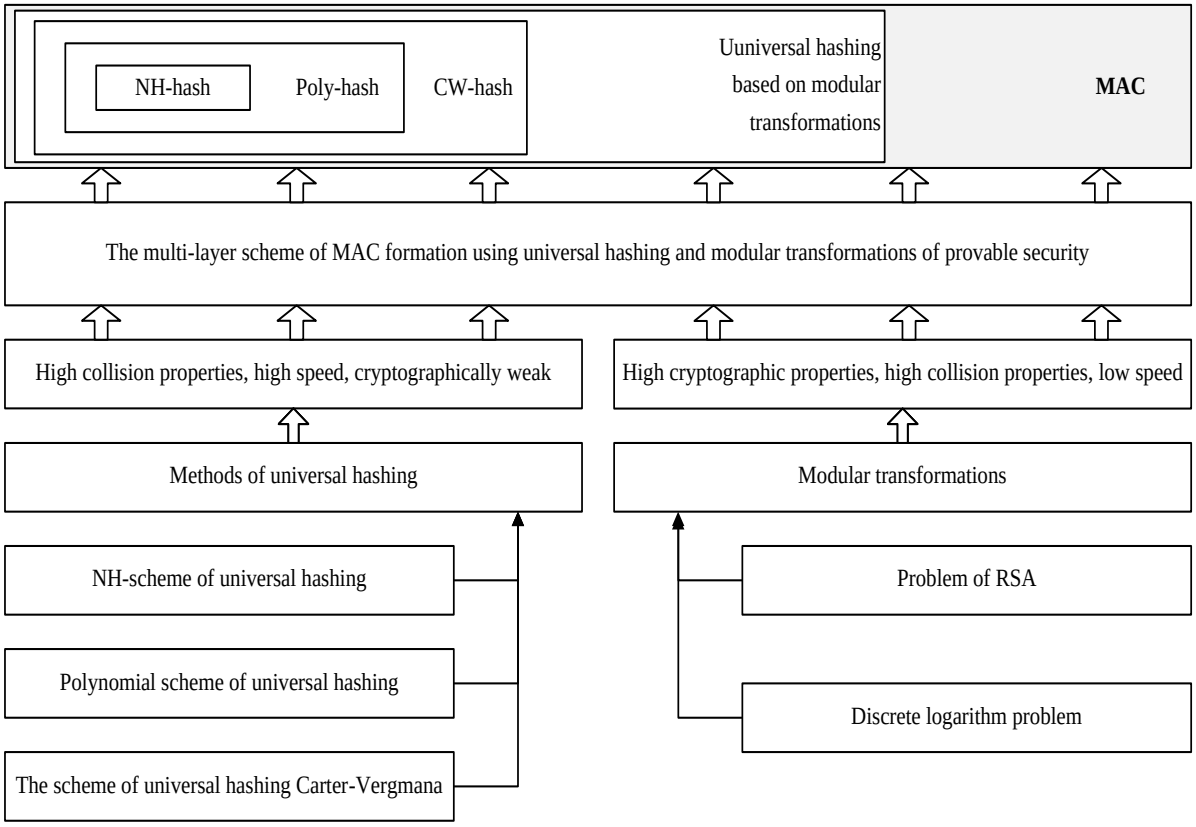

Fig. 1. - Improved scheme of cascade formation of control codes integrity and authenticity of data using modular transformations

For comparison with other schemes in terms of the hash key durability and performance can take the following assumptions. Let one multiplication operation on numbers, of 2^m order requires $\left\lceil \frac{m}{L} \right\rceil$ bitwise addition modulo two (XOR), where L - processor 's digit capacity of the computer system,

$|x|$ - rounded to the nearest integer x. This assumption is most often used when evaluating the complexity of the implementation of cryptographic algorithms [2, 3]. In this case, the assessment of $\left|\frac{m}{L}\right|$ gives an approximate number of cycles of L-bit processor required for the implementation of the multiplication of numbers, which bit length does not exceed from m. At the same time, hashing using modular transformations processing $m/8$ bytes of information data immediately.

Tables 2 and 3 show the results of comparative studies of practical realization of algorithms depending on the number of operations and logical structure of hashing schemes.

Table 2 The number of logical operations in the practice of hashing algorithms

Algorithm	Logical operations									
	AND	OR	XOR	ROTR	SHR	+	ROLs	NOT	MOD	MUL
MD-5	-	-	-	-	-	960	256	-	-	-
RIPEMD-128	-	-	-	-	-	396	128	-	-	-
RIMEMD-160	-	-	-	-	-	650	320	-	-	-
SHA-1	400	240	320	-	-	320	160	-	-	-
SHA-256	320	-	448	384	-	448	-	64	-	-
SHA-384	400	-	560	480	-	560	-	80	-	-
SHA-512	400	-	560	480	-	560	-	80	-	-
MASH1	-	6	4	-	6	1	1	-	6	5
MASH2	-	6	4	-	6	1	1	-	6	260

$\|$ – concatenation, + – addition, *and* – bitwise «AND», *or* – bitwise «OR», *xor* – excluding «OR», *shr* – Shift Right, *rotr* – (Rotate Right) , *ROLs* – cyclic shift to the left by s positions

Table 3 The number of steps and cycles in hashing scheme

Algorithm	Quantity of steps	Number of rounds r	Number of steps in round s	Different constants for each algorithm
MD-	64	4	16	Step
RIPEMD-128	64	4	16	Round
RIMEMD-160	80	5	16	Round
SHA-1	80	4	20	Round
SHA-2 - 256	64	1	64	Step
SHA-2 - 384	80	1	80	Step
SHA-2 - 512	80	1	80	Step

Table. 4 shows the results of comparative studies of the performance of key hashing schemes for fixed safety parameter. Parameter of speed is expressed in an amount of S cycles of 32-bit processor, required for generating one byte of output data. Safety parameter was recorded over the length of the secret key that should be hacked. For modular arithmetic's circuits is given the equivalent length of the key of block symmetric cryptographic algorithm.

Table 4 Estimation of the complexity of hashing algorithms in the number of S cycles of 32-bit processor on a single byte of data processed

Hashing function	Strength level (key length)	Number of rounds *S*
SHA-2 (512)	512	80
SHA-2 (256)	256	64
SHA-1	160	80
RIPEMD-160	160	160
MD5	128	64
Hashing based on modular arithmetic	80	512
	128	1536
	256	7680

The data presented in the Table. 4 show that the use of modular transformations for solving key hashing significantly increases the computational complexity, the performance of algorithms is reduced to 1-2 times. At the same time, developed key hashing schemes have provably resistant safety level (problem of finding the key hash or the inverse image is reduced to solving a certain theoretical and complex problems). In addition, in [1, 4, 5] it is proved that such authentication schemes satisfy the properties of universal hashing to ensure the highest collision characteristics generated by the MAC.
Let estimate the complexity of implementing an improved algorithm UMAC, if the volume of input data is increased.

At the heart of the developed scheme of formation of the MAC using modular transformations is the use of:

- On the first layers - high-speed universal hashing methods (NH-hashing, polynomial hashing, hashing Carter-Vergmana);
- On the last layer - safe strictly universal hash-based modular transformations. The use of a multi-layer structure can also significantly reduce the computational cost of the formation control codes integrity and authenticity of large data sets.

Let us explain the last thesis by the following arguments. Let the first layers of universal hash (as in the method prototype UMAC) are implemented using high-speed (but cryptographically weak) schemes Carter-Vergmana, polynomial structures and so on. (See. Fig. 1). Assume, that the complexity of such a transformation is equal to the complexity of the scheme UMAC, ie about 6 cycles per byte information data. In fact, this estimate is too high, as the most expensive in the scheme UMAC is the last layer of encryption, using the encryption algorithm AES. In other words, the evaluation in six cycles per byte of data to be processed is an estimation of the worst case, i.e. evaluation of the "top" [1, 4, 5]. Assume also that on the last stage of the cryptographic algorithm is used proposed scheme of provably resistant universal hash-based modular transformations instead AES (see. The model in Fig. 1). To assess the complexity of this final stage of formation of the MAC, data in Table 4 should be used. Then the resulting complexity as the number of CPU cycles per byte of data being processed has averaged score on all layers of the transformation in the proposed cascade structure of the calculation codes of controling the integrity and authenticity of data.

Since a major portion of the processed data is supplied only to the first conversion layers (see. The model in Fig. 1) and the last layer with modular cryptographic transformation is used only once for processing hash result for the previous layers of scheme, the resultant estimate of the complexity of large volumes of data to be processed strive for estimating the complexity of the scheme UMAC.
To confirm the above arguments Table. 5 shows a approximate estimation of the complexity of forming control codes of integrity and authenticity of data of proposed scheme using modular transformations.

Table 5 Estimation of the complexity of forming the MAC scheme in an amount of S cycles of 32-bit processor per byte of processed data

Strength level (key length)	Length of input data, bites									
	128	256	512	1024	2048	4096	8192	16384	32768	65536
80	518	262	134	70	38	22	14	10	8	7
128	–	–	1158	582	294	150	78	42	24	15
256	–	–	–	–	7206	3606	1806	906	456	231

The data presented in Table. 5, obtained by calculation by averaging the upper bounds on the complexity of universal hashing on the first layers of the transformations (6 cycles per byte) and evaluation of the complexity of modular transformations (using the round function of the tab. 4. A dash in the table. 5 bear the places in which to hashing Mapping (cryptographic layer) can not be performed. For example, the modular transformations for the level of resistance to an equivalent symmetric cipher key length of 128 bits should be implemented on the module length of not less than 3072 bits (see. Table. 4) that the input data of 256 bytes (2048 bits) of data is impossible. Analysis of the data in table. 5 confirms the above argument to reduce the complexity of the specific conversion (the amount of CPU cycles per byte of input data) with increasing length of the processed information data. In practice, this means that with increasing length of the blocks data, the suggested scheme of formation control codes integrity and authenticity on the computational complexity becomes equivalent to algorithms MD-5 and SHA-1 and SHA-2 algorithms, CBC MAC-Rijndael that is used today in network security protocols (including protocols IPsec).

Table. 6 shows a comparison of the computational complexity of some hash functions. Performance data to the proposed scheme for the MAC with modular transformations are for a minimum level of resistance (capacity of the set of key data of block symmetric cipher is equal to 280) and a sufficient level of resistance (for modular transformation is equivalent to the length of the block symmetric cipher key is 128 bits). Length of generated MAC is equal to 80 bits, and 128, respectively.

Table 6 Estimation of the complexity of forming the MAC with various schemes

Algorithm	Length of input data, bites					
	2048	4096	8192	16384	32768	65536
HMAC-MD5 (128 бит)	9	9	9	9	9	9
HMAC-RIPE-MD (160 бит)	27	27	27	27	27	27
HMAC-SHA-1 (160 бит)	25	25	25	25	25	25
HMAC-SHA-2 (512бит)	84	84	84	84	84	84
CBC MAC-Rijndael (128 бит)	26	26	26	26	26	26
CBC MAC-DES (64 бита)	62	62	62	62	62	62
Improved scheme of UMAS with modular transformations (80 bit)	38	22	14	10	8	7
Improved scheme of UMAS with modular transformations (128 bit)	294	150	78	42	24	15

For all the functions listed in Table. 6 (except for proposed using modular transformations) the complexity of the specific formation control codes integrity and authenticity of data does not depend on the volume of data to be processed (Table. 6 filled by data from the report of the competition NESSIE [2]). For an improved method UMAC using modular transformations specific complexity with increasing length of data to be processed is reduced. Thus, for high-level resistance (equivalent to the length of a symmetric cipher key block is 128 bits) even for blocks of 32768 bytes of data are comparable with the known and used in network security protocols MAC generated algorithm. For a minimum level of resistance (the cardinality of the key data block symmetric cipher is 280) The proposed scheme cascade

formation control codes integrity and authenticity of data using modular transformations is not inferior in speed used to date the formation of MAC algorithms in protocols network security, including the protocols IPsec for data packets of 2048 bytes

Thus, the research results show that the developed scheme of formation control codes integrity and authenticity of data using modular transformations enables high collisional properties of secure hashing. Furthermore, due to the multilayer structure hashing code there is significant reduce of the computational complexity and raising of the processing speed of information messages.

The obtained theoretical results can justify the practical recommendations on the use of the developed models and methods of forming the cascade control codes of integrity and authenticity of data to improve the security of telecommunication systems and networks.

To ensure the integrity and authenticity of data in telecommunications networks used manipulation detection code (MDC), message authentication codes (MAC). For example, a network security protocol IPSec control codes for generating the integrity and authenticity have mandatory ICV algorithms (to ensure compatibility of software from different manufacturers): HMAC-MD5, HMAC-SHA-1, as well as other (additional) algorithms, for example, DES-MAC. These mechanisms are applied by default to ensure the integrity and authenticity of data packets in all implementations of IPv6.

4 Assessment of the statistical security of hashing scheme based on NIST STS package

To test the stability of algorithms, hashing applicants use a set of tests NIST STS for certain research methodology of the statistical properties of hash functions [6].

For testing were taken the following parameters:

– test sequence length $n = 10^6$ bits;

– number of test-sequence $m = 100$;

– significance level $\alpha = 0{,}01$.

Thus, the volume of the test selection is:

– $N = 10^6 \times 100 = 10^8$ bits;

– number (q) for different lengths q = 189, so the statistical portrait of generator is 18900 values of probability P.

Test results of hashing algorithms are summarized in Table. 7.

Table 7 Results of tested hashing algorithms

Generator	The number of tests in which the testing have passed over 99% of sequences	The number of tests in which the testing have passed over 96% of sequences
BBS	134 (71%)	189 (100%)
FIPS 197	126 (67%)	189 (100%)
Blake	130 (69%)	189 (100%)
CubeHash	137 (73%)	189 (100%)
ECHO	139 (74%)	189 (100%)
Grostl	140 (75%)	189 (100%)
Keccak	134 (71%)	187 (98,94%)
MASH-1	101 (53%)	47 (24%)
MASH-2	126 (67%)	189 (100%)
MASH(EC)	141 (74%)	189 (100%)
UMAC 32	167 (88%)	189 (100%)
HMAC-SHA-256	134 (71%)	187 (98%)
EMAC	138 (73%)	189 (100%)
RIPEMD-160	129 (68%)	189 (100%)
UMAC+MASH-2	173 (91%)	189 (100%)

These results confirm the theoretical studies of resistance developed by the cascade method of hashing UMAC with the last layer as pseudorandom pad of modular transformations algorithms MASH-1 and MASH-2.

5 Conclusions

Based on the data shown in Table. 6, it can be argued that the developed scheme of forming a cascade control codes of integrity and authenticity of data using modular transformations is not inferior to the used in the protocols IPSec mechanisms by speed, while increasing the input sequence data provides a significant gain in speed of formation of the hash code. At the same time, the proposed scheme provides demonstrable proof security and conflict-level properties of strictly universal hashing. Since the protocol specification AH and ESP IPSec provides the use of new, more efficient algorithms of ICV, for the protection of data packets in communication networks is proposed the use of the developed models and methods of forming the cascade control codes of integrity and authenticity of data based on the modular transformations.

References

[1] Kuznetsov O.O. *Information security in information systems* / Kuznetsov O.O., Yevseyev S.P., O.G. Korol. - H .: Type. KhNUE, 2011. - 504 p.

[2] *Final report of European project number* IST-1999-12324, *named New European Schemes for Signatures, Integrity, and Encryption,* April 19, 2004 – Version 0.15 (beta), Springer-Verlag.

[3] *Cryptography and Network Security: Principles and Practice* / Stallings W. 1997. - 752

[4] Korol O.G. *Study methods Provision authenticity and integrity of data based on unilateral hash functions* // O.G. Korol, S.P. Evseev. Scientific and technical journal "Information Security". Special Issue (40). - 2008. - P. 50 - 55.

[5] Korol O.G. *Enhanced MAC algorithm based on the use of modular transformations* // O.G. Korol. "Radio Electronics, Computer Science, Control". № 1. - 2015. - P. 60 - 67.

[6] *Statistical testing technique NIST STS and mathematical proofs tests*. - Kharkiv: Institut information technology. - 2004. - 62 p.

[7] *Status Report on the First Round of the SHA-3 Cryptographic Hash Algorithm Competitionhttp* Andrew Regenscheid, Ray Perlner, Shu-jen Chang, John Kelsey.

OLGA KOROL
Simon Kuznets Kharkiv National University of Economics
Department of Information Systems
9A, Prospect Lenina, Kharkiv
UKRAINE

MYKHAILO DOROKHOV
University of Tartu
Institute of Computer Science
2, J. Liivi, Tartu
ESTONIA