IRSTI 27.39.21 DOI: https://doi.org/10.26577/JMMCS2024-v124-i4-a8

Zh. Temirbekova[1], **Z. Abdiakhmetova**[1*], **S. Tynymbayev**[2], **S. Altynbek**[3], **G. Ziyatbekova**[1,4]
[1]Al-Farabi Kazakh National University, Almaty, Kazakhstan
[2]International University of Information Technologies, Almaty, Kazakhstan
[3] Kazakh University of Technology and Business, Astana, Kazakhstan
[4] Institute of Information and Computational Technologies SC MSHE RK, Almaty, Kazakhstan
*e-mail: zukhra.abdiakhmetova@gmail.com

DEVELOPMENT OF AN ENCRYPTION LIBRARY TO ENSURE SECURITY IN THE INTERNET OF THINGS

The article explores the primary applications of homomorphic encryption. A review of existing solutions revealed that current libraries only handle bits or bit arrays and do not support operations like division and subtraction. However, real-world problems require the capability to perform integer-based operations. This gap highlighted the necessity for developing homomorphic division and subtraction functionalities, along with the creation of a custom library for working with integers. A method for performing homomorphic division and subtraction on encrypted data is proposed. Using this method and architecture as a foundation, a library was developed to support homomorphic operations on integers, including addition, subtraction, multiplication, and division. This significantly enhances the potential applications of homomorphic encryption. The article also provides execution time measurements for various operations on encrypted data and evaluates the efficiency of the developed library.

Key words: homomorphic encryption, confidential data protection, IoT devices, ARM microcontroller, cryptography.

Ж.Е. Темирбекова[1], З.М. Абдиахметова[1*], С. Тынымбаев[2], С. Алтынбек[3], Г. Зиятбекова[1,4]
[1]Әл-Фараби атындағы Қазақ ұлттық университеті, Алматы қ., Қазақстан
[2]Халықаралық ақпараттық технологиялар университеті, Алматы қ., Қазақстан
[3]Қазақ технология және бизнес университеті, Астана қ., Қазақстан
[4]ҒК ҚР ҒЖБМ Ақпараттық және есептеуіш технологиялар институты, Алматы қ., Қазақстан
*e-mail:*zukhra.abdiakhmetova@gmail.com

Интернет заттардың қауіпсіздігін қамтамасыз ету үшін шифрлы кітапхана әзірлеу

Мақалада гомоморфты шифрлаудың негізгі қолданбалары қарастырылады. Қолданыстағы шешімдерді шолу ағымдағы кітапханалардың тек биттерді немесе разрядтық массивтерді өңдейтінін және бөлу және алу сияқты операцияларды қолдамайтынын көрсетті. Дегенмен, нақты әлемдегі мәселелер бүтін санға негізделген операцияларды орындау мүмкіндігін талап етеді. Бұл олқылық бүтін сандармен жұмыс істеу үшін теңшелетін кітапхананы құрумен қатар гомоморфты бөлу және алу функцияларын дамыту қажеттілігін көрсетті. Шифрланған деректерде гомоморфты бөлу мен алуды орындау әдісі ұсынылған. Осы әдісті және архитектураны негіз ретінде пайдалана отырып, қосу, алу, көбейту және бөлуді қоса алғанда, бүтін сандардағы гомоморфты операцияларды қолдау үшін кітапхана әзірленді. Бұл гомоморфты шифрлаудың әлеуетті қолданбаларын айтарлықтай арттырады. Мақалада шифрланған деректер бойынша әртүрлі операциялардың орындалу уақытының өлшемдері берілген және әзірленген кітапхананың тиімділігі бағаланады.

Түйін сөздер: гомоморфты шифрлау, құпия деректерді қорғау, IoT құрылғылары, ARM микроконтроллері, криптография.

Ж.Е. Темирбекова[1], З.М. Абдиахметова[1*], С. Тынымбаев[2],
С.А. Алтынбек[3], Г.З.Зиятбекова[1,4]
[1]Казахский национальный университет имени Аль-Фараби, г. Алматы, Казахстан
[2]Международный университет информационных технологий, г. Алматы, Казахстан
[3]Казахский университет технологии и бизнеса, г. Астана, Казахстан
[4]Институт информационных и вычислительных технологий КН МНВО РК, г. Алматы, Казахстан
*e-mail: zukhra.abdiakhmetova@gmail.com

Разработка библиотеки шифрования для обеспечения безопасности в интернете вещей

В статье рассматриваются основные приложения гомоморфного шифрования. Обзор существующих решений показал, что текущие библиотеки обрабатывают только биты или битовые массивы и не поддерживают такие операции, как деление и вычитание. Однако для реальных задач требуется возможность выполнять целочисленные операции. Этот пробел подчеркнул необходимость разработки функций гомоморфного деления и вычитания, а также создания пользовательской библиотеки для работы с целыми числами. Предлагается метод выполнения гомоморфного деления и вычитания для зашифрованных данных. Используя этот метод и архитектуру в качестве основы, была разработана библиотека для поддержки гомоморфных операций над целыми числами, включая сложение, вычитание, умножение и деление. Это значительно расширяет потенциальные приложения гомоморфного шифрования. В статье также приводятся измерения времени выполнения различных операций над зашифрованными данными и оценивается эффективность разработанной библиотеки.

Ключевые слова: гомоморфное шифрование, защита конфиденциальных данных, устройства IoT, микроконтроллер ARM, криптография.

1 Introduction

Today, IoT applications and devices are widely used in all sectors, from the national security industry to manufacturing.[1]. Although the total number of devices is projected to reach 83 billion by 2024, the safety of these devices is a major concern. In case of failure to take appropriate security measures, there is a risk that any IoT-enabled device will not be able to work, as well as user data.
While IoT devices expected to number 83.5 billion by 2024, their security remains a significant concern. Without proper security measures, any IoT-enabled device could potentially compromise both its functionality and user data. According to a 2023 report by Palo Alto Networks, 98data on the network is not stored in secret, and allowing attackers to spy on unencrypted network traffic, collect personal or confidential information, and then use that data by the attacker for their personal purposes. According to SAM Seamless Network, more than 1.5 billion IoT devices were attacked in 2021, including about 900 million phishing attacks. The protection of IoT devices plays a key role in ensuring national security for several reasons:

- Critical infrastructure: many IoT devices are used in critical infrastructure such as energy networks, transport systems and healthcare. Violation of these systems can have serious consequences for society, the economy and national security;

- Cyber attack capabilities: IoT devices often have limited resources to protect against cyber threats. Using them as access points for cyber attacks can open the door to large-scale attacks on large systems and networks;

- Mass use: every year the number of connected IoT devices increases, which increases the surface area of potential cyber threats. The protection of these devices is becoming increasingly important to prevent mass cyber attacks;

- Privacy and Privacy: IoT devices can collect a lot of personal information about users. Damage to this information can endanger the privacy, privacy and security of citizens, which is also part of national security;

- Economic consequences: cyber attacks on IoT devices can lead to significant economic losses, both for individual companies and for the country as a whole. This can affect investments, production processes and competitiveness in the global market;

- Legislation and regulation: given the growing awareness of threats, many countries are developing legislative and regulatory measures to protect IoT devices and data that affect national policy and security.

Thus, the protection of IoT devices is a necessary component of the overall national security strategy, which provides protection against cyber threats, the preservation of critical infrastructure and the protection of personal information of citizens.

IoT devices are very different from traditional computers because they do not have the processing power and resources to perform complex and resource-intensive tasks on their own. Unlike full-fledged computers that can process and analyze large amounts of data, perform calculations, and work with different applications, IoT devices most often perform one specialized task or function. IoT devices operate within a specific group or cluster to collaboratively address problems. To secure the information transmitted among these devices, it is essential to encrypt the data and enable operations on this encrypted data as if it were unencrypted, ensuring the integrity of the overall results. This achieved through homomorphic encryption, which implemented in Atmel AVR microcontrollers (such as, Atmega 32u4, and Atmega 2560, DFRobot Beetle BLUE, Atmega 328) used in applications like national security, consumer electronics, and manufacturing.

In recent years, a lot of work has appeared in the world on TGSH (full homomorphic encryption) for IoT devices. Sujoy S. R., Goyuri P., Deepika N. (2021) shows in their work that full homomorphic encryption algorithms can be applied to IoT applications and devices, as well as aimed at ensuring higher computing speed while maintaining data confidentiality [3]. Goran D., Milan M., Pavle V. [4] in their work "Evaluating the implementation of homomorphic encryption in an IoT device"evaluated the features of the homomorphic encryption mechanisms of BFV and BGV and measured computational performance. The Raspberry Pi 4 evaluates the encryption schemes on the IoT platform based on the B model, showing that homomorphic encryption operations can be used in embedded devices and is primarily aimed at improving privacy and providing high bandwidth and low latency to speed up applications. Among the representatives of the Russian scientific community, the works of the following scientists can be especially noted: I. B. Saenko, V. A. Desnitsky (Moscow), I. V. Kotenko (Sverlosk), P. D. Zegzhda (St. Petersburg). Scientists from the II and CT of the Republic of Kazakhstan, such as R.G. Biyashev, S.E. Nysanbayeva, N.A. Kapalova and A. Kunbolat, have made significant contributions to research into cryptographic information protection tools. Their work covers a wide range of aspects of cryptography, including the development of new methods and algorithms for ensuring data security, as well as the analysis and optimization of existing cryptographic systems. Taking into account the analyzes made, there is a need for methods, algorithms that effectively ensure the security of IoT applications and devices and determine whether the topic under consideration is an urgent problem.

2 Materials and methods

Homomorphic encryption is a new cryptographic method that allows you to process and analyze data in encrypted form, without reverse encryption of pre-encrypted data. This cryptosystem is especially useful for IoT applications where data privacy and security are important. Currently, the scientific works of scientists in the field of homomorphic encryption for IoT devices and devices are noted: Zwika Brakerski et al. [5] introduced a new full homomorphic encryption scheme in 2012 that significantly reduced the computational complexity of fully homomorphic encryption schemes. This allowed for experimental use in IoT applications with limited computing resources. Hu Wang et al. [6] in 2016 he proposed a homomorphic encryption scheme for IoT devices that can be implemented on devices with limited resources. The scheme is based on approximate calculations, this homomorphic encryption reduced the calculation time. Hui Wang et al. [7] in 2017 proposed a homomorphic encryption scheme that can handle multiple data sources with different formats and sizes for the privacy of IoT data. The scheme uses an encryption key management approach that provides flexible and efficient computing with encrypted data.
In 2020, Xiaolei Dong et al. [8] proposed a data storage and retrieval scheme for IoT devices utilizing homomorphism encryption. This approach enables efficient storage and retrieval of encrypted data while ensuring its confidentiality and security.
L.K. Babenko, E.A. Tolomonenka [9] in their work, IoT shows that it is possible to solve the problem by using a homomorphic encryption algorithm to securely transmit and work with data between devices, that is, since light cryptography does not allow processing encrypted data. In general, the authors concluded that it is effective to use homomorphic encryption to protect IoT applications and maintain data privacy. For the first time in a scientific article, a homomorphic encryption library architecture was developed to ensure the security of the IoT system of devices, and Atmel AVR was introduced into the flash memory of the microcontroller. Currently, many well-known companies are engaged in the production of microcontrollers. In addition, unlike standard products such as operational amplifiers or voltage regulators, microcontrollers from different companies differ from each other. Many of them have different command systems. AVR to ensure data security in the initial state [10, p. 178], 8051 [11, p. 18], PIC [12, p. 317], the arm [13, p. 12] microcontroller was considered. Comparisons conducted regarding the discharge characteristics of various microcontrollers, their energy consumption, types and capacities of built-in memory, clock frequencies, as well as the composition and cost of peripheral devices. As shown in Table 1, clock frequency is a critical parameter when comparing microcontrollers, as it influences their operational speed. A higher clock frequency results in faster performance and more accurate responses to external events. Among the microcontrollers analyzed, both AVR and ARM exhibit high

clock frequencies. However, the AVR microcontroller has the added benefit of support from the Arduino platform and programming resources. Additionally, the AVR microcontroller demonstrated favorable metrics in terms of memory capacity and types, as well as cost.

Table 1: Comparison AVR, 8051, PIC, ARM microcontrollers

Feature	AVR	8051	PIC	ARM
Architecture	Modified Harvard	Modified Harvard	Harvard	ARM (Cortex)
Instruction Set	RISC	CISC	RISC	RISC
Clock Speed	Up to 20 MHz	12 MHz	Up to 10 MHz	32 MHz
Power Consumption	Low power	Low to medium power	Medium to high power	Low to high power depending on core
Peripherals	Basic to advanced (e.g., ADC, UART)	Basic peripherals, limited expansion	Advanced (e.g., ADC, UART, USB, CAN)	Extensive (e.g., USB, Ethernet, touch)
Community Support	Large (Arduino)	Established but niche	Large and growing	Very large, strong support for embedded systems
Price	2000-7000	12000-55600	9250-60000	9850-6750

During the analysis, high-performance engine control timers and an AVR microcontroller chosen, as they encompass all the essential peripherals, including multiple interfaces. These integrated features offer a significant advantage over microcontrollers with fewer peripherals in the proposed architectures. In the work, SD and microSD cards were used to work with various files. They can store information read from sensors and transducers. The total amount of memory on Arduino boards is up to 4 KB. Allows you to store different amounts of information using SD cards [13]. A memory module card was used to connect the SD card to the Arduino board. It has a resistor, voltage regulator and card slot. Using the module supports reading, writing and storing data on the memory card. Preparing the module for the MicroSD card. To save data to a microSD card, you need the module shown in Figure 1 below.

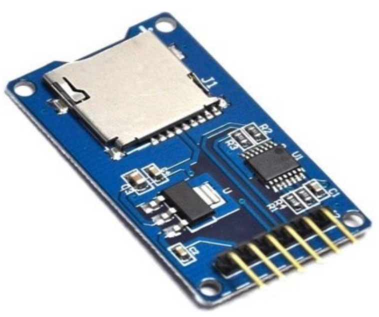


Figure 1: MicroSD card module

Before using the microSD card module through the microcontroller from the AtmelAVR group, it must be formatted. To do this, the following actions are performed:
- a microSD card is inserted into the computer. Go to "Computer"and right-click on the microSD card icon. In the menu that appears, the "Format"button is pressed. A new window will appear. Select "FAT32 click "Start"and the instructions will follow. A module for a MicroSD card is included. After formatting the MicroSD card, you need to connect it to the module.

The microcontroller utilizes a MicroSD card along with an Arduino board to encrypt text data. In the subsequent step, encrypted files read from and written to the MicroSD card. A MicroSD card module employing the SPI communication protocol facilitates the connection between the MicroSD card and the Arduino board.

The principle scheme for registering data on an SD card using microcontrollers is drawn in the Fritzing software environment. The principle scheme of the ATmega328 microcontroller is shown in Figure 2.

Table 2: Arduino Uno data logging to MicroSD card module.

micro-SD card	Vcc	GND	CS	SCK	MISO	MOSI
Arduino Uno	+5V	GND	4	13,12	11	11

To connect a MicroSD card, 6 contacts are used, and the interaction is carried out through the SPI interface. To connect the card, you need an ATmega328 microcontroller, a card module and 6 wires. Registration of Arduino Uno data on a MicroSD card is shown in Table 2. Digital I / O is connected as follows: MOSI and MISO contacts on the Arduino Uno board to D11, SCK - D13, CS-4, VCC - +5 v, GND-GND. The board has connectors for connecting to 3.3 and 5 volts. The power supply of the MicroSD card is 3.3 volts, so it is necessary to use a microcontroller with the same power supply, otherwise voltage level converters will be required. The connection to the ATmega328 microcontroller with the MicroSD card module and the principle scheme of the ATmega328 microcontroller can be seen in Figure 2.

On the Arduino UNO board for two microcontrollers, respectively, two Quartz resonators with a frequency of 16 MHz are installed, which is a high speed, that is, it transmits pulses per 16 million seconds. The mentioned microcontroller considers exactly these pulses, and then, by their number, makes a conclusion about how much time has passed since the start of any procedure. The connection to the ATmega328 microcontroller with the microSD card module can be seen in Figure 3.

One approach in homomorphic encryption is to use variable polynomial rings, where encryption is performed using mathematical structures based on ring polynomials. However, despite their danger, such algorithms face a number of problems, such as high computational complexity, large memory costs, and performance issues. Such an improvement of the homomorphic encryption algorithm in variable polynomial rings is relevant in the field of modern cryptography. Main problems of existing homomorphic encryption algorithms Homomorphic encryption algorithms often require significant computational resources, since operations on encrypted data are more complex than similar operations on open data. For example, multiplication and division operations in polynomial rings can be very time-consuming. Polynomial rings often lead to a significant increase in the size of encrypted

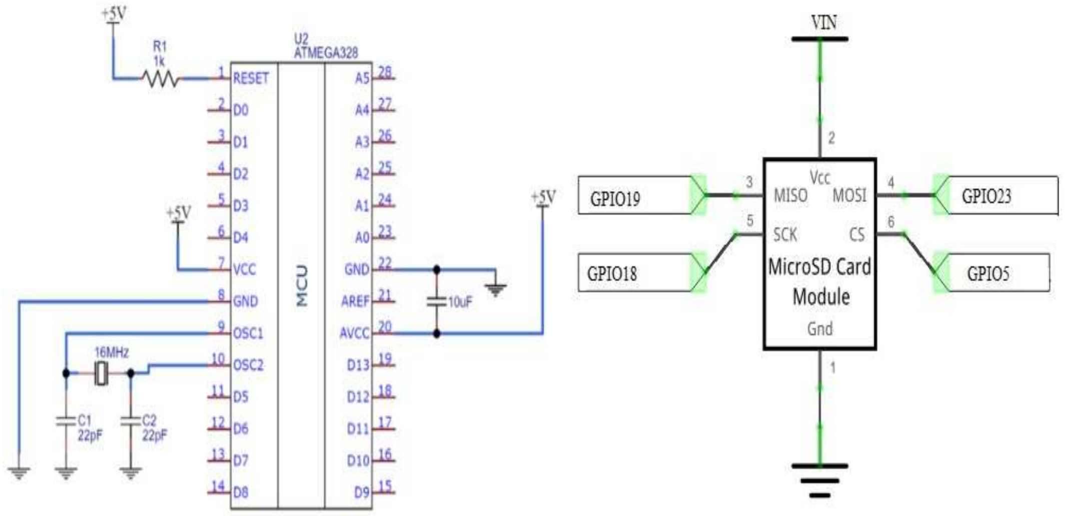


Figure 2: a-ATmega328 microcontroller communication; B MicroSD card module and ATmega328 Microcontroller binding principle scheme

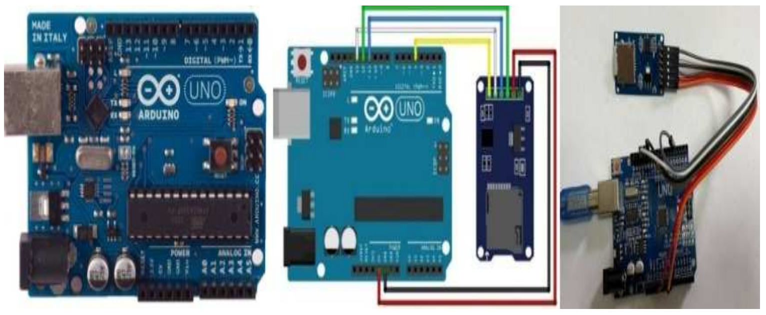


Figure 3: Connecting the ATmega328 microcontroller to the microSD card module

data. In order to efficiently process encrypted data, it is necessary to use a lot of memory, which makes the method ineffective for every application [14]. Currently, the development and refinement of homomorphic encryption is a pressing issue. In the article division and substruction operations were added to encrypted data in the system proposed by S.F. Krendelev.

Each number $a \in Z_n$ is associated with a polynomial $a(x) = a_0 + a_1 x + ... + a_k x^k$, where k and a_i are chosen randomly. For two polynomial representations of numbers a_0 and b_0 according to their structure, the free term of their sum $a(x) + b(x)$ and product $a(x) * b(x)$ is $a(0) + b(0)$ and $a(0) * b(0)$. Then $\phi : Z_n[x] \to Z_n[y]$, $x = c_0 + c_1 y + ... + c_t y^t = \phi(y)$ is a homomorphism preserving addition and multiplication. S.F. Krendelev's implementation is more efficient than Gentry's. In addition, it has a number of disadvantages:
1. An unbounded increase in the degrees of polynomials can result in inefficient calculations;
2. Even though all operations essentially conducted on empty terms, it is necessary to store and perform calculations on high-order polynomials in memory;
3. In the S.F. Krendelev system, division and substruction operations on encrypted data are

not performed.
Algorithm 1. Fitting a polynomial to an integer.
1) Let's say $z \in Z$ any integer corresponding to the polynomial. Let's say a number is $n > 0-$ the degree of the polynomial ;
2) $a_0, a_1, ...a_n \in Za$ polynomial whose coefficients are randomly selected $f(x) = a_0 + a_1x + ... + a_nx^n$ is created;
3) $f(x_0) = f(\frac{p}{q}) = a_0 + a_1(\frac{p}{q}) + ... + a_n(\frac{p}{q})^n$ is calculated, from here, $q^n f(\frac{p}{q}) = q^n a_0 + q^{n-1}pa_1 + ... + p^n a_n q^n f(\frac{p}{q}) \in Z$ is taken;
4) $g_z(x) = q^n f(x) - q^n f(\frac{p}{q}) + z$ polynomial corresponding to the number z is calculated.
Algorithm 2. Matching an integer to a polynomial (algorithm inverse of 1).
1) Let the polynomial obtained as a result of Algorithm 1 $g_z(x)$ be given;
2) Then $g_z(x_0) = q^n f(x_0) - q^n f(\frac{p}{q}) + z = z$ the original number.
Concept 1. Built according to Algorithm 1 $h : Z \to Z[x]$-representation. The representation given there is injective.
The proof is evident from the random selection of coefficients, establishing a homomorphism from the ring of integers Z to the polynomial ring. Now, let's examine the creation of a homomorphism from the ring Z to the ring $Z[x]$.

Definition 1 (Ring Homomorphism): Let $f : A- > B$, where A and B are rings that include addition, multiplication, zero, and one.
Theorem 1: Consider the ring $Z[x]$ of integers and the polynomial ring, where $P[x]$ is the representation generated by the first algorithm. Then P$P[x]$is a homomorphism.

Proof: $z_1, z_2 \in Z$ let us show that property (1) holds, i.e
$P(z_1 + z_2) = P(z_1) +_{z[x]} P(z_2)$.
$P(z_1) +_{z[x]} P(z_2) = q^n f_1(x) - q^n f_1(\frac{p}{q}) + z_1 + q^n f_2(x) - q^n f_2(\frac{p}{q}) + z_2 =$
$= q^n a_0 + q^n a_1 x + ... + q^n a_n x^n - (q^n a_0 + q^{n-1}pa_1 + ... + p^n a_n) + z_1 +$
$+ q^n b_0 + q^n b_1 x + ... + q^n b_n x^n - (q^n b_0 + q^{n-1}pb_1 + ... + p^n b_n) + z_2 =$
$=(a_1 + b_1)(q^n x - q^{n-1}p) + ... + (a_n + b_n)(q^n x - p^n) + z_1 + z_2 =$
$=q^n c_0 + q^n c_1 x + ... + q^n c_n x^n - (q^n c_0 + q^{n-1}pc_1 + ... + p^n c_n) + z_1 + z_2$
$=P(z_1 + z_2)$
where, $c_i = a_i + b_i, i \in \{0..n\}$ Based on the above calculations, it is possible to create an encryption and reverse encryption algorithm.
Implementation of division operation on encrypted data in S.F. Krendelev system:
Definition 1. Let Z be some field. A polynomial in one variable in the field Z is a formal summation of the form:

$$f(x) = a_n x^n + a_{n-1}x^{n-1} + a_1 x + a_0, \qquad (1)$$

where,$a_i \in Z, i \in \{0, 1, ..., n\} n \in N$ Note. Any element of the field Z is considered to be a polynomial of zero degree, a polynomial of arbitrary degree with zero coefficients is a zero polynomial, and a unit polynomial of the Z field is considered to be a unit polynomial and are denoted by $\nu(x)$ and $E(x)$, respectively.
On the set of all polynomials in one variable in the field Z, the addition and multiplication operations of polynomials can be determined according to the following rules. Suppose $f(x)$ is a polynomial of type 2 and

$$g(x) = b_n x^n + b_{n-1}x^{n-1} + b_1 x + b_0, \tag{2}$$

where, $b_i \in Z, j \in \{0, 1, \ldots, m\} m \in N - 2.2$ polynomial of the form [15, p. 13; 16, p. 423].
Let the polynomial f(x) be written as:

$$f(x) = a_0 x^n + a_1 x^{n-1} + a_{n-1} + a_n (a_0 \neq 0) \tag{3}$$

Let's show it as follows:

$$f(x) = (x - c) * (b_0 x^{n-1} + b_1 x^{n-2} + b_{n-2}x + b_{n-1}) + r, \tag{4}$$

If we mentally open the brackets to the right of the equality and equalize the coefficients of the same degree in the equations (3) and (4), is obtained:

$$\begin{aligned} b_0 &= a_0, \\ b_1 &= a_1 + cb_0, \\ b_2 &= a_2 + cb_1, \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ b_k &= a_k + cb_{k-1} \\ b_{n-1} &= a_{n-1} + cb_{n-2} \\ &\ldots\ldots\ldots\ldots\ldots\ldots \\ r &= a_n + cb_{n-1} \end{aligned}$$

(4) is usually written in Tabular Form and is called the Gorner scheme. The Gorner scheme is shown in the form of Table 3.

Table 3: Gorner scheme

	a_0	a_1	...	a_k	...	a_n
$x = c$	b_0	$b_1 = a_1 + cb_0$	...	$b_k = a_k + cb_{k-1}$	...	$r = a_n + cb_{n-1}$

Obviously, the coefficients $r = f(c)$, b_i are the coefficients of the division $q(x)$ when dividing $f(x)$.
Theorem 2. For any number C in the field Z, the polynomial $f(x)$ of the type n degree (3) can always and moreover uniquely be represented as:

$$f(x) = b_0 + b_1(x - c) + b_2(x - c)^2 + \ldots + b_n(x - c)^n, b_n \neq 0, \tag{5}$$

Representation in the form (5) is called classifying the value f(x) in degrees of the x - c substruction.
Now, let's examine the challenge of establishing a homomorphism from the ring Z to the ring

$Z[x]$. Definition 2 (Ring homomorphism): Let $f : - >$, where A and B are rings that include addition, multiplication, zero, and one. The function f is a ring homomorphism if it satisfies the following conditions [15-16]:

$$f(a +_a b) = f(a) +_B f(b) \tag{6}$$

$$f(a -_A b) = f(a) -_B f(b) \tag{7}$$

$$f(a *_A b) = f(a) *_B f(b) \tag{8}$$

$$(a/_A b) = f(a)/_B f(b) \tag{9}$$

$$f(0_A) = 0_B \tag{10}$$

$$f(1_A) = 1_B \tag{11}$$

Let Z be given a ring of integers, a ring of polynomials $Z[x]$, and $P(x)$ compiled by Algorithm 1. Then $P(x)$, $P : Z \to Z[x]$ homomorphism.
Encryption and reverse encryption algorithm Key generation. If the Secret Key of a given x_0 cipher is, z_1, $z_2 \in Z$, $z_1 > z_2$, let's show that the (9) property is executed, i.e.:

$$P(Z_1/_z Z_2) = P(Z_1)/_{z[x]} P(Z_2)$$

$$P(z_1) = q^n f_1(x) - q^n f_1(\tfrac{p}{q}) + z_1 = q^n a_0 + q^n a_1 x + q^n a_2 x^2 + ... - q^n a_0 -$$

$$-q^{n-1} a_1 p - q^{n-2} a_2 p^2 - \cdots + z_1 = a_1 q^{n-1}(qx - p) + a_2 q^{n-2}(q^2x^2 - p^2) + ... z_1$$

$\frac{P(Z_1)}{P(Z_1)} = \frac{a_1 q^{n-1}(qx-p)+a_2 q^{n-2}(q^2x^2-p^2)+...+z_1}{b_1 q^{n-1}(qx-p)+b_2 q^{n-2}(q^2x^2-p^2)+...+z_2}$

$\frac{P(Z_1)}{P(Z_2)} = c_1 q^{n-1}(qx - p) + c_2 q^{n-2}(q^2x^2 - p^2) + ... + \frac{z_1}{z_2}$

$\frac{P(z_1)}{P(z_2)} = P(\frac{z_1}{z_2})$ for c_i can select $c_0 = \forall$

$c_i = \frac{a_i z_2 - b_i z_1}{z_2 * P[x](z_2)}, i = \overline{1, n}$ $c_i = \frac{a_i z_2 - b_i z_1}{z_2 * P_{[x=p/q]} z_2}$

$\frac{a_1 q^{n-1}(qx-p)+a_2 q^{n-2}(q^2x^2-p^2)+...+z_1}{b_1 q^{n-1}(qx-p)+b_2 q^{n-2}(q^2x^2-p^2)+...+z_2} =$

$\frac{a_1z_2-b_1z_1}{z_2*b_1q^{n-1}(qx-p)+b_2q^{n-2}(q^2x^2-p^2)+...+z_2}*q^{n-1}(qx-p)$

$+\frac{a_2z_2b_2z_1}{z_2*b_1q^{n-1}(qx-p)+b_2q^{n-2}(q^2x^2-p^2)+...+z_2}*q^{n-2}(q^2x^2-p^2)+\frac{z_1}{z_2}$

$a_1q^{n-1}(qx-p)+a_2q^{n-2}(q^2x^2-p^2)+...+z_1=\frac{a_1z_2-b_1z_1}{z_2}*q^{n-1}(qx-p)+$

$+\frac{a_1z_2-b_1z_1}{z_2}*q^{n-2}(q^2x^2-p^2)+...+\frac{z_1}{z_2}*b_2q^{n-2}(q^2x^2-p^2)+...+z_2$

$a_1q^{n-1}(qx-p)+a_2q^{n-2}(q^2x^2-p^2)+...+z_1=q^{n-1}(qx-p)[\frac{a_1z_2-b_1z_1}{z_2}+\frac{z_1}{z_2}*b_1$

$+q^{n-2}(q^2x^2-p^2)[\frac{a_1z_2-b_1z_1}{z_2}+\frac{z_1}{z_2}*b_2]+...+z_1$

$a_1q^{n-1}(qx-p)+a_2q^{n-2}(q^2x^2p^2)+...+z_1=a_1q^{n-1}(qx-p)+a_2q^{n-2}(q^2x^2-p^2)+...+z_1$

$\frac{P(z_1)}{P(z_1)}=P(\frac{z_1}{z_2})$ since the P is homomorphism

.

Proof. let $z_1, z_2 \in Z$ show that the property (3) is executed, i.e.:

$P(z_1 -_z z_2) = P(z_1) -_{z[x]} P(z_2,)$

$P(z_1) -_{z[x]} P(z_2) = (q^n(f_1(x) - q^nf_1(\frac{p}{q}) + z_1) - (q^nf_2(x) - q^nf_2(\frac{p}{q}) + z_2 == q^na_0+$

$q^na_1x+...+q^na_nx^n - q^na_0 - q^{n-1}pa_1 - ... - p^na_n + z_1 - q^nb_0 - q^nb_1x - ... - q^nb_nx^n +$

$q^nb_0 + q^{n-1}pb_1 + ... + p^nb_n - z_2 + (a_1-b_1)(q^nx - q^{n-1}p) + (a_2-b_2)(q^nx^2 - q^{n-2}p^2)+$

$... + (a_n - b_n)(q^nx^n - p^n) + z_1 - z_2.$

Let's show that one part is equal to the other:

$P(z_1 -_z z_2) = (q^nf_1(x) - q^nf_1(\frac{p}{q}) + z_1) - (q^nf_2(x) - q^nf_2(\frac{p}{q}) + z_2 == q^n(a_0 - b_0)+$

$q^na_1x+...+q^na_nx^n - q^na_0 - q^{n-1}pa_1 - ... - p^na_n + z_1 - q^nb_0 - q^nb_1x - ... - q^nb_nx^n+$

$q^nb_0 + q^{n-1}pb_1 + ... + p^nb_n - z_2 = (a_1-b_1)(q^nx - q^{n-1}p) + (a_2-b_2)(q^nx^2 - q^{n-2}p^2)+$

$... + (a_n - b_n)(q^nx^n - p^n) + z_1 - z - 2.$

In S.F. Krendelev's work, homomorphic encryption explored for addition and multiplication operations. This article introduces a method of fully homomorphic encryption that also facilitates subtraction and division operations. Here's an example of how the system operates:

$P(z_1 -_z z_2) = (q^nf_1(x) - q^nf_1(\frac{p}{q}) + z_1) - (q^nf_2(x) - q^nf_2(\frac{p}{q}) + z_2 = q^n(a_0 - b_0)+$

$$q^n(a_1-b_1)x+q^n(a_2-b_2)x^2-q^n(a_0-b_0)-q^{n-1}p(a_1-b_1)-q^{n-2}p^2(a^2-b^2)+z_1-$$

$$z_2=(a_1-b_1)(q^nx-q^{n-1}p)+(a_2-b_2)(q^nx^2-q^{n-2}p^2)+...+(a_n-b_n)(q^nx^n-p^n)+z_1-z_2.$$

It is also clear that the representation of P compares 1 to 1 and 0 to 0, corresponding to the formulas (10) and (11). Therefore, the representation of P is a homomorphism.
In S.F. Krendelev's work, homomorphic encryption explored for addition and multiplication operations. This article introduces a method of fully homomorphic encryption that also facilitates subtraction and division operations.
Here's an example of how the system operates:

Key: $x_0=\frac{15}{3}$.

Enc: $Z\to Z[x]$.

1) $z_1=8, z_2=4$;

2)$f_1(x)=5+3x-8x^2+9x^3, f_2(x)=7-2x+4x^2+5x^3$;

3)$q^nf_1(\frac{p}{q})=3^3*5+3^2*15*3-8*3*3^29*3^3$;

$q^nf_2(\frac{p}{q})=3^3*7-2*3*15+4*3*15^2+15^3*5=19674$;

4) encryptable polynomial for z_1 and z_2:

$g_{z_1}=1029x^3-2058x^2+1372x-1060$, $g_{z_2}=3087x^3+2058x^2-2744x-12744$;

Subtractions and division Homomorphism of the system:

$g_{z_1}+g_{z_2}=-2058x^3-4116x^2+4116x+11684$;

$\frac{g_{z_1}}{g_{z_2}}=\frac{1}{3}+\frac{-2744x^2+2286\frac{2}{3}x+3188}{3087x^3+2058x^2-2744x-12744}$;

Dec:$Z[x]\to Z$.

$g_{z_1}x_0-g_{z_2}x_0=-2058(\frac{11}{7})^3-4116(\frac{11}{7})^2+4116*\frac{11}{7}+11684=2$;

$\frac{g_{z_1}x_0}{g_{z_2}x_0}=\frac{1}{3}+\frac{-2744(\frac{11}{7})^2+2286\frac{2}{3}*\frac{11}{7}+3188}{3087(\frac{11}{7})^3+2058(\frac{11}{7})^2-2744*\frac{11}{7}-12744}=1,4.$

The encryption algorithm is shown as follows: 1) Let the password be "Informatics". 2) A polynomial with coefficients $a_0=8, a_1=-2, a_2=1->8-2x+x^2$ is created 3) The symbols are converted to the ASCII table 4) Then the polynomial to be encrypted is $z_I, z_n, z_f, z_o, z_m, z_a, z_t, z_i, z_c, z_s$ are as follows:

$$g_{z_I}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 84 = 49x^2 - 98x + 84;$$

$$g_{z_n}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 101 = 49x^2 - 98x + 101;$$

$$g_{z_f}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 99 = 49x^2 - 98x + 99;$$

$$g_{z_0}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 104 = 49x^2 - 98x + 104;$$

$$g_{z_r}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 110 = 49x^2 - 98x + 110;$$

$$g_{z_m}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 111 = 49x^2 - 98x + 111;$$

$$g_{z_a}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 108 = 49x^2 - 98x + 108;$$

$$g_{z_t}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 111 = 49x^2 - 98x + 111;$$

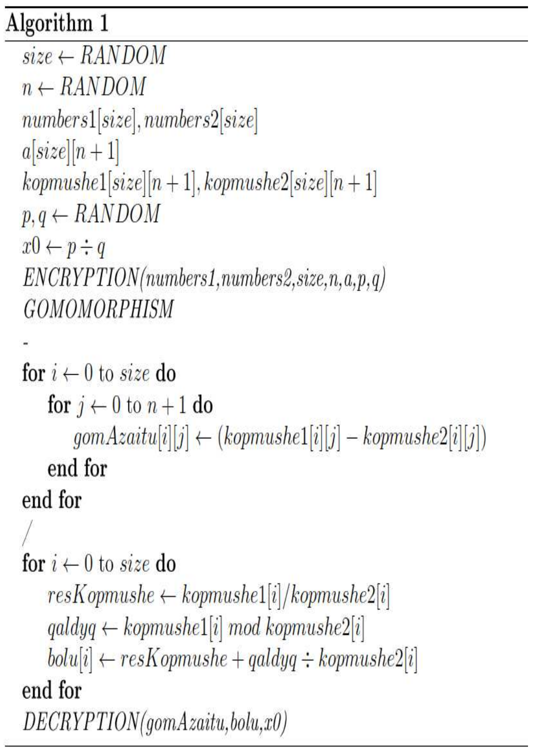
$$g_{z_i}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 103 = 49x^2 - 98x + 103;$$

$$g_{z_c}(x) = 7^2(8-2x+x^2) - (7^2*8+7^{2-1}*5*(-2)+7^0*5^2*(-1)) + 121 = 49x^2 - 98x + 121;$$

Decrypted numbers (84,101,99,104,110,111,108,111,103,121) translated by value to the ASCII table. The initial word "Technology"is obtained.
The pseudo - code of subtraction and division operations added to the complete homomorphic encryption in a ring of polynomials with a variable is shown in Figure 4.

The results and discussion.

In the Arduino IDE integrated development environment, in C++, a static library was created for full homomorphic encryption in a ring of polynomials with a homomorphic modified variable in a ring of polynomials with a variable for different microcontrollers.
The Boost library is a powerful tool for developing software in C++, providing a set of general-purpose and high-performance libraries. One of the most important aspects of the library is its support for working with large numbers, which makes it especially useful in tasks related to cryptography and homomorphic encryption. [17]. In the context of HE, the Boost library performs several key tasks, such as supporting the execution of many operations on encrypted numbers and reducing computational inaccuracies (approximations of values). Homomorphic encryption allows you to perform calculations on encrypted data without decrypting it, which ensures a high level of confidentiality. However, one of the main problems when working with such encryption is the accuracy of calculations and performing complex operations, such as division, on encrypted data. Key tasks solved when implementing the modified library Operations with large numbers require precise control over their representation and manipulation. Standard libraries often have problems with accuracy when operating on large integers. The Boost library provides mechanisms for working with numbers of arbitrary length, which helps to avoid rounding errors and data loss when manipulating large numbers. The library development included support for FHE, which



```
Algorithm 1
  $size \leftarrow RANDOM$
  $n \leftarrow RANDOM$
  $numbers1[size], numbers2[size]$
  $a[size][n+1]$
  $kopmushe1[size][n+1], kopmushe2[size][n+1]$
  $p, q \leftarrow RANDOM$
  $x0 \leftarrow p \div q$
  $ENCRYPTION(numbers1,numbers2,size,n,a,p,q)$
  $GOMOMORPHISM$
  -
  for $i \leftarrow 0$ to $size$ do
      for $j \leftarrow 0$ to $n+1$ do
          $gomAzaitu[i][j] \leftarrow (kopmushe1[i][j] - kopmushe2[i][j])$
      end for
  end for
  /
  for $i \leftarrow 0$ to $size$ do
      $resKopmushe \leftarrow kopmushe1[i]/kopmushe2[i]$
      $qaldyq \leftarrow kopmushe1[i]\ mod\ kopmushe2[i]$
      $bolu[i] \leftarrow resKopmushe + qaldyq \div kopmushe2[i]$
  end for
  $DECRYPTION(gomAzaitu,bolu,x0)$
```

Figure 4: Pseudo-code of the proposed subtraction and division operations

allows performing all arithmetic operations (addition, multiplication, etc.) on encrypted data. Unlike partial homomorphic encryption schemes, FHE allows for full data processing without disclosing its contents. This is especially important for ensuring confidentiality in cloud computing and other secure environments. One of the most difficult tasks in implementing homomorphic encryption is the division operation. Division on encrypted data cannot be performed within the framework of standard homomorphic schemes, since it violates the data structure and leads to a loss of confidentiality. As part of the library modification, a special method was developed that supports homomorphic division, allowing this operation to be performed without disclosing the data. This achievement significantly expands the library's capabilities to handle more complex computational tasks. The modified library, which supports fully homomorphic encryption and operations with large numbers, significantly improves the ability to work with encrypted data, allowing all mathematical operations, including division, to be performed without disclosing the contents. The library's architecture, based on optimized homomorphic division methods and support for highly efficient cryptographic algorithms, makes it a valuable tool for developing secure and confidential systems suitable for use in modern computing environments that require data protection. The architecture of the developed library is illustrated in Figure 5 [18].

The Secret Key class is an important component of the cryptographic library responsible for creating, managing, and using secret keys. Using the standard library's random number generator, which randomizes based on the current time, the Secret Key class efficiently

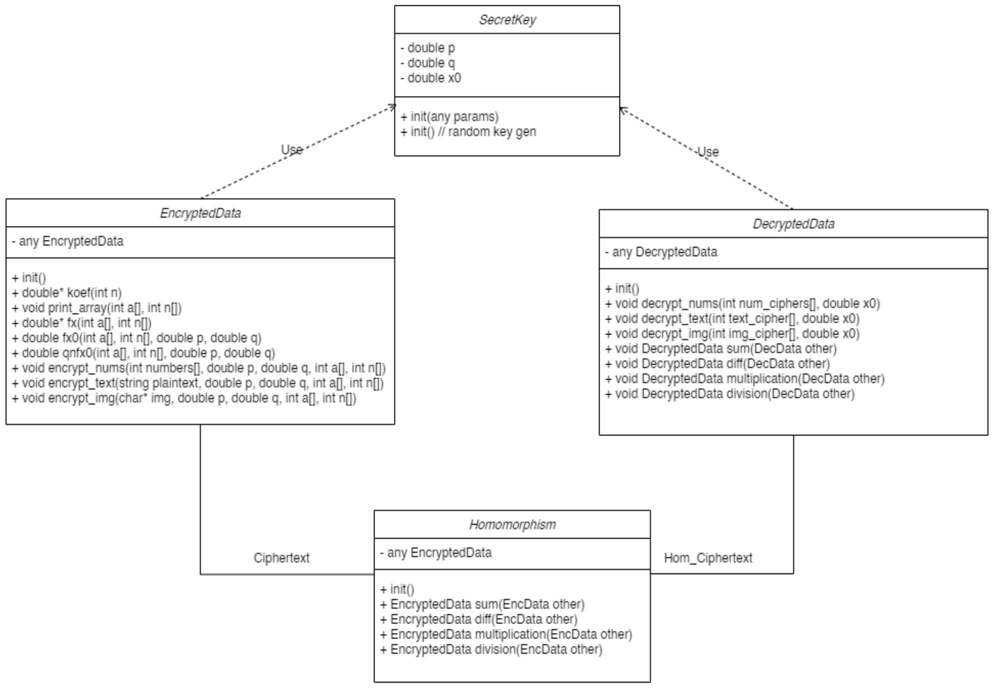

SecretKey
- double p
- double q
- double x0
+ init(any params)
+ init() // random key gen
Use
Use
EncryptedData
- any EncryptedData
+ init()
+ double* koef(int n)
+ void print_array(int a[], int n[])
+ double* fx(int a[], int n[])
+ double fx0(int a[], int n[], double p, double q)
+ double qnfx0(int a[], int n[], double p, double q)
+ void encrypt_nums(int numbers[], double p, double q, int a[], int n[])
+ void encrypt_text(string plaintext, double p, double q, int a[], int n[])
+ void encrypt_img(char* img, double p, double q, int a[], int n[])
DecryptedData
- any DecryptedData
+ init()
+ void decrypt_nums(int num_ciphers[], double x0)
+ void decrypt_text(int text_cipher[], double x0)
+ void decrypt_img(int img_cipher[], double x0)
+ void DecryptedData sum(DecData other)
+ void DecryptedData diff(DecData other)
+ void DecryptedData multiplication(DecData other)
+ void DecryptedData division(DecData other)
Homomorphism
- any EncryptedData
+ init()
+ EncryptedData sum(EncData other)
+ EncryptedData diff(EncData other)
+ EncryptedData multiplication(EncData other)
+ EncryptedData division(EncData other)
Ciphertext
Hom_Ciphertext

Figure 5: Library architecture of complete homomorphic encryption in a ring of polynomials with an advanced variable

generates unique and random keys, as well as polynomial coefficients for cryptographic operations. Despite its simplicity, this implementation is efficient enough for many tasks, but in particularly sensitive applications, more complex random number generation mechanisms may be required to increase the cryptographic strength of the system. The Encrypted Data module is an important part of the cryptographic library designed to work with encrypted data, in particular, with homomorphically encrypted numbers. Homomorphic encryption allows you to perform operations on encrypted data without revealing its content, which opens up many opportunities for secure data processing in areas such as cloud computing, privacy, and financial technology.The main goal of the Encrypted Data module is to support working with encrypted data at different stages of the cryptographic process, from its creation to extracting clear data from ciphertext. In this context, the module implements key operations such as encryption, decryption, and the ability to perform calculations on encrypted numbers using homomorphic encryption. Many modern homomorphic encryption schemes, such as those based on polynomial rings, allow polynomials to be used to represent encrypted data. These schemes support basic mathematical operations on polynomials, which is especially important for full homomorphic encryption. Polynomial addition: When an addition operation is performed on encrypted data, it can be interpreted as adding the corresponding polynomials that represent that data. The result will also be an encrypted polynomial. Polynomial subtraction: Subtraction works similarly to addition, but to calculate the difference between two encrypted values, the corresponding polynomials are subtracted. The result is an encrypted polynomial, which can be decrypted to obtain the difference. Polynomial multiplication: Polynomial multiplication in the context of homomorphic encryption is used to perform multiplication operations on encrypted data. This allows calculations that require multiplication (such as in financial calculations or data processing) to be performed without revealing information about the data itself. Polynomial division: Polynomial division in homomorphic schemes is a more complex operation that requires additional mathematical transformations. This can be implemented through special division protocols that take into account the structures of polynomial rings and their peculiarities when working with encrypted data.Decrypted Data encrypts encrypted data using a secret key and reverse Homomorphism.

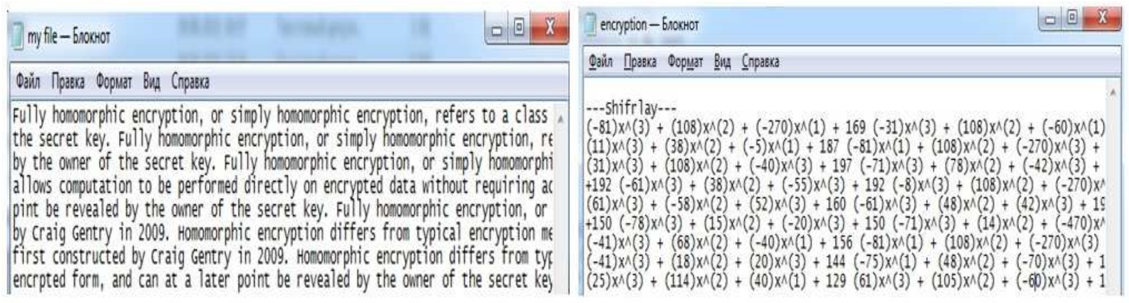
Checking for homomorphism. An array named resBuf is opened to check Homomorphism by the addition operation. Its measure is taken in accordance with which the length of the polynomial is greater. Next, all elements of the resBuf are taken as 0. Then the index polynomial values corresponding to each index are added throughout the cycle. To calculate the sum value, open the variable resu=0 and find resu+=resbuff[0]*x0. The same is done for the rest of the polynomial.

The homomorphism check by subtraction operation is the same as checking the algorithm for adding the numbers of the first polynomial to the resBuf array and subtracting the numbers of the second polynomial from it.

Checking homomorphism by the multiplication operation. The array named Kob will open. Its Dimension is [(first polynomial dimension)*(second polynomial dimension)]. Divided into two columns, in the first column are the coefficients that precede x, and in the second column are the degrees of that X. The Resu array is opened and the same ranks are added to it.

Checking homomorphism by the division operation. The Gorner scheme was used to perform the division operation into encrypted data. When dividing a polynomial by a polynomial, the quotient and the remainder are obtained.

To construct polynomials, the Koef(double n) function is used, where the coefficients are randomly selected. Full homomorphic encryption and reverse encryption in a ring of polynomials with a variable are shown in Figure 6.


my file — Блокнот
Файл Правка Формат Вид Справка
encryption — Блокнот
Файл Правка Формат Вид Справка

```
Fully homomorphic encryption, or simply homomorphic encryption, refers to a class
the secret key. Fully homomorphic encryption, or simply homomorphic encryption, re
by the owner of the secret key. Fully homomorphic encryption, or simply homomorphi
allows computation to be performed directly on encrypted data without requiring ac
pint be revealed by the owner of the secret key. Fully homomorphic encryption, or
by Craig Gentry in 2009. Homomorphic encryption differs from typical encryption me
first constructed by Craig Gentry in 2009. Homomorphic encryption differs from typ
encrpted form, and can at a later point be revealed by the owner of the secret key
```

```
---Shifrlay---
(-81)x^(3) + (108)x^(2) + (-270)x^(1) + 169 (-31)x^(3) + (108)x^(2) + (-60)x^(1)
(11)x^(3) + (38)x^(2) + (-5)x^(1) + 187 (-81)x^(1) + (108)x^(2) + (-270)x^(3) +
(31)x^(3) + (108)x^(2) + (-40)x^(3) + 197 (-71)x^(3) + (78)x^(2) + (-42)x^(3) +
+192 (-61)x^(3) + (38)x^(2) + (-55)x^(3) + 192 (-8)x^(3) + (108)x^(2) + (-270)x^
(61)x^(3) + (-58)x^(2) + (52)x^(3) + 160 (-61)x^(3) + (48)x^(2) + (42)x^(3) + 19
+150 (-78)x^(3) + (15)x^(2) + (-20)x^(3) + 150 (-71)x^(3) + (14)x^(2) + (-470)x^
(-41)x^(3) + (68)x^(2) + (-40)x^(1) + 156 (-81)x^(1) + (108)x^(2) + (-270)x^(3)
(-41)x^(3) + (18)x^(2) + (20)x^(3) + 144 (-75)x^(1) + (48)x^(2) + (-70)x^(3) + 1
(25)x^(3) + (114)x^(2) + (40)x^(1) + 129 (61)x^(3) + (105)x^(2) + (-60)x^(3) + 1
```

Figure 6: Fully homomorphic encryption and reverse encryption in a ring of polynomials with variables

Testing the performance of created libraries on the AtmelAVR (Atmega 328) microcontroller Initially, on the Atmega 328 microcontroller FHE in a ring of polynomials with a modified variable, comparisons were made with different pairs of two-digit numbers:

key generation, encryption, reverse encryption, (addition, subtraction, multiplication, division) times, and the complexity of the algorithm was considered. The result can be seen in Table 4.

Table 4: Execution time of the created library on the Atmega 328 microcontroller

2-digit, different even numbers	Key generation, encryption, addition, reverse encryption, s	Key generation, encryption, subtraction, reverse encryption, s	Key generation, encryption, multiplication, reverse encryption, s	Key generation, encryption, distribution, reverse encryption, s
100	1,95	1,92	2,52	2,44
200	2,625	2,605	3,44	3,33
300	3,736	3,727	4,94	4,77
400	5,032	5,027	6,68	6,67
500	6,931	6,947	9,28	9,2

Compared to the addition operation, the multiplication operation demonstrates worse performance, indicating that its algorithmic complexity is higher. It was also observed that memory operations consume the majority of processor time. This suggests that these schemes may be suitable for specific software applications. TThe polynomial addition operation has linear complexity O(n) and requires proportional increase in memory as the size of the polynomials increases. Although the computational time requirements grow linearly, this does not lead to significant performance issues. The polynomial multiplication operation has quadratic complexity O(n?), which leads to a significant increase in execution time and memory required for computation, especially when working with large polynomials. The quadratic growth of the complexity of polynomial multiplication significantly reduces the performance of cryptographic systems when processing long polynomials, which is an important factor to consider when designing efficient cryptographic algorithms with homomorphic encryption [20].

Homomorphic operations are usually performed between two encrypted texts. If one of the operands can be plain text, it can significantly improve performance. The size of the resulting ciphertext remains the same as the entered ciphertext, and the redrawing step can be skipped. SEAL, Helib, TFHE [21, 22] provide functions for adding, extracting, and multiplying plaintext encrypted text. The library of polynomials with a modified variable presented in the article supports the arithmetic operations of addition, subtraction and multiplication and addition of encrypted text. Modern homomorphic encryption libraries are not designed for microcontrollers. The presented library can be used for microcontroller and computer [23-26].

When adding two polynomials, it turns out that their result does not exceed the powers of the added terms, and when multiplied, it is equal to the sum of the powers of the multipliers. From this we can conclude that the number of sums performed does not negatively affect the performance of the calculation, and multiplication leads to a slowdown in the performance of the calculation. To reduce the growth rate of degrees, it is recommended to create polynomials of minimal degrees as initial encryption polynomials.

3 Conclusion

The HomomorphicControllerVersion01 library represents a major advance in homomorphic encryption for Atmel AVR microcontrollers, ensuring data security in IoT ecosystems. The implementation of the homomorphic division method significantly expands the application of homomorphic encryption in cloud computing, information security, and machine learning. This research has high practical significance, providing effective tools for data protection in the context of limited computing resources and expanding the potential for secure computing in various fields.

References

[1] Goran D., Milan M., Pavle V. Evaluation of Homomorphic Encryption Implementation in IoT Device // Journal of Electrical Systems and Information Technology - 2021. - Vol. 8, № 3. - P. 568-592.

[2] Iliar Ch., Shea Sh. IoT security (internet of things security) https://www.techtarget.com/iotagenda/definition/IoT-security-Internet-of-Things-security. 12.02.2023.

[3] Sujoy S.R., Goyuri P., Deepika N. Interoperability in IoT for Smart Systems// SMIT. - 2021. - Vol.2, Issue 1. - P. 364-402.

[4] Goran D., Milan M., Pavle V. Evaluation of Homomorphic Encryption Implementation in IoT Device // Journal of Electrical Systems and Information Technology - 2021. - Vol. 8, № 3. - P. 568-592.

[5] Brakerski, Z.; Gentry, C.; Vaikuntanathan, V. Fully homomorphic encryption without bootstrapping // In Proceedings of the Conference on Innovations in Theoretical Computer Science (ITCS), - Cambridge: MA USA, 2012; - P. 309 - 325

[6] Iliar C., Hu Wang, Yan Y., Xiaojun T., Qinghua L., Xiangyu Ch. Dynamically Reconfigurable Encryption and Decryption System Design for the Internet of Things Information Security // Sensors (Basel). - 2016. - Vol. 1, № 2. - P. 143 - 182.

[7] Hu Wang, Zhang D.G., Dong D.C., Peng H.T. Research on development of embedded uninterruptable power supply system for IOT-based mobile service, // Computers and Electrical Engineering. - 2017. - Vol. 3, № 6, - P. 1377-1387.

[8] Dong D.C., Peng H.T. Secured Data Transmission in IoT using Homomorphic Encryption // Computers and Electrical Engineering. - 2020. - Vol. 4, № 3, - P. 845-867.

[9] Babenko P.K. Packet symmetric fully homomorphic encryption based on matrix polynomials // Proceedings of the Institute for System Programming of the Russian Academy of Sciences. - 2014. - Vol. 26, No. 5. - P. 99-116.

[10] Mikushin A. Interesting facts about microcontrollers. - M.: BHV-Petersburg, 2016. - 357 p.

[11] Yu.S. Magda Microcontrollers of the 8051 series, a practical approach - Moscow: DMK Press, 2018 - 228 p.

[12] Amgad M., Suliman Mohamed F., Saddam F. Smart health monitoring system using IoT based smart fitness mirror // TELKOMNIKA Telecommunication, Computing, Electronics and Control. - 2020. - Vol. 18, № 3. - P. 317-331.

[13] Korolev N. Atmel: 32-bit Flash microcontrollers on the AVR32 core // Components and Technologies. - 2018. - V. 12, No. 11, - P. 312-347.

[14] Evstifeev A.V. Microcontrollers AVR Mega family. User's guide. - M.: Publishing house Dodeka, 2017. V. 2. - 592 p.

[15] Krendelev S.F. Homomorphic encryption (secure cloud computing) // RusCrypto. Technical sciences. - 2011. No. 2 (96). - P. 93-117.

[16] Farooq M., Waseem M, Khairi A., Mazhar S., A Critical Analysis on the Security Concerns of Internet of Things (loT) //Perception. - 2020. - Vol.11, - P. 11-23.

[17] Smart N.P., Vercauteren F. Fully homomorphic encryption with relatively small key and ciphertext sizes, public Key Cryptography // PKC Springer Berlin Heidelberg, 2010. - Vol. 6056, - P. 420-443.

[18] Temirbekova Zh.E., Pyrkova A.Yu. "Using FHE in a binary ring Encryption and Decryption with BLE Nano kit microcontroller"// E3S Web of Conferences 202 (ICENIS 2020), 2020. - P. 687-712.

[19] Temirbekova Zh.E., Pyrkova A.Yu. Copyright, object name: HomomorphicController version 1.0, №16878, 2021

[20] Kabdrakhova, S.S. On Algorithm of Finding Solutions of Semiperiodical Boundary Value Problem for Systems of Nonlinear Hyperbolic Equations. Springer Proceedings in Mathematics and Statistics, 2017, 216, P. 142–157

[21] Mansurova, M., Belgibayev, B., Zaitin, Y., ... Sarsembayeva, T., Tyulepberdinova, G. Coordinate Robot for Smart Search and Recognition by QR Code of Preparations of a Pharmacy Warehouse. 17th IEEE International Conference on Application of Information and Communication Technologies, AICT 2023 - Proceedings, 2023

[22] Balakayeva, G., Darkenbayev, D., Zhanuzakov, M. DEVELOPMENT OF A SOFTWARE SYSTEM FOR PREDICTING EMPLOYEE RATINGS | ROZWYJ OPROGRAMOWANIA DO PRZEWIDYWANIA OCEN PRACOWNIKYW. Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Srodowiska, 2023, 13(3), P. 121–124

[23] R. K. Uskenbayeva, A. A. Kuandykov, A. A. Kuatbayeva, A. B. Kassymova, G. K. Kuatbayeva and B. K. Zhussipbek Burn disease data analysis model in SAS UE,2021 IEEE 23rd Conference on Business Informatics (CBI), Bolzano, Italy, 2021, pp. 197-201, doi: 10.1109/CBI52690.2021.10072.

[24] Nurtileu Assymkhan and Amandyk Kartbayev Advanced IoT-Enabled Indoor Thermal Comfort Prediction Using SVM and Random Forest Models. International Journal of Advanced Computer Science and Applications(IJACSA), 15(8), 2024. http://dx.doi.org/10.14569/IJACSA.2024.01508102.

[25] Bhimavarapu, U.; Battineni, G.; Chintalapudi, N. Improved Optimization Algorithm in LSTM to Predict Crop Yield. Computers 2023, 12, 10. https://doi.org/10.3390/computers12010010.

[26] Fariza Nussipova, Shynggys Rysbekov, Zukhra Abdiakhmetova, Amandyk Kartbayev Optimizing loss functions for improved energy demand prediction in smart power grids International Journal of Electrical and Computer Engineering (IJECE) Vol. 14, No. 3, June 2024, pp. 3415 3426 ISSN: 2088-8708, DOI: 10.11591/ijece.v14i3. pp. 3415-3426

Information about authors: *Temirbekova Zhanerke Erlanovna – PhD, Associate Professor at Faculty of Information Technology of Al-Farabi Kazakh National University (Almaty, Kazakhstan, e-mail: temyrbekovazhanerke2@gmail.com);*

Abdiakhmetova Zukhra Muratovna (corresponding author) – PhD, Associate Professor at Faculty of Information Technology of Al-Farabi Kazakh National University (Almaty, Kazakhstan, e-mail: zukhra.abdiakhmetova@gmail.com);

Tynymbayev Sakhybay – Candidate of Technical Sciences, Professor of International University of Information Technologies (Almaty, Kazakhstan, e-mail: s.tynym@mail.ru);

Altynbek Serik Atakonysuly – PhD, Vice-Rector for Research and External Relations of the K.Kulazhanov Kazakh University of Technology and Business (Astana, Kazakhstan, e-mail: serik_aa@bk.ru);

Gulzat Ziyatbekova – PhD, Associate Professor Al-Farabi Kazakh National University, Department of Artificial Intelligence and Big Data, Faculty of Information Technology (Almaty, Kazakhstan, e-mail: ziyatbekova1@gmail.com).

Авторлар туралы мәлімет: *Темирбекова Жанерке Ерланқызы – PhD, әл-Фараби атындағы ҚазҰУ ақпараттық технологиялар факультетінің доценті (Алматы, Қазақстан, e-mail: temyrbekovazhanerke2@gmail.com);*

Абдиахметова Зухра Мұратқызы (корреспондент-автор) - PhD, әл-Фараби атындағы ҚазҰУ Ақпараттық технологиялар факультетінің доценті (Алматы, Қазақстан, электрондық пошта: zukhra.abdiakhmetova@gmail.com);

Тынымбаев Сахыбай – техника ғылымдарының кандидаты, Халықаралық ақпараттық технологиялар университетінің профессоры (Алматы, Қазақстан, электрондық пошта: s.tynym@mail.ru);

Алтынбек Серік Атақонысұлы – PhD, Қ.Құлажанов атындағы Қазақ технология және бизнес университетінің ғылыми жұмыстар және сыртқы байланыстар жөніндегі проректоры, (Астана қ., Қазақстан, электрондық пошта: serik_aa@bk.ru);

Гүлзат Зиятбекова – PhD, әл-Фараби атындағы Қазақ ұлттық университеті, ақпараттық технологиялар факультеті, жасанды интеллект және үлкен деректер кафедрасы (Алматы, Қазақстан, электрондық пошта: ziyatbekova1@gmail.com).

Received: September 11, 2024
Accepted: December 22, 2024