The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering and Mathematics (EPSTEM), 2025

Volume 33, Pages 62-72

IConTech 2025: International Conference on Technology

Implementation of Encryption Using Glushkov Product of Automata

Zhanat Saukhanova
L.N. Gumilyov Eurasian National University

Altynbek Sharipbay
L.N. Gumilyov Eurasian National University

Gulmira Shakhmetova
L.N. Gumilyov Eurasian National University

Alibek Barlybayev
L.N. Gumilyov Eurasian National University

Raykul Sayat
L.N. Gumilyov Eurasian National University

Khasenov Altay
L.N. Gumilyov Eurasian National University

Abstract: This article discusses the software implementation of an encryption algorithm based on the Glushkov product of finite automata. The main focus of this work is on the application of this mathematical model in cryptography, which allows formalizing the process of key generation and construction of block ciphers. The paper provides a theoretical overview of finite automata without output, their properties and features, as well as a formal definition of the Glushkov product. The encryption algorithm is described, its stages are detailed, including the construction of a key automaton and the process of encryption and decryption of data. To confirm the effectiveness of the proposed method, its software implementation in Python was carried out. Experimental results demonstrate the practical applicability of the algorithm, its cryptographic resistance and potential directions for further development. The study shows that the Glushkov product can serve as a basis for the development of new cryptographic schemes with a high degree of protection.

Keywords: Finite automata, Product Glushkov, Permutation automaton, Cryptography, Automata theory

Introduction

Nowadays, automata theory and cryptography are two important areas of theoretical computer science that are actively used in modern information technologies. Automata theory, which studies models of discrete computations and their behavioral characteristics, provides a powerful tool for analyzing and synthesizing data processing algorithms. In turn, cryptography, based on mathematical methods of transforming information, plays a key role in ensuring confidentiality, integrity and authentication of data in digital systems (Idrees et.al., 2020). One of the fundamental concepts of automata theory is the finite state machine model, an abstract computing device that has been successfully applied to describe a variety of processes, including formal languages, pattern recognition, and processing of input data streams. These principles are also reflected in cryptographic algorithms, especially in stream encryption, where the generation of pseudorandom sequences is based on finite state machines.

There are two main classes of finite automata (Shakhmetova et.al.,2024a) – automata with output and automata without output. A finite automaton without output or an automata-recognizer is designed to check whether an input sequence belongs to a certain set. Its main task is to analyze the input data and make a decision on compliance with a given language or rules. In turn, a automata-transducer converts input data into output, thereby implementing an automaton mapping. Both of these types of finite automata are widely used in cryptographic systems.

Automata with outputs, which include Mealy and Moore machines, are widely used in cryptosystems implementing data transformations (Shakhmetova et.al.,2024b). They are used to build both symmetric and asymmetric cryptographic algorithms, providing key generation, stream encryption, and hashing. In particular, Mealy machines are used in cryptosystems based on weak inversion. The founder of the use of finite automata was the scientist Tao Renji, who in 1985 presented a cryptosystem- FAPKC- Finite Automata Public Key Cryptosystem, which has different versions: FAPKC, FAPKC3 and FAPKC4 (Tao&Chen, 1986; Tao et.al.,1997; Tao &Chen, 1999). In 1995 Gysin proposed a One-key cryptosystem based on a non-linear extended Mealy machine (Gysin, 1995). Lakshmi and Gandhi (2012) in their paper considered Mealy/Moore finite automata and recursive functions as applied to cryptographic problems. In (Abubaker&Kui) a system DAFA was proposed that combines elements of DES and finite state machines. Peña and Torres (2016) developed a method for authenticated encryption based on finite state machines with memory (Peña& Torres, 2016). In 2022, a group of scientists Kodada et al. (2022) presented a cryptosystem for cloud computing based on finite-state machines with finite-order memory (Kodada, 2022).

No less interesting for cryptologist scientists were automata without outputs. This group includes cryptographic methods based on deterministic and non-deterministic finite automata that operate without an output. In 2010, P. Domosi's cryptosystem was proposed based on the Rabin-Scott model (Domosi, 2010). In 2015, Horvath and Domosi developed a cryptosystem based on Glushkov's product (permutation automata) (Domosi&Horváth,2015). In 2016, improved cryptographic schemes using nondeterministic finite automata without outputs were presented, including: Modified Domosi cryptosystem (Khaleel et.al.,2016a), a stream cipher based on nondeterministic finite automata (Khaleel et.al., 2016b), a new block cipher based on finite automata systems (Khaleel et.al., 2016c). In (Jawaharlal, 2020), a multi-factor key cryptographic system based on deterministic finite automata was proposed. In 2023, scientists Moatsum Alawida et al. presented a new image encryption scheme for UAV data protection using a combination of DNA encoding and finite automata (Alawida,2023).

This paper studies finite automata without output and the Glushkov product in the context of their application in cryptography. A theoretical overview of key concepts related to finite automata without output, as well as a formal definition and properties of the Glushkov product, is presented. A model of a cryptographic algorithm based on this mathematical construction is described and its formal analysis is carried out. A software implementation of the studied model is carried out, the results of which are confirmed by experimental data demonstrating the efficiency and practical applicability of this cryptographic protocol. Possible directions for further research in the field of application of the Glushkov product in cryptographic systems are also discussed.

Method

Finite Automata without Outputs

An automaton without an output, also known as a finite automata-recognizer, is a mathematical model used to recognize certain sequences of symbols (languages). Unlike automata with outputs, where each state or transition corresponds to a certain output signal, automata without outputs focus exclusively on determining whether the input sequence belongs to a given language. Despite this feature, this type of automata has found its application in data encryption.

According to (Tao, 2008), the finite automata are an algebraic structure $A = < X, Q, \delta >$, where: $X = \{x_1, x_2, \dots, x_n\}$ is a non-empty and finite set of the input alphabet; $Q = \{q_1, q_2, \dots, q_m\}$ is a non-empty and finite set of states; δ: $Q \times X \to Q$ is the transition function. The elements of the set Q^+ are called states of the finite automata, and the elements of the set X^* are called input symbols, where X^* is the set of all possible words, including the empty set ε, and Q^+ is $Q^+ \backslash \{\varepsilon\}$.

Finite automata without output are divided into deterministic and non-deterministic (Sharipbay, 2015). *A non-deterministic finite automaton (NFA)* is a finite automaton in which the transition function is ambiguous, and

during the operation of the NFA, in one cycle it can make a transition to one or more states different from the initial one. In turn, a *deterministic finite automaton (DFA)* is a special case of an NFA in which for each pair (q, x) there is a unique next state. Further work is based on the use of a deterministic finite automaton without an output, which will be defined in the text as a finite automaton.

The operation of a finite automaton is represented as follows: the relation $(q, ax) \vdash (q', x)$ means that, being in the current state q, reading the input symbol a, the automaton goes to the state q' and then reads the next symbol of the word x. In the row of states $q_0, q_1, \dots, q_m$ ($\{q_0, q_1, \dots, q_m\} \in Q$), the sequence of states $q_1, q_2, \dots, q_{m-1}$ is called *intermediate states*. An extended version of the transition function will be considered:

$$\delta^*: Q \times X^* \to Q^+,$$

where $\delta^*(q, \varepsilon) = q$, $\delta^*(q, xa) = \delta(q, x)\ \delta^*(\delta(q, x), a) = q, q \in Q, x \in X, a \in X^*$. On other words, under the action of empty word ε, the automaton A does not go anywhere, and for each input word $x_1, x_2, \dots, x_n \in X^+$, where $X^+ = X^* \backslash \varepsilon$, and $x_1, x_2, \dots, x_n \in X$ there exist $q_0, q_1, \dots, q_m \in Q$ with transition functions

$$\delta(q_0, x_1) = q_1, \delta(q_1, x_2) = q_2, \dots, \delta(q_{m-1}, x_n) = q_m$$

such that $\delta(q_0, x_1 x_2 \dots x_n) = q_1 \dots q_m$.

In what follows, the transition function is considered in an expanded form, so we will denote it by δ. For clarity and convenience of presentation of the transition function of a finite automata, it is advisable to use a tabular format (see Table 1).

Table 1. Finite automata transition functions

	q_0	q_1	…	q_m
x_1	$\delta(q_0, x_1)$	$\delta(q_1, x_1)$	…	$\delta(q_m, x_1)$
x_2	$\delta(q_0, x_2)$	$\delta(q_1, x_2)$	…	$\delta(q_m, x_2)$
…	…	…	…	…
x_n	$\delta(q_0, x_n)$	$\delta(q_1, x_n)$	…	$\delta(q_m, x_n)$

In the transition table, the rows correspond to the input symbols of the automaton $x \in X^*$, and the columns correspond to its states $q \in Q$. Each cell of this table contains a state to which the automaton transitions in accordance with the transition function $\delta(q, x)$. If the value of the transition function is not defined for some pair (q, x), the corresponding cell remains empty.

Product Glushkov of Automata

The Glushkov product of automata is a collection of permutation automata. Each automaton included in this collection changes its state under the influence of a local state transition function and a global input. In this case, the synchronous action of local transitions affects the global transition of the entire collection of automata. Consequently, the result of the Glushkov product of automata is a new finite automaton with a consistent state dynamic (Domosi & Horváth, 2015). Next, we consider the formal definition of the Glushkov product of automata. First, a permutation automaton will be defined.

An automaton $A = < X, Q, \delta >$ is called *permutational* if all rows in the transition table are permutations of the set of states. In other words, for each pair $b \in Q, x \in X$, there is only one $a \in Q$ such that $\delta(a, x) = b$. (Dömösi & Horváth, 2015).

Let there be a collection of permutation automata $\mathcal{A}_i = (X_i, \mathrm{Q}_i, \delta_i)$, where $i \in \{1, \dots, n\}, n \geq 1$. The feedback function φ_i is defined as a mapping $Q_1 \times \dots \times Q_n \times X \to \mathrm{X}_i$ ($i \in \{1, \dots, n\}$).

The automaton $\mathcal{A} = \mathcal{A}_1 \times \dots \times \mathcal{A}_n(\mathrm{X}, (\varphi_1, \dots, \varphi_n))$ is called the Glushkov product of automata $\mathcal{A}_i$ taking into account the feedback function φ_i, which has a set of final states $Q = Q_1 \times \dots \times Q_n$, a finite set of input symbols X, and the transition function δ is defined as follows:

$$\delta\big((q_1, \dots, q_n), x\big) = \Big(\delta_1\big(q_1 \varphi_1(q_1, \dots, q_n, x)\big), \dots, \delta_n\big(q_n, \varphi_n(q_1, \dots, q_n, x)\big)\Big) \text{ for all } (q_1, \dots, q_n,) \in Q \text{ and } x \in \mathrm{X}.$$

The feedback function $\varphi_i, i \in \{1, \dots, n\}$ is used in the extended sense: $\varphi_i^*: Q_1 \times \dots \times Q_n \times \mathrm{X}^* \to \mathrm{X}_i^*$, where $\varphi_i^*(q_1, \dots, q_n, \varepsilon) = \varepsilon$, and

$\varphi_i^*(q_1, \dots, q_n, ax) = \varphi_i^*(q_1, \dots, q_n, a)\varphi_i\big(\delta_1\big(q_1, \varphi_i^*(q_1, \dots, q_n, a)\big), \dots, \delta_n\big(q_n, \varphi_n^*(q_1, \dots, q_n, a)\big), x\big)$, for all $q_i \in Q, i \in \{1, \dots, n\}, a \in \Sigma^*, x \in \mathrm{X}$. Further in the article, $\varphi_i^*, i \in \{1, \dots, n\}$ will be designated as φ_i.

As is known, the automata $\mathcal{A}_i$ used in Glushkov's product must be isomorphic in states to the original automaton. Two finite automata are called *isomorphic in states* if there is a one-to-one correspondence between their sets of states that preserves the structure of transitions and sets of accepted states.

Formally, two automata $\mathcal{A}_1 = (X_1, \mathrm{Q}_1, \delta_1)$ and $\mathcal{A}_2 = (X_2, \mathrm{Q}_2, \delta_2)$ are considered isomorphic in states if there exist bijections $\sigma\sigma_1: \mathrm{Q}_1 \to \mathrm{Q}_2, \sigma_2: X_1 \to X_2$ such that $\sigma_1: \big(\delta_2(q, x)\big) = \delta(\sigma_1(q), \sigma_2(x))$, where $q \in Q_2, x \in X_2$. In the case when $X_2 = X_1$ and $\sigma_2(x) = x$, we can say that $\mathcal{A}_2$ is an isomorphic automaton $\mathcal{A}_1$ (Tao, 20.

Encryption Algorithm Glushkov Product of Automata

The paper presents a description of algorithm of the light version of the cryptosystem based on Glushkov Product of automata which was presented in work (Domosi et.al., 2019). This cryptosystem served as a basis for subsequent research in the field of automaton cryptography. The cryptosystem developed by Harvard and Dömösi is of considerable interest to specialists studying the use of alternative mathematical models in cryptography. In this cryptosystem, the key automaton is a sequentially functioning Glushkov product with the following properties:

- The Glushkov product is formed from permutation finite automata that are isomorphic in state.
- The sets of state sets of the automata are identical.
- The automata have identical sets of states and input symbols, which are sets of all strings of fixed length in a given alphabet.

Algorithm:

Let $\mathcal{A}_1 = (X_1, \mathrm{Q}_1, \delta_1)$ be a permutation automaton, where $X_1 = \mathrm{Q}_1 = \{0, 1, 2, \dots, 255\}$.

At the first stage, the key automaton is constructed in the following way:

Step 1: n and k positive integers are selected. The number n affects the number of automata in the Glushkov product. The number k determines the number of rounds.

Step 2: the initialization vector $r_1 \dots r_n \in X^n$ is generated, which is truly random. It should be taken into account that the alphabet of pseudo-random numbers X is also the alphabet of the open and encrypted text.

Step 3: digraph $D = (V, E)$ with $V = \{1, \dots, n\}, E = \{(n, 1), (1,2), \dots, (\text{n-1,n})\}$, which defines a D-product: $\mathcal{A}_D = \mathcal{A}_1 \times \dots \times \mathcal{A}_n\big(X^n, (\varphi_1, \dots, \varphi_n)\big)$ of permutation automata $\mathcal{A}_2$,…, $\mathcal{A}_n$ isomorphic in states to $\mathcal{A}_1$, for each $(a_1 \dots, a_n)$, $(x_1 \dots, x_n) \in X^n, i \in \{1, \dots, n\}$ is defined.

Step 4: construct n-1 automata isomorphic to $\mathcal{A}_1$. For this, generate bijective mappings $\psi_1 \dots \psi_{n-1}$. In order to construct a transition table for isomorphic automata $\mathcal{A}_2$,…, $\mathcal{A}_n$, the formula for states is used:

$$\mathcal{A}_2 : \psi_1\big(\delta_1(q_1, x)\big) = \delta_2(\psi_1(q_1), x) \dots$$

…

$$\mathcal{A}_n : \psi_{n-1}\big(\delta_1(q_1, x)\big) = \delta_n(\psi_{n-1}(q_1), x) \dots$$

In this way, transition tables are constructed for all $\mathcal{A}_2$,…, $\mathcal{A}_n$ automata.

The second stage describes the process of encryption of the plaintext using the Glushkov product automaton. The plaintext is read block by block, passing the first block of the plaintext through the key automaton, the first block of the ciphertext is obtained, then the second, third and so on are generated. The blocks of the plaintext are encrypted as follows:

Step 1: the plaintext is split into plaintext blocks $a_1 \dots a_n \in X^n$. The number of symbols in each plaintext block depends on the number of isomorphic automata.

Step 2: a word $w_1, \dots, w_k \in X^n$ of pseudorandom sequences is generated, where $w_1 \dots, w_k \in \Sigma^n$, which are then used in the encryption process as a key at each round. Each vector $w_j = (x_1 \dots x_n), j = 1 \dots k$ has length n.

Step 3: an invertible function φ_i is selected such that

$$\varphi_i: \mathcal{A}_1 \times \dots \times \mathcal{A}_n \times X \to X_i \text{ for all } i = 1 \dots n$$

$$\mathcal{A}_1 = (X_1, \mathrm{Q}_1, \delta_1),\ \mathcal{A}_2 = (X_2, \mathrm{Q}_2, \delta_2)\ , \dots, \mathcal{A}_n = (X_n, \mathrm{Q}_n, \delta_n)$$

$$X_1 = X_2 = \dots = X_n = \mathrm{Q}_1 = \mathrm{Q}_2 = \dots = \mathrm{Q}_n = \overline{(0, 255)}$$

$$\varphi_1\big(a_1 \dots a_n, (x_1, \dots, x_n)\big) = a_n \oplus x_n$$

$$\varphi_i\big(a_1 \dots a_n, (x_1, \dots, x_n)\big) = a_{i-1} \oplus x_{i-1}$$

where $\oplus$ - bitwise addition modulo 2, and $i = 2 \dots n$

4 шаг: key automata $\mathfrak{B} = (X^n, \mathrm{X}^n, \delta_{\mathfrak{B}})$ is Production Glushkov of automata $\mathcal{A}_D$, where for any $(a_1 \dots a_n), (x_1 \dots x_n) \in \mathrm{X}^n$ has transaction function $\delta_{\mathfrak{B}}\big((a_1 \dots a_n), (x_1 \dots x_n)\big) = (b_1 \dots b_n)$. Thus:

$$b_1 = \delta_{\mathcal{A}_1}(a_1, \varphi_1\big(a_1, \dots, a_n, (x_1, \dots, x_n)\big)), \text{ where } \varphi_1\big(a_1, \dots, a_n, (x_1, \dots, x_n)\big) = a_n \oplus x_n$$

$$b_2 = \delta_{\mathcal{A}_2}(a_2, \varphi_2\big(b_1, a_2, \dots, a_n, (x_1, \dots, x_n)\big)), \text{ where } \varphi_2\big(b_1, a_2, \dots, a_n, (x_1, \dots, x_n)\big) = b_1 \oplus x_1$$

…

$$b_n = \delta_{\mathcal{A}_n}(a_n, \varphi_n\big(b_1, b_2, \dots, a_n, (x_1, \dots, x_n)\big)), \text{ where } \varphi_n\big(b_1, b_2, \dots, a_n, (x_1, \dots, x_n)\big) = b_{n-1} \oplus x_{n-1}$$

Шаг 5: $(d_1 \dots, d_n)$ and $(e_1 \dots, e_n)$ vectors are defined.

If k=1, these vectors are defined as follows:

$(d_1, \dots, d_n) = (a_1 \dots, a_n)$ – bloc of plaintext.

$$(e_1, \dots, e_n) = \delta_{\mathfrak{B}}((d_1 \dots, d_n), w_1).$$

$$e_1 = \delta_{\mathcal{A}_1}(d_1, \varphi_1\big(d_1, \dots, d_n, (x_1, \dots, x_n)\big) = \delta_{\mathcal{A}_1}(d_1, d_n \oplus x_n),$$

$$e_2 = \delta_{\mathcal{A}_2}(d_2, \varphi_2\big(e_1, d_2, \dots, d_n, (x_1, \dots, x_n)\big) = \delta_{\mathcal{A}_2}(d_2, e_1 \oplus x_1),$$

…

$$e_n = \delta_{\mathcal{A}_n}(d_n, \varphi_n\big(e_1, e_2, \dots, d_n, (x_1, \dots, x_n)\big) = \delta_{\mathcal{A}_n}(d_n, e_{n-1} \oplus x_{n-1})\ .$$

If k>1, encryption is performed in the following way:

$$(e_1 \dots, e_n) = \delta_{\mathfrak{B}}((d_1 \dots, d_n), w_1 \dots w_i)$$

After k rounds, states $(e_1 \dots, e_n)^k$ will be obtained. The ciphertext $c_1 \dots c_n$ is the concatenation of the calculated state blocks $(e_1, \dots, e_n)$ at each round.

The third stage describes the decryption process. The key machine reads the encrypted data in blocks sequentially and, having processed the first block of the ciphertext $c_1, \dots, c_n \in X^n$, calculates the corresponding blocks of the plaintext in the order reverse to their original sequence. This process occurs according to the steps described below:

Step 1: the ciphertext is divided into ciphertext blocks $c_1, \dots, c_n \in X^n$. The number of symbols in each ciphertext block depends on the number of isomorphic automata.

Step 2: a word $w_1, \dots, w_k \in X^n$ of pseudorandom sequences is generated, where $w_1, \dots, w_k \in X^n$, which were used as a key during encryption in each round. Each vector $w_j = (x_1 \dots x_n), j = 1 \dots k$ has length n.

Step 3: an invertible function φ_i is selected similar to that described in step 3 of the encryption stage.

Step 4: during decryption, the inverse automaton $\mathfrak{B}^{-1} = (X^n, \mathrm{X}^n, \delta_{\mathfrak{B}^{-1}})$, is used to the key automaton $\mathfrak{B} = (X^n, \mathrm{X}^n, \delta_{\mathfrak{B}})$, which was used during encryption. Thus, the vector $(d_1 \dots, d_n)$ is restored as follows:

If k=1, the vectors are defined as follows:

$(e_1, \dots, e_n) = (c_1 \dots, c_n)$ – ciphertext block

$$(d_1 \dots, d_n) = \delta_{\mathfrak{B}}((e_1 \dots, e_n), w_1).$$

$$d_n = \delta_{\mathcal{A}_n^{-1}}(e_n, \varphi_n(\, e_1, \dots\ e_{n-1}, e_n\ , (x_1, \dots, x_n)) = \delta_{\mathcal{A}_n^{-1}}(e_n, e_{n-1} \oplus\ x_{n-1})\ ,$$

$$d_{n-1} = \delta_{\mathcal{A}_{n-1}^{-1}}(e_{n-1}, \varphi_{n-1}(\, e_1, \dots\ e_{n-1}, d_n\ , (x_1, \dots, x_n)) = \delta_{\mathcal{A}_{n-1}^{-1}}(e_{n-1}, e_{n-2} \oplus\ x_{n-2})\ ,$$

$$d_1 =\ \delta_{\mathcal{A}_1^{-1}}(e_1, \varphi_1(\, e_1, \dots\ d_{n-1}, d_n\ , (x_1, \dots, x_n)) = \delta_{\mathcal{A}_n^{-1}}(e_1, d_n \oplus\ x_n).$$

If $k > 1$, decryption is done in the following way:

$$(d_1 \dots, d_n) = \delta_{\mathfrak{B}}((e_1 \dots, e_n), w_1 \dots w_i)$$

Thus, we can obtain a block of plaintext after k rounds of inverse transformation.

Results and Discussion

In this section of the article will discuss a demonstration example that will clearly demonstrate the use of the Glushkov product of automata as encryption and decryption of plaintext.

We will further illustrate an example on how to apply product Glushkov of automata on encryption and decryption process. Take $\mathcal{A}_1 = (X_1, \mathrm{Q}_1, \delta_1)$ be a permutation automaton, where $X_1 =\ \mathrm{Q}_1 = \{0, 1, 2, 3\}$. The transition table 2:

Table 2. Transition functions of $\mathcal{A}_1$

	0	**1**	**2**	**3**
0	1	2	3	0
1	0	3	2	1
2	2	1	3	0
3	0	1	3	2

We will choose n=3, it is mean that will be constructed two isomorphic automata $\mathcal{A}_2$ and $\mathcal{A}_3$. For that we generate bijective mappings ψ_1 and ψ_2.

$$\psi_1: \psi_1(0) = 3, \psi_1(1) = 1, \psi_1(2) = 0, \psi_1(3) = 2$$

$$\psi_2: \psi_2(0) = 2, \psi_2(1) = 0, \psi_2(2) = 3, \psi_2(3) = 1$$

According bijective mapping construct transition function for $\mathcal{A}_2$ and $\mathcal{A}_3$ (table 3 and table 4).

Table 3. Transition functions of $\mathcal{A}_2$

	0	**1**	**2**	**3**
0	0	1	3	2
1	3	2	0	1
2	3	1	2	0
3	1	0	2	3

Table 4. Transition functions of $\mathcal{A}_3$

	0	**1**	**2**	**3**
0	2	1	3	0
1	2	0	1	3
2	0	3	1	2
3	3	0	1	2

Thus, key automata are Glushkov product of automata $\mathcal{A}_D = \mathcal{A}_1 \times \mathcal{A}_2 \times \mathcal{A}_3\big(X^3, (\varphi_1, \varphi_2, \varphi_3)\big)$

Let, plaintext is 103, for encrypting this text we generate key $w = (x_1, x_2, x_3) = (3, 0, 1)$, k = 1.

Define vector $(d_1, d_2, d_3) = (1, 0, 3)$.

Encryption is done as follows:

$$e_1 = \delta_{\mathcal{A}_1}(d_1, \varphi_1\big(\, d_1, d_2, d_3, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_1}(d_1, d_3 \oplus x_3) = \delta_{\mathcal{A}_1}(1, 3 \oplus 1) = \delta_{\mathcal{A}_1}(1, 2) = 1$$

$$e_2 = \delta_{\mathcal{A}_2}(d_2, \varphi_2\big(\, e_1, d_2, d_3\,, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_2}(d_2, e_1 \oplus x_1) = \delta_{\mathcal{A}_2}(0, 1 \oplus 3) = \delta_{\mathcal{A}_2}(0, 2) = 3$$

$$e_3 = \delta_{\mathcal{A}_3}(d_3, \varphi_3\big(\, e_1, e_2\,, d_3, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_3}(d_3, e_2 \oplus x_2) = \delta_{\mathcal{A}_3}(3, 3 \oplus 0) = \delta_{\mathcal{A}_3}(3, 3) = 2$$

So, cyphertext $(c_1, c_2, c_3) = (1, 3, 2)$

For decrypting cyphertext we need generate inverse of key automata (table 5, 6, 7).

Table 5. Transition functions of $\mathcal{A}_1^{-1}$

	0	**1**	**2**	**3**
0	3	0	1	2
1	0	3	2	1
2	3	1	0	2
3	0	1	3	2

Table 6. Transition functions of $\mathcal{A}_2^{-1}$

	0	**1**	**2**	**3**
0	0	1	2	3
1	2	3	1	0
2	3	1	2	0
3	1	0	2	3

Table 7. Transition functions of $\mathcal{A}_3^{-1}$

	0	**1**	**2**	**3**
0	3	1	0	2
1	1	2	0	3
2	0	2	3	1
3	1	2	3	0

Key use same us encryption stage $w = (x_1, x_2, x_3) = (3, 0, 1)$

The decryption process is similar to encryption, but in reverse.

$$(e_1, e_2, e_3) = (1, 3, 2)$$

$$d_3 = \delta_{\mathcal{A}_3^{-1}}(e_3, \varphi_3\big(\, e_1, e_2, e_3\,, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_3^{-1}}(e_3, e_2 \oplus x_2) = \delta_{\mathcal{A}_3^{-1}}(2, 3 \oplus 0) = \delta_{\mathcal{A}_3^{-1}}(2, 3) = 3$$

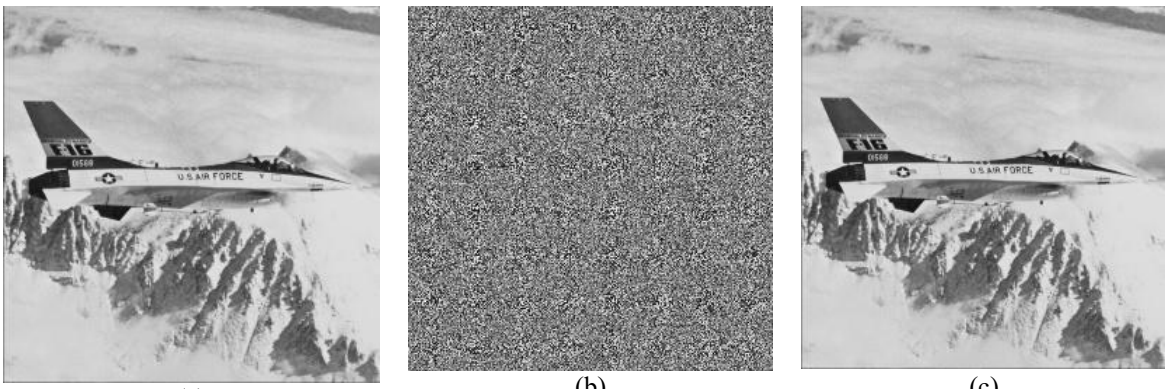
$$d_2 = \delta_{\mathcal{A}_2^{-1}}(e_2, \varphi_2\big(\, e_1, e_2, d_3\,, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_2^{-1}}(e_2, e_1 \oplus x_1) = \delta_{\mathcal{A}_2^{-1}}(3, 1 \oplus 3) = \delta_{\mathcal{A}_2^{-1}}(3, 2) = 0$$

$$d_1 = \delta_{\mathcal{A}_1^{-1}}(e_1, \varphi_1\big(\, e_1, d_2, d_3\,, (x_1, x_2, x_3)\big) = \delta_{\mathcal{A}_1^{-1}}(e_1, d_3 \oplus x_3) = \delta_{\mathcal{A}_1^{-1}}(1, 3 \oplus 1) = \delta_{\mathcal{A}_1^{-1}}(1, 2) = 1$$

So, we get decrypted text $(d_1, d_2, d_3) = (1, 0, 3)$

The above example is given to illustrate the operation of encrypting and decrypting using Glushkov product of automata work. Obviously, in practice that cryptographic algorithm is much more complicated. For more information on how to build these types of automata, see the work (Domosi et.al., 2019).

The software implementation of the considered cryptosystem was made by the Python programming language. To demonstrate the operation of the algorithm under consideration, a gray image *Airplane.tiff* with size 512x512 pixels was taken (https://sipi.usc.edu/database). To encrypt this image, 16 finite automata were taken, and the number of rounds is 8. For images with size 512*512 pixels will be encrypted 16 384 blocks. The key of size 16 is generated randomly, in our case, the same key is used in each round. Results of encrypting and decrypting are shown on figure 1.



(a) (b) (c)

Figure 1. (a) Original image, (b) encrypted image, (c) decrypted image.

NIST Test Results

The statistical tests developed by the National Institute of Standards and Technology (NIST) Information Technology Laboratory are a set of 15 methods designed to assess the degree of randomness of binary sequences. These tests are based on the analysis of various statistical characteristics inherent in random sequences. Successful completion of these tests is interpreted as evidence of high cryptographic strength of the data under study (Pareschi et.al., 2012). Thus, the NIST test set is an effective tool for analyzing the randomness of encrypted information, which in turn indicates the reliability of the cryptographic methods used.Table 8 contains the results of NIST statistical tests on the experimental image.

Table 8. NIST results

Test Name	**P-Value**	**Conclusuion**
01. Frequency Test:	0.6114533432130285	True
02. Block Frequency Test:	0.9512055879042081	True
03. Run Test:	0.04314964466418219	True
04. Run Test (Longest Run of Ones):	0.37066285180866043	True
05. Binary Matrix Rank Test:	0.7262509735848526	True
06. Discrete Fourier Transform (Spectral) Test:	0.9487819898568091	True
07. Non-overlapping Template Matching Test:	0.8067096223594483	True
08. Overlappong Template Matching Test:	0.5373254259563948	True
09. Universal Statistical Test:	0.9470101746108605	True
10. Linear Complexity Test:	0.8350400376461669	True
11. Serial Test:	0.9804137326823029	True
	0.815036343351716	True
12. Approximate Entropy Test:	0.8733927390000743	True
13. Cumulative Sums (Forward):	0.37820269368211135	True
13. Cumulative Sums (Backward):	0.37820269368211135	True
14. Random Excursion Test:		
STATE *xObs*	*P-Value*	*Conclusion*
-4 0.3757434402332362	0.9959714520151304	True
-3 1.997984	0.8494239179678532	True
-2 4.9027160493827155	0.4278679719139393	True

-1	2.0	0.8491450360846096	True
1	8.6	0.12612244119041105	True
2	8.359012345679012	0.13752824367716304	True
3	5.431520000000001	0.3655134039059185	True
4	4.419958350687214	0.4906636246748899	True

15. Random Excursion Variant Test:

STATE	*COUNTS*	*P-Value*	*Conclusion*
-9.0	61	0.503594688207976	True
-8.0	71	0.5964828214814786	True
-7.0	80	0.6948866023724733	True
-6.0	74	0.5793585907298551	True
-5.0	70	0.4795001221869535	True
-4.0	89	0.7687675574059754	True
-3.0	93	0.8248125741940059	True
-2.0	94	0.8064959405073401	True
-1.0	99	0.9436280222029834	True
+1.0	109	0.5245182802130763	True
+2.0	125	0.30743416592739536	True
+3.0	128	0.3759205825480747	True
+4.0	111	0.7687675574059754	True
+5.0	111	0.7954250063905932	True
+6.0	110	0.8311704095417624	True
+7.0	107	0.8908084551935809	True
+8.0	128	0.6092056132701683	True
+9.0	150	0.3911725228101395	True

Passing all NIST tests comprehensively demonstrates that the encrypted image meets international standards for cryptographic data strength.

Conclusion

In this research, finite automata without outputs were studied as a basis for constructing encryption algorithms. A cryptographic scheme based on the Glushkov product of automata was implemented in software, allowing for the encryption and decryption of data using a formal automaton-based model. The developed program demonstrates the operation of the encryption process through the sequential composition of isomorphic automata, ensuring high variability and structural complexity of the key automaton.

The considered algorithm has several advantages. It features a clear mathematical structure, flexibility in configuring encryption parameters (such as the number of automata and rounds), and a modular architecture that facilitates software and potentially hardware implementation. The use of isomorphic permutation automata increases resistance to cryptanalytic attacks by introducing combinatorial complexity. Furthermore, the ability to generate pseudorandom transformations on each encryption round contributes to high diffusion and confusion properties, essential for secure encryption. The compositional structure of the Glushkov product contributes to enhancing data protection by increasing the cryptographic strength of the encryption process. The results of NIST statistical tests confirmed the high level of randomness and reliability of the proposed method.

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM Journal belongs to the authors.

Conflict of Interest

The authors declare that they have no conflicts of interest

Funding

This research is funded by the Science Committee of the Ministry of Education and Science of the Republic of Kazakhstan (Grant No. AP19677422).

Acknowledgements or Notes

*This article was presented as an oral presentation at the International Conference on Technology (www.icontechno.net) held in Trabzon/Türkiye on May 01-04, 2025.

References

Abubaker, S., & Wu, K. (2012). Dafa-a lightweight des augmented finite automaton cryptosystem. *International Conference on Security and Privacy in Communication Systems* (pp. 1-18). Berlin, Heidelberg: Springer.

Alawida, M., Teh, J. S., & Alshoura, W. H. (2023). A new image encryption algorithm based on DNA state machine for UAV data encryption. *Drones*, *7*(1), 38.

Domosi, P. B., Horvath, G., Medveczki, M. S., & Salga, P. (2019). *U.S. Patent No. 10,419,207*. Washington, DC: U.S. Patent and Trademark Office.

Domosi, P. (2010). A novel cryptosystem based on finite automata without outputs. In *Automata, formal languages and algebraic systems* (pp. 23-32). https://www.worldscientific.com/doi/abs/10.1142/9789814317610_0002

Domosi, P., & Horváth, G. (2015). A novel cryptosystem based on Gluškov product of automata. *Acta Cybernetica*, *22*(2), 359-371.

Gysin, M. (1995). A one-key cryptosystem based on a finite nonlinear automaton. *International Conference on Cryptography: Policy and Algorithms* (pp. 165-173). Berlin, Heidelberg: Springer.

Idrees, B., Zafar, S., Rashid, T., & Gao, W. (2020). Image encryption algorithm using S-box and dynamic Hénon bit level permutation. *Multimedia Tools and Applications*, *79*(9), 6135-6162.

Jawaharlal, S. M., Narayanan, A., Santhar, S., & Sivaramalingam, B. (2019). *U.S. Patent No. 10,320,758*. Washington, DC: U.S. Patent and Trademark Office.

Khaleel, G., Turaev, S., & Tamrin, M. M. (2016c). A new block cipher based on finite automata systems. *International Journal on Perceptive and Cognitive Computing*, *2*(1), 23-26.

Khaleel, G., Turaev, S., & Zhukabayeva, T. (2016b). A novel stream cipher based on nondeterministic finite automata. In *Information technologies in science, management, social sphere and Medicine* (pp. 439-444). Atlantis Press.

Khaleel, G., Turaev, S., Tamrin, M. I. M., & Al-Shaikhli, I. F. (2016a). Performance and security improvements of Domosi's cryptosystem. *Internatıonal Journal of Applied Mathematics & Statıstics*, *55*(2), 32-45.

Kodada, B. (2022). FSAaCIT: Finite state automata based one-key cryptosystem and chunk-based indexing technique for secure data de-duplication in cloud computing. *Authorea Preprints*.

Lakshmi, S. (2012). *On finite state machines and recursive functions–applications to cryptosystems*. *(*Doctoral dissertation).

Pareschi, F., Rovatti, R., & Setti, G. (2012). On statistical tests for randomness included in the NIST SP800-22 test suite and based on the binomial distribution. *IEEE Transactions on Information Forensics and Security*, *7*(2), 491-505.

Peña, P. I. S., & Torres, R. E. G. (2016). Authenticated encryption based on finite automata cryptosystems. *13th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE)* (pp. 1-6). IEEE.

Shakhmetova, G., Barlybayev, A., Saukhanova, Z., Sharipbay, A., Raykul, S., & Khassenov, A. (2024b). Enhancing visual data security: A novel FSM-based ımage encryption and decryption methodology. *Applied Sciences*, *14*(11), 4341.

Shakhmetova, G., Saukhanova, Z., Sharipbay, A., Barlybayev, A., Sayat, R., Altay, K., & Saukhanova, M. (2024a). Ontological model of cryptosystems based on the theory of finite automata. In *2024 International Visualization, Informatics and Technology Conference (IVIT)* (pp. 8-13). IEEE.

Sharipbay, A. (2015). *Theory of language and automata*. Almaty: Evero.

Tao, R. (2008). *Finite automata and application to cryptography.* Chicago: Springer.

Tao, R., & Chen, S. (1986). Two varieties of finite automaton public key cryptosystem and digital signatures. *Journal of Computer Science and Technology*, *1*(1), 9-18.

Tao, R., & Chen, S. (1999). The generalization of public key cryptosystem FAPKC4. *Chinese Science Bulletin*, *44*, 784-790.

Tao, R., Chen, S., & Chen, X. (1997). FAPKC3: A new finite automaton public key cryptosystem. *Journal of Computer Science and Technology*, *12*(4), 289-305.

Author(s) Information

Zhanat Saukhanova
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan

Altynbek Sharipbay
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan

Gulmira Shakhmetova
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan
Contact e-mail: *sh_mira2004@mail.ru*

Alibek Barlybayev
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan

Raykul Sayat
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan

Khassenov Altay
L.N. Gumilyov Eurasian National University
Pushkin street 11, Astana, Kazakhstan

To cite this article:

Saukhanova, Z., Sharipbay, A., Shakhmetova, G., Barlybayev, A., Sayat, R., & Altay, K. (2025). Implementation of encryption using Glushkov product of automata. *The Eurasia Proceedings of Science, Technology, Engineering and Mathematics (EPSTEM), 33,* 62-72.