Yu. V. Baryshev, Cand. Sc. (Eng); A. O. Komarov

METHODS OF PARALLEL HASHING INFEASIBLE TO GENERIC ATTACKS

The given paper contains the analysis of modern state-of-art of hash functions and the attacks on them. As a result of the carried out analysis, it is determined that one of the most dangerous are generic attacks, based on multicollisions. To improve the infeasibility of constructions, providing paralleling a number of hashing constructions and algorithms have been suggested. Reduction functions of hashing methods on the base of these constructions are determined. Experimental research, carried out enabled to confirm the improvement of infeasibility of the given hashing method to generic attacks on the base of multicollisions.

Key words: *hashing, parallel hashing, multicollision, Joux attack, hash construction, reduction function.*

Introduction

Many processors are developed with built-in two, four and more cores, as a result the programs should completely use this ability of computational platforms. Thus, there appears the necessity in paralleling, the computation parallelization. Nowadays, all paralleling methods of cryptographic hashing are vulnerable to multicollisions[1 – 4]. However, infeasibility to milticollisions does not depend on cryptographic primitives, used at each iteration in the process of hashing, but depends on the constructions used [3 – 5].Consequently. the problem of hashing methods development, that would enable to use the possibility of computations paralleling with keeping the infeasibility of hash functions to breach, remains relevant to this day.

The aim of the given research was to increase the hashing methods infeasibility, that provide paralleling of computations, to multicollisions. the following tasks were solved to reach this aim:

- already known hash functions and attack on them were analyzed;
- hash functions for parallel hashing infeasible to multicollisions were developed;
- hashing algorithms were developed;
- means, implementing these algorithms were developed.

Analysis of the known hashing constructions

I. Damgaard and R. Merkle independently of each other proposed theorems, that show if there exists reduction function infeasible to collisions for input data of state length$f(\cdot)$: $\{0, 1\}^b \times \{0, 1\}^t \rightarrow \{0, 1\}^t$, then infeasible to collisions reduction function for input data of variable length $h(\cdot)$: $\{0, 1\}^* \rightarrow \{0, 1\}^t$can be designed due to iterative invocations of reduction function $f(\cdot)$. Hence, if the reduction function $f(\cdot)$ is vulnerable to certain attack, then the iterated hash function $h(\cdot)$ would also be vulnerable to attacks, through, in general case, the opposite result is not correct [1 – 3].

Hashing construction, proposed by I. Damgaard and R. Merkle has the following form:

$$h_i = f(h_{i-1}, m_i), \quad (1)$$

where h_i – intermediate hash value, obtained after processing of i-th data block; m_i– i-th data block.

There exists numerous similar constructions, that use block cipher as reduction function. One of the most widely used constructions of such type is Davies-Meyer construction. For certain block cipher $E_k(\cdot)$ based on the key k hashing construction has the following form [1, 2, 6, 7]:

$$h_i = E_{m_i}(h_{i-1}) + h_{i-1} \quad (2)$$

It is seen from the construction (2), that in fact given class of constructions is analogous to the construction (1).

S. Lucks proposedthe wide-pipe and double-pipe construction, that uses two reduction functions.

For wide-pipe construction, if$i\in[1;l]$ is processed in the following way:

$$h_i = f'(h_{i-1}, m_i),$$

$$h(M) = f''(h_{i-1}, h_l),$$

Where the output values of $f'(\cdot)$ have two times greater length, than $f''(\cdot)$ [5].

The construction has drawback, concerned with the fact, that two times resources are needed for computation of output values of reduction function $f'(\cdot)$ as compared with $f(\cdot)$ in Merkle-Damgaard construction.

Attacks on hash functions can be classified into two wide categories: brute force attacks and cryptanalytic attacks [1, 2, 4]. All the algorithms are vulnerable to attacks, that do not depend on the algorithm – brute force attack, "birthday" attack. the only way to avoid them – is to increase the length of hash value and the key.

According to [2], attacks on hash functions are characterized as it is shown in Fig. 1.

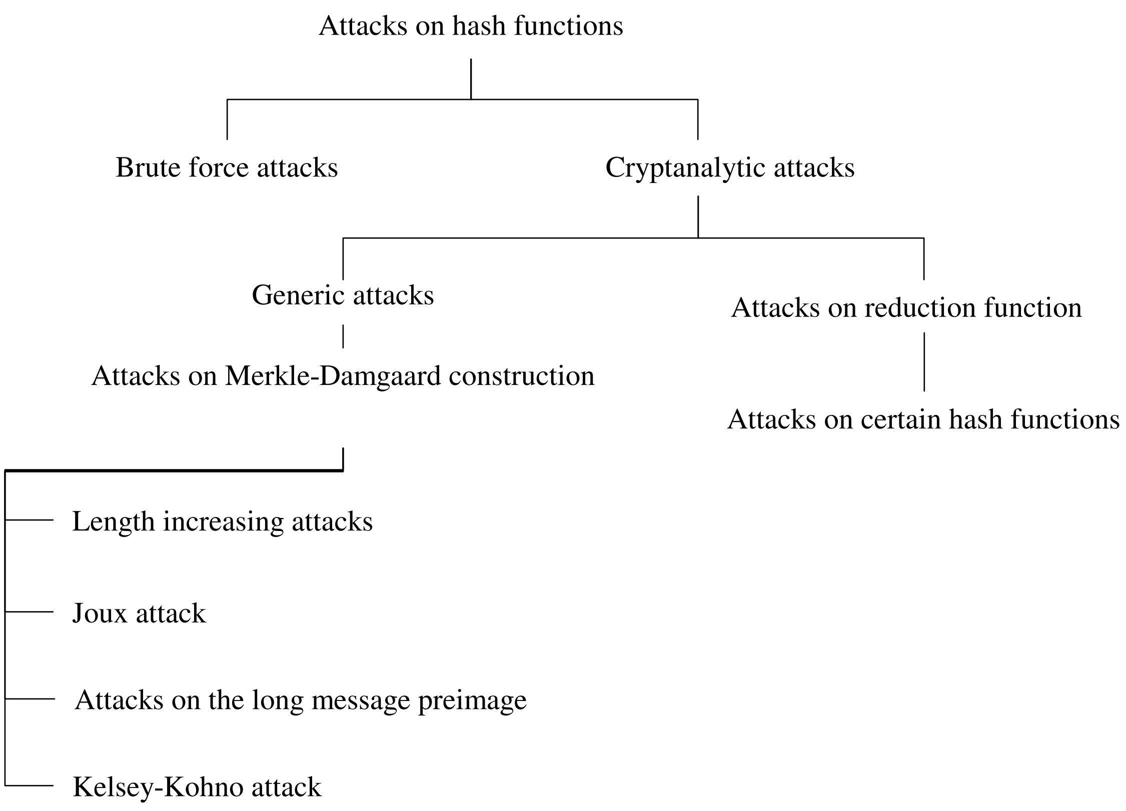
Attacks on hash functions
Brute force attacks
Cryptanalytic attacks
Generic attacks
Attacks on reduction function
Attacks on Merkle-Damgaard construction
Attacks on certain hash functions
Length increasing attacks
Joux attack
Attacks on the long message preimage
Kelsey-Kohno attack

Fig. 1. Gauravaram's attacks classification

Among the attacks of the given classes the most dangerous are generic attacks, because unlike the attacks on reduction function they threaten not only one hash function, but the whole class of hash functions, which use certain construction despite of greater recourses, needed for their implementation. After the advent of Joux attack, greater part of hash functions, which were built on the base of Merkle-Damgaard construction ceased to be considered infeasible.

Especially, it concerned hash functions, which used the idea of Preneel [6], regarding "cascading" of hashvalues – parallel computation of hash functions that provide output values of small data length (16, 32, 64 bits – depending on computational platforms), and concatenation of their output values for obtaining output hash value. Simultaneously, unlike brute force attacks general attacks, for their implementation require far smaller amount of resources and, in certain cases, this amount of resource could be available for intruders, unlike the amount of the resources, necessary for implementation of brute force attacks for hash functions, that have modern length of output hash values (256, 512, 1024, 2048 bits).

Direct "brute force" attack [1, 7] could be realized in order to find preimage by certain hash value or find preimage, that gives certain hash value. The point of the attack is serial or random brute force of the input messages and comparison of computation result of hash function for these messages with certain one. The complexity of such attack is assessed by 2^{n-1}operations of reduction functions computation, where n – the length of the values in bits.

Antoine Joux described the general attack, using multicollisions on Merkel-Damgaard hash construction, where he showed that the construction of 2^l collisions requires less time, then 2^l realizations (implementation) of "birthday" attacks [1, 3, 7]. Such result was achieved due to original approach to collisions construction (designing). Fig. 2 shows the scheme of collisions on Merkel-Damgard hash function.

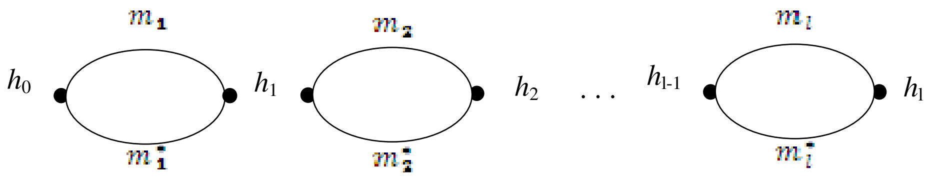

Fig. 2. Graph of messages hashing process in case of Joux attack on Merkel-Damgaard construction

It is seen from Fig 2 that the attack provides l realizations of "birthday" attacks, i.e., it has the complexity of $O(l \cdot 2^{0.5 \cdot n})$ computations of reduction function $f(\cdot)$ for Merkel-Damgaard construction. Characteristic feature of the given method is that all the messages that form the collisions are of the same length [3].

Constructions of the enhanced resistance

Since general attacks can be applied for various hash functions, that use one construction, therefore vulnerability to these attacks is concealed in constructions, that is why methods of counteraction must be introduced on constructions level.

To improve multicollisions strength, the following construction is proposed:

$$\begin{cases} h_i = f(h_{i-1}, m_i, m_{i-r_i}) \\ r_i = rand(m_i) \end{cases};$$

where $rand(\cdot)$ – function, that provides uniform distribution of output values r_i at each iteration (if i - r_i< 0, then block i - r_i + l is selected).

Let Joux attack occur for the given construction. Then, the intruder, according to "birthday" paradox during $2^{0.5 \cdot n}$ computations finds the collision for data block m_i:

$$\begin{cases} h_i = f(h_{i-1}, m_i, m_{i-r_i}) = f(h_{i-1}, m_i^*, m_{i-r_i^*}) \\ r_i = rand(m_i) \\ r_i^* = rand(m_i^*) \end{cases};$$

where $r_i \in [1; l-1]$, $r_i \in N$.

The graph of hashing process obtains the form, shown in Fig. 3.

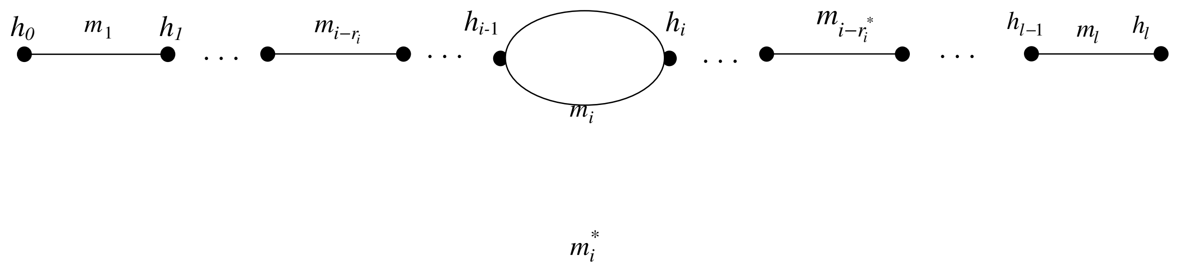

Fig. 3.Graph of hashing process while realization of Joux attack on the suggested construction

Such an attack will become possible only if for $\forall j \in N,\ j \in [1;l],\ j - r_j \neq i,$ that is unlikely in case, when output values of *rand*(·) function conform the law of uniform distribution if $l > 2$.

To improve the resistance of such method with serial computation, it is suggested to determine the number of the second block of data, depending on quarter amount of *rand*(·) function arguments:

$$\begin{cases} h_i = f(h_{i-1}, m_i, m_{i-r_i}) \\ r_i = rand(m_i, m_{i+1}) \end{cases};$$

As the given construction does not allow to construct multicollisions, it can be used for construction of hash functions, providing paralleling:

$$\begin{cases} h_i^{(1)} = f^{(1)}(h_{i-1}^{(1)}, m_i, m_{i-r_i}); \\ h_i^{(2)} = f^{(2)}(h_{i-1}^{(2)}, m_i, m_{i-r_i}); \\ \quad\quad \ldots \\ h_i^{(q)} = f^{(q)}(h_{i-1}^{(q)}, m_i, m_{i-r_i}); \\ \quad\quad r_i = rand(m_i). \end{cases}$$

In case of reduction function, the result of which is processed depending on the value of data block m_{i-r_i}, where r_i depends on the greater amount of blocks (m_i та m_i+1), such hashing construction is suggested (proposed):

$$\begin{cases} h_i^{(1)} = f^{(1)}(h_{i-1}^{(1)}, m_i, m_{i-r_i}); \\ h_i^{(2)} = f^{(2)}(h_{i-1}^{(2)}, m_i, m_{i-r_i}); \\ \quad\quad \ldots \\ h_i^{(q)} = f^{(q)}(h_{i-1}^{(q)}, m_i, m_{i-r_i}); \\ \quad r_i = rand(m_i, m_{i+1}). \end{cases}$$

Maximum amount *rand*(·) function arguments may be $l-1$ amount of data blocks, which input message is divided into. Such construction has the following form for the algorithm with serial processing:

$$\begin{cases} \quad\quad h_i = f(h_{i-1}, m_i, m_{i-r_i}); \\ r_i = rand(m_1, m_2, \ldots, m_{i+1}, m_{i+2}, m_{i+3} \ldots, m_l). \end{cases}$$

In the given case, m_i data block is rejected. For parallel computation the construction will have the following form:

$$\begin{cases} h_i^{(1)} = f^{(1)}(h_{i-1}^{(1)}, m_i, m_{i-r_i}); \\ h_i^{(2)} = f^{(2)}(h_{i-1}^{(2)}, m_i, m_{i-r_i}); \\ \ldots \\ h_i^{(q)} = f^{(q)}(h_{i-1}^{(q)}, m_i, m_{i-r_i}); \\ r_i = rand(m_1, m_2, ..., m_{i+1}, m_{i+2}, m_{i+3} ..., m_l). \end{cases}$$

Thus, constructions, using the third argument in reduction function were developed. Usage of the suggested constructions will allow to counteract multicollisions and accelerate computations at the expense of paralleling the operations being performed, on condition that corresponding safe reduction function will be selected.

Reduction functions

For construction of hash functions, based on hashing constructions, given above, it is necessary to implement reduction functions. Nowadays numerous methods of their implementation are known, but from the point of view at infeasibility/rapidity the most widely spread methods are [1,7-10]:

- on the base of exponentiation by prime integer modules (discrete log) [1, 7]:

$$f(a,b,c) = g^{a+b+c} \bmod p \, .$$

- on the base of elliptical curves (discrete log): point is selected on the elliptical curve x, and coordinate y is calculated for this point. The curve is set by the formula, for instance, $y^2 = x^3+ax^2+bx+c$.

- on the base of non-linear logic functions (SHA2, SEAL) [10, 11]:

$$f(a,b,c) = (a \wedge b) \vee (a \wedge c) \, ;$$

$$f(a,b,c) = a \oplus b \oplus c \, ;$$

$$f(a,b,c) = (a \wedge b) \vee (a \wedge c) \vee (b \wedge c) \, ;$$

$$f(a,b,c) = (a \wedge b) \oplus (a \wedge c) \oplus (b \wedge c) \, ;$$

$$f(a,b,c) = (a \wedge b) \oplus (\bar{a} \wedge c) \, ;$$

The reduction function, based on exponentiation by prime integer modules and on the base of elliptical curves is theoretically resistant to attacks [7]. Although its complexity evaluations are less than the evaluations of brute force attacks implementation, they are rather high to provide hashing, its breach is impossible form practical point of view.

The latter five transformations enable to combine quickly the input values at the expense of the rate of logic operations computations by microprocessors, however their resistance is not theoretically proved and it is impossible to guarantee that they would not be broken in the future. In spite of this, they were investigated in details in the works of different scientists (as a result of their usage in the International Standard of hashing, that were valid for 20 years) did not show their vulnerability [10, 11].

Five pseudo-random sequences of 6400 bytes length were generated for testing. As a result of adding message size and dividing into 256 bits, 1000 data blocks were obtained. Table 1 shows the frequency of certain block selection at each iteration of hashing for each of five sequences. Generation of data block number relatively one data block (m_i) was used.

Table 1

Frequency of block selection as the argument of reduction function

	Experiment №1	Experiment№2	Experiment №3	Experiment№4	Experiment№5
Blocks were not chained	351/1000	40/1000	36/1000	33/1000	40/1000
Blocks were chained at one iteration	400/1000	359/1000	402/1000	389/1000	310/1000
Blocks were chained at two iterations	170/1000	150/1000	170/1000	250/1000	210/1000
Blocks were chained at three and more iterations	81/1000	110/1000	82/1000	40/1000	91/1000

It is seen from Table 1, that approximately a third of blocks do not participate in blocks chaining, nearly 40% are used at certain iteration 1 time, and less than a third of blocks are used in reduction function more than once. In the method of pseudorandom numbers changes, the obtained result of experimental research will change.

The given experiment was carried and for the realization, where the generated number of data block depends on the value of two blocks at each iteration (m_i and m_{i+1}). In the given case each data block became the argument of reduction function minimum one time (not taking into account the fact that it is an argument m_i at i^{th} iteration) (Table 2).

Table2

Frequency of block selection as the argument of reduction function

	Experiment№1	Experiment№2	Experiment№3	Experiment№4	Experiment№5
Blocks were not chained	-	-	-	-	-
Blocks were chained at one iteration	1000/1000	1000/1000	1000/1000	1000/1000	1000/1000
Blocks were chained at two iterations	361/1000	432/1000	371/1000	389/1000	380/1000
Blocks were chained at three and more iterations	271/1000	240/1000	200/1000	271/1000	262/1000

Comparing the data in Tables, the conclusion can be drawn that the amount of chained data blocks increased from 60 to 100% and coupling of blocks on the whole (the number of block usage at other iterations) increased more than 2 times.

Conclusions

From the analysis of the attacks, using multicollisions, it follows that the basic method of their counteraction is violation of hashing process iteration. A number of constructions of message hashing were suggested, they provide usage of the argument in reduction function, that is determined by certain pseudo-random law. Analysis of this approach enabled to substantiate formally the increase of resistance, in particular, to Joux and Kelsey-Kohno attacks. By the result of experimental research it was revealed, that in case of usage of the certain type of data blocks chaining, the obtained infeasibility increase varies depending on the choice of pseudo number generator. Correspondingly, this characteristic will be studied further.

REFERENCES

1. Баришев Ю. В. Методи та засоби швидкого багатоканального гешування даних в комп'ютерних системах : монографія / Ю. В. Баришев, В. А. Лужецький; за заг. ред. В. А. Лужецького. – Вінниця : ВНТУ, 2016. – 144 с.

2. Gauravaram P. Cryptographic Hash Functions: Cryptanalysis, Design and Applications [Електронний ресурс] / Praveen Gauravaram. – 2009. – 298 c. – Режим доступу до ресурсу: http://eprints.qut.edu.au/16372/1/Praveen_Gauravaram_Thesis.pdf. – Назва з екрану.

3. Joux A. Multicollisions in Iterated Hash Functions. Application to Cascaded Constructions / Antoine Joux // Lecture Notes in Computer Science. – 2004. – № 3152. – С. 306 – 316

4. Kelsey J. Second preimages on n-bit hash functions for much less than 2n work / J. Kelsey, B. Schneier // EUROCRYPT. – 2005. – P. 474 – 490.

5. Lucks S. Design Principles for Iterated Hash Functions [Електронний ресурс] / S. Lucks // Cryptology ePrint Archive. – 2004. – 22 с. – Режим доступу до ресурсу: http://eprint.iacr.org/2004/253.pdf. – Назва з екрану.

6. Preneel B. Analysis and Design of Cryptographic Hash Functions: PhD thesis [Електронний ресурс] / Bart Preneel. – Leuven: Katholieke Universiteit Leuven, 1993. – 323 с. – Режим доступу до ресурсу: http://homes.esat.kuleuven.be/~preneel/phd_preneel_feb1993.pdf. – Назва з екрану.

7. Schneier B. Applied Cryptography, Second edition: Protocols, Algorithms, and Source Code in C / Bruce Schneier. – New-York: Wiley Computer Publishing, John Wiley & Sons, Inc, 1996. – 1027 с.

8. Schneier B. The Skein Hash Function Family [Електронний ресурс] / [B. Schneier, N. Ferguson, S. Lucks and others]. – Режим доступу: URL https://www.schneier.com/skein1.3.pdf. – Назва з екрану.

9. Bertoni G. The Keccak sponge function family [Електронний ресурс] / G. Bertoni, J. Daemen, M. Peeters, G. V. Assche. – Режим доступу: URL http://keccak.noekeon.org/specs_summary.html. – Назва з екрану.

10. Secure Hash Signature Standard (SHS): FIPS PUB 180-2. – [Чинний від 2002-08-01]. Processing Standards Publication. – Gaithersburg 2002. – 76 с. (NIST).

11. Implementation of SHA-256 in C [Електронний ресурс] / Режим доступу: URL http://bradconte.com/sha256_c. – Назва з екрану.

BaryshevYurii – Cand. Sc. (Eng), Assistant Professor with Information protection Department, e-mail:yuriy.baryshev@gmail.com.

Komarov Andrii – Master's student with Information protection Department, e-mail: komand9@gmail.com.

Vinnytsia National Technical University.