Methods of information systems protection

UDC 004.056.55 **doi:** https://doi.org/10.20998/2522-9052.2024.2.11

Vladimir Pevnev[1], Oles Yudin[1], Peter Sedlaček[2], Nina Kuchuk[3]

[1] National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine
[2] University of Žilina, Žilina, Slovakia
[3] National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine

METHOD OF TESTING LARGE NUMBERS FOR PRIMALITY

Abstract. The current stage of scientific and technological development entails ensuring information security across all domains of human activity. Confidential data and wireless channels of remote control systems are particularly sensitive to various types of attacks. In these cases, various encryption systems are most commonly used for information protection, among which large prime numbers are widely utilized. The subject of research involves methods for generating prime numbers, which entail selecting candidates for primality and determining the primality of numbers. **The subject of research** involves methods for generating prime numbers, which choice selecting candidates for primality and determining the primality of numbers. **The objective of the work** is the development and theoretical justification of a method for determining the primality of numbers and providing the results of its testing. The aim to address the following main tasks: analyze the most commonly used and latest algorithms, methods, approaches, and tools for primality testing among large numbers; propose and theoretically justify a method for determining primality for large numbers; and conduct its testing. To achieve this aim, general scientific methods have been applied, including analysis of the subject area and mathematical apparatus, utilization of set theory, number theory, fields theory, as well as experimental design for organizing and conducting experimental research. **The following results have been obtained:** modern methods for selecting candidates for primality testing of large numbers have been analyzed, options for generating large prime numbers have been considered, and the main shortcomings of these methods for practical application of constructed prime numbers have been identified. Methods for determining candidates for primality testing of large numbers and a three-stage method for testing numbers for primality have been proposed and theoretically justified. The testing conducted on the proposed primality determination method has demonstrated the correctness of the theoretical conclusions regarding the feasibility of applying the proposed method to solve the stated problem. **Conclusions.** The use of a candidate primality testing strategy allows for a significant reduction in the number of tested numbers. For numbers of size 200 digits, the tested numbers is reduced to 8.82%. As the size of the tested numbers increases, their quantity will decrease. The proposed method for primality testing is sufficiently simple and effective. The first two stages allow for filtering out all composite numbers except for Carmichael numbers. In the first stage, using the first ten prime numbers filters out over 80 percent of the tested numbers. In the second stage, composite numbers with factors greater than 29 are sieved out. In the third stage, Carmichael numbers are sieved out. The test is polynomial, deterministic, and unconditional.

Keywords: prime numbers; generation of large prime numbers; primality testing method; Carmichael numbers.

Introduction

New information technologies are offering promising opportunities for advancement in science and technology. In recent years, self-driving cars and self-driving vehicles have become a reality, unmanned aerial vehicles capable of staying in the air for many hours and flying thousands of kilometers, and factories that produce highly complex products are able to do it without human involvement in the production cycle.

Unauthorized access capabilities to control systems have sharply increased. For instance, in 2015, a research [1] detailed the entire procedure for cyber intrusion the control system of a Jeep Cherokee. In May 2015, there was a report of hijacking control of an aircraft mid-flight [2]. The capabilities of malicious actors have significantly expanded with the development of artificial intelligence systems. The incorporation of artificial intelligence into cybercrime may lead to the emergence of new types of attacks and evasion methods [3, 4]. Attacks leveraging artificial intelligence may be more sophisticated to detect and defend against, posing a threat to the security of applications and user confidentiality [5, 6].

The simplest way to gain access to the control system of almost any object is to connect to its communication channels. This method encounters almost no barriers, especially when dealing with remote objects. Two types of manipulation on the communication channel can be considered. The first is attempting to intercept control or infiltrate the control system, and the second is disabling or destroying g the communication channel. Infiltrating the control system is possible when there are vulnerabilities in the software. In this case, the security of the software will play a significant role, which is achieved through its testing [7, 8]. In the case of the second type of manipulation, for instance, it is possible to force an unmanned aerial vehicle to make an emergency landing or terminate its mission and return to its base [9].

To counteract potential infiltration into the communication channel, it is imperative to first ensure the availability of said channel. The availability of communication resources encompasses the ability to furnish users with necessary system resources, spatial topology of users, and resilience of communication components amid organized electronic countermeasures [10-12]. To ensure resilient system management, it is essential to guarantee the integrity of command information. Integrity, in this context, refers to the system's capacity to resist unauthorized alterations to information and/or restore distorted information within a specified timeframe using embedded mechanisms. The most prevalent means of ensuring information integrity involve methods of interference-resistant coding and hide transmission fact [13, 14]. These two system properties, addressed concurrently, will facilitate the provision of requisite quality of management information [15, 16]. Encryption of the communication channel may serve as one of the elements capable of secure against information imposition.

Analysis of recent research and publications

Modern encryption systems can be categorized into two big classes: symmetric and asymmetric systems. The principles of their construction underlie their differences. While the cryptographic strong of symmetric systems relies on the secrecy of keys used for encryption and decryption, for asymmetric systems, it is the complexity of the problem upon which the system is based [17, 18]. Utilizing encryption mechanisms implies the ability to exchange confidential information over open communication channels, including radio channels.

The most widely used encryption system, which is also used for digital signatures, is the RSA (Rivest-Shamir-Adleman) asymmetric encryption system. In the scientific realm, asymmetric encryption systems are also referred to as public-key cryptography.

At the core of this system are two large prime numbers. The number obtained by multiplying these prime numbers forms the basis of the RSA system. In number theory, the problem of finding and proving the primality of a number is one of the key challenges. The complexity of finding a prime number is due to the fact that, firstly, there is no known law governing their distribution along the number line, and secondly, prime numbers form an infinite set.

To construct large prime numbers, a method described in many sources is utilized [19, 20]. The essence of the method is as follows. A sequence of prime numbers $p_1 < p_2 < p_3 < ...$ is constructed until a prime number of the required size is found. The initial prime odd number p1 is chosen arbitrarily. After prime number p_{i-1} has been constructed, a random number r, $1 \le r \le p_{i-1} - 1$, is selected. Let $r = 2^s \times t$, where t is odd. Then the candidate for the next prime number is p_i:

$$n = 2 \times r_{pi} - 1 + 1 = 2^{s+1} \times p_{i-1} t + 1 .$$

Next, n is checked for primality using known methods. The drawback of such an approach is evident: the probability of guessing a prime number with large numbers (> 200 digits) is too small. Another group of methods [21, 22] is based on selecting an arithmetic sequence or sum of prime number addends with unity.

One of the most well-known methods for finding prime numbers in this group is using Mersenne numbers. Using this method, as of June 2023, the largest known prime number was discovered, which is represented as $2^{82589933-1}$ [23]. This number contains 24,862,048 decimal digits and spans over 5,583 pages of continuous text (in the journal format). It may seem that the task of finding prime numbers is solved, at least for practical purposes. However, there are very few such numbers, and the presented number is only the 51th Mersenne number.

The main disadvantage of such methods for obtaining predicted prime numbers is that they are quite easy to replicate. If the prime numbers obtained in this way are used as keys in encryption systems, it becomes possible to construct a pool of keys that are most commonly used by users [24]. This leads to their compromise quite rapidly. Information security experts have already openly declared this problem [25].

Despite the relevance of the problem of constructing large prime numbers and numerous algorithms for their construction, or more precisely, algorithms for determining primality, the greatest achievement of recent decades is the Agrawal-Kayal-Saxena test, proposed by Indian researchers to the mathematical community in 2004 [26]. The test has purely theoretical significance due to its high complexity $O(\log^{21/2} n)$. Subsequently, this estimate was improved to $O(\log^6 n)$ [27].

The purpose of the work is to develop and theoretically justify a method for determining the primality of numbers and providing the results of its testing.

Method for testing arbitrary numbers for primality

Before proposing a method for testing numbers for primality, the concept of a pseudoprime number should be introduced. A pseudoprime number N is defined as a number whose primality has not been proven. If it is proven to be composite, it is removed from the list of pseudoprime numbers [28]. During the primality testing process, a three-stage approach can be utilized. In the first stage, using the proposed method for generating pseudoprime numbers [28], the product of prime numbers is computed, starting from 2 until the product becomes close to the number that under analysis.

In practice, the number of prime numbers in the multiplication can be significantly larger. The maximum prime number in the multiplication should ideally not exceed the square root of the number being tested. The more numbers involved, the higher the probability of obtaining a multiplier in the number under investigation. However, it is important not to go overboard. If all prime numbers less than a thousand are multiplied, the result will be a number with 416 decimal digits, and there are still 78,408 prime numbers remaining in the first million [29]. Therefore, for computing the Greatest Common Divisor (GCD) as the product of prime numbers, the number $p_{10}\# = 6469693230$ was chosen. Using the first ten prime numbers allows filtering out over 82% of the numbers in the numerical range. Of course, the primorial of ten for determining the primality of numbers with two hundred or more decimal digits is a very small number, for such numbers, it is necessary to take a primorial of at least 93 prime numbers ($p_{93}\# = 5.0956E+200$, with the maximum prime number being 487). Clearly, demonstrating the work with such numbers is impractical.

Afterwards, the GCD between the investigated number and the obtained multiplication is computed. If the GCD is greater than one, then the investigated number is composite. If the GCD equals one, then the process proceeds to the second stage. For example, the numbers listed in Table 1 are examined.

Table 1 – **Results of using the first stage**

The number being testees	$p_{10}\#$	GCD	Type of number	Note
410041	6469693230	1	pseudoprime number	
101101		1001	composite number	Carmichael number
223092907		1	pseudoprime number	
223093339		7	composite number	

It should be noted that one of the Carmichael numbers was eliminated, which is a major challenge for determining the primality of numbers.

In the second stage, the RSA algorithm [29] is used to test the number for primality. The RSA algorithm utilizes an interesting mathematical property of numbers called the multiplicative inverse. The uniqueness of a pair of direct K and inverse K^{-1} numbers is its uniqueness. Uniqueness means that neither of these numbers appears in any other pair with a fixed Euler function, and their product equals one in the $Z\ (p-1)$ ring.

Unlike the RSA algorithm, we use a number that is tested for primality, and we are not interested in the intermediate result of the message computation encrypted with the public key.

Under these conditions, the proposed algorithm will take the following form.

1. A prime number P is selected for primality testing.
2. The Euler's totient function $\varphi(P)$ of the prime number P is computed.

$$\varphi(P) = P - 1 .$$

3. A number K is arbitrarily chosen while adhering to the conditions.

$$K < \varphi(P)\,,\quad GCD(K, \varphi(P)) = 1 .$$

4. The inverse of the number K, denoted as K^{-1}, is computed.
5. The correctness of the pair selection is verified.

$$K \times K^{-1} \equiv 1 \bmod \varphi(P) .$$

6. The number C_1 is computed using an arbitrary number C as follows:

$$C_1 = C^{K \times K-1} \bmod N .$$

7. If $C_1 \neq C$, then the number P being checked is composite; otherwise, the number will be pseudoprime number.

If after the completion of the proposed algorithm it is determined that the number being tested is composite, then the task is solved. If the tested number is identified as pseudoprime number, there could be two possibilities. The first possibility is that the number is prime. The second possibility is that the number is a Carmichael number, meaning it is composite. As an example, the numbers listed in Table 2 will be considered.

Table 2 – **Results of the second stage utilization**

Testing number	K	K^{-1}	$K*K^{-1}$	Result	Type of number
101101	17	4153	70601	42523	composite number
	23	13187	303301	10	pseudoprime number
410041	41	10001	410041	10	pseudoprime number
223092907	25	35694865	892371625	10	pseudoprime number
223093339	43	31129303	223093339	220637944	composite number

As seen from Table 2, the Carmichael number 101101 was again identified as composite with key K = 17. However, with a key equal to 23, the result came out as prime number.

From this, it can be concluded that the final outcome will depend on the size of the key.

The second and third numbers again confirmed their pseudoprime number. The fourth number turned out to be composite again.

The third stage involves the process of raising the encrypted message to the power of the private key. A hypothesis was formulated: the primality of a number can be determined by considering the number of steps required to obtain the desired result. To test this hypothesis, let's take the number 101101. This number is a Carmichael number, so according to the results of the second stage, the number was identified as pseudoprime number. Let's calculate the public and private keys for this number in the Table 3.

In the table, we will introduce the following notation:

– Pub Key – public key;
– Priv Key – private key;
– NC – the number of correct solutions during calculations;
– MSPC – the minimum size of a personal computer at which a correct solution is obtained;
– 2, 3, 14, 99, 150 – numbers on which exponentiation operations with the public key (Pub Key) and private key (Priv Key) were performed;
– Step – cycle size.

The prime numbers 2, 3, 5 were not used as the public key because they are not coprime with the number 101100.

As the numbers for testing the proposed hypothesis, numbers from 2 to 150 were selected. Accordingly, each of these numbers needs to be raised to the power of the public key modulo the number 101101, which is being tested for primality.

Table 3 – **Calculation result for number 101101**

Pub Key		7	11	13	17	19	23	Крок
Priv Key		14443	9191	7777	95153	95779	13187	
2	NC	12	8	7	89	92	13	300
	MSPC	343	791	1177	1253	679	287	
3	NC	12	8	7	89	92	13	300
	MSPC	343	791	1177	1253	679	287	
14	NC	416	265	224	2738	2757	380	10
	MSPC	13	11	7	13	39	7	
99	NC	20	25	21	276	275	38	100
	MSPC	143	91	77	253	679	187	
150	NC	12	8	7	89	92	13	300
	MSPC	343	791	1177	1253	679	287	

The next operation involved raising the obtained number to the power of the private key. During the calculations, it was recorded how many computation results match the chosen number for hypothesis testing. The number of steps after which the respective number is obtained is used as one of the operands in the GCD operation, with the other operand being the number 101101.

If the result of the operation is greater than one, then the number being tested is composite. Otherwise, the GCD determination operation is repeated until one of the operands becomes the private key. If the GCD becomes equal to one during this process, then the number being tested is prime.

As an example, let's consider the numbers listed in Table 4.

The term "current private key" (CPK) refers to the key size at which the selected number for computation is obtained.

Table 4 – **Results of the Third Stage Utilization**

Testing number (A)	K	$10^K \bmod A$	CPK	GCD	Result
101101	11	90991	11	11	composite number
410041	41	10	41	41	composite number
223092907	25	7499520	35694865	1	prime number

In this case, as in all previous ones, the number 10 is chosen for this purpose.

Let's consider the first row. For the number 101101, an open key equal to 11 is chosen. The number 10 is raised to the power of 11 modulo 101101. The result is 90991. This number is then raised to the power of 2, 3, and so on, until the resulting number becomes 10. This occurs on the eleventh step. We seek the GCD for the number 101101, which is being checked for primality, and the number of steps, which is 11, required to obtain the number 10. GCD (101101, 11) = 11. Conclusion: the number 101101 is composite.

Similar actions are performed for the other two numbers. The only observation to make concerns the excess calculations for the third number. If no value equal to 10 is found among the values up to the current private key, which equals half of the private key (see Table 2), the computations can be terminated.

For a more comprehensive verification of the proposed hypothesis, a large number of experiments were conducted, in which the correctness of the hypothesis was tested on various Carmichael numbers, keys, and different arbitrary numbers. Table 5 presents a portion of the experiments for the number 29341 with open keys K = 7, 11, 13, 17, 19, 23, 29, 31, 37, 41 for the number 2.

Let's consider the data presented in Table 5.

Particular interest, in terms of the complexity of the algorithm provided, is the number of steps required to solve the problem.

Table 5 – **The determination of the primality of the number 29341**

Pub Key	7		11		13		17		19		23		29		31		37		41	
Priv Key	8383		18671		2257		15533		21619		3827		11129		10411		793		12881	
MSPC	103		131		97		53		19		47		149		151		73		101	
Step	180		180		180		180		180		180		180		180		180		180	
	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD	Checkpoint	GCD
Results of experiment	103	1	131	1	97	1	53	1	19	1	47	1	149	1	151	1	73	1	101	1
	283	1	311	1	277	1	233	1	199	1	227	1	329	1	331	1	253	1	281	1
	463	1	491	1	457	1	413	1	379	1	407	37	509	1	511	1	433	1	461	1
	643	1	671	61	637	13	593	1	559	13	587	1	689	13	691	1	613	1	641	1
	823	1	851	37	817	1	773	1	739	1	767	13	869	1	871	13	793	793	821	1
	1003	1	1031	1	997	1	953	1	919	1	947	1	1049	1	1051	1	973	1	1001	13
	1183	13	1211	1	1177	1	1133	1	1099	1	1127	1	1229	1	1231	1	1153	1	1181	1
	1363	1	1391	13	1357	1	1313	13	1279	1	1307	1	1409	1	1411	1	1333	1	1361	1

These data are located in the table under the column titled “Checkpoint”. The maximum value for finding the solution was achieved when using Pub Key = 17 and corresponded to 1313 steps. In reality, this number amounts to 188 steps (excluding the exponentiation of the number to Pub Key). To determine the step size, we raise the number to a power until we obtain a result corresponding to the number obtained when 2 is raised to the power of Pub Key. The resulting number will correspond to the step size. Simultaneously, the result of stepwise exponentiation to the power of the private key is checked. As soon as the result becomes comparable to the original number, the number of steps is recorded. This number corresponds to MSPC, which is the first checkpoint. The next checkpoint is the current value of the private key, calculated as the sum of the MSPC and the step size. By incrementally increasing the step size, subsequent checkpoint values are determined. Thus, it is possible to calculate the number of steps required to determine the primality of the number.

Table 6 presents the results of experimental studies with the number 29341. The term “current private key” (CPK) refers to the minimum key size at which the GCD is greater than 1.

Table 6 – **Results of experiments with the number 29341**

Selected number	Pub Key	7	11	13	17	19	23	29	31	37	41
	Priv Key	8383	18671	2257	15533	21619	3827	11129	10411	793	12881
2	MSPC	103	131	97	53	19	47	149	151	73	101
	Step	180	180	180	180	180	180	180	180	180	180
	CPK	1183	671	637	1313	559	407	689	871	793	1001
	GCD	13	61	13	13	13	37	13	13	793	13
3	MSPC	13	41	7	53	19	47	59	61	73	11
	Step	90	90	90	90	90	90	90	90	90	90
	CPK	13	221	637	143	559	407	689	871	793	1001
	GCD	13	13	13	13	13	37	13	13	793	13
4	MSPC	13	41	7	53	19	47	59	61	73	101
	Step	90	90	90	90	90	90	90	90	90	90
	CPK	13	221	637	143	559	407	689	61	793	1001
	GCD	13	13	13	13	13	37	13	61	793	13
5	MSPC	103	131	97	53	19	47	149	151	73	101
	Step	180	180	180	180	180	180	180	180	180	180
	CPK	1183	671	637	1313	559	407	689	871	793	1001
	GCD	13	61	13	13	13	37	13	13	793	13
6	MSPC	43	11	37	53	19	47	29	31	13	41
	Step	60	60	60	60	60	60	60	60	60	60
	CPK	403	611	37	533	259	407	629	91	13	221
	GCD	13	13	37	13	37	37	37	13	13	13
7	MSPC	103	131	97	53	19	47	149	151	73	101
	Step	180	180	180	180	180	180	180	180	180	180
	CPK	1183	671	637	1313	559	407	689	871	793	1001
	GCD	13	61	13	13	13	37	13	13	793	13
8	MSPC	43	11	37	53	19	47	29	31	13	41
	Step	60	60	60	60	60	60	60	60	60	60
	CPK	403	611	37	533	259	407	629	91	13	221
	GCD	13	13	37	13	37	37	37	13	13	13

When considering the results of the conducted experiment, it should be noted that it is impossible to predict the step size, which is essential for determining the GCD. Among the seven presented result groups, a step size of 180 is present in three groups, while step sizes of 90 and 60 appear twice in two groups. The best result was achieved when using numbers 3 and 4 with Pub Key = 7. Identification of the investigated number as prime occurred on the 13th step. The largest number of steps was required in the case of a step size of 180 and Pub Key = 7, amounting to 183 steps. In the case of investigating a prime number, the number of steps will be no more than the sum of the step size and half of Pub Key, divided by the step size. If, in this case, all comparisons show 1, it can be concluded that the investigated number is prime.

Here is a brief overview of the proposed method for checking the primality of an arbitrary number *A*.

First stage: Check number *A* for the presence of small divisors. If number *A* is divisible by at least one divisor, then it is composite. At this stage, the GCD can be determined.

Second stage: Assume that the number *A* being tested is prime. Based on this assumption, we compute the direct *K* and the inverse K^{-1} of *A*. The verification is carried out by raising an arbitrary number D to the power of $K^* K^{-1}$ modulo *A*. If the result of the computation is not equal to *D*, then number *A* is composite.

Third stage: At this stage, the GCD calculation is performed between the current inverse number $K_{current\ inverse}$ and the number *A*. The current inverse number $K_{current\ inverse}$ ranges from 1 to K^{-1}. If the GCD value is greater than one, then number *A* is composite. Otherwise, number *A* is prime.

The strategy for selecting candidates for prime numbers

It is important to take into account that specified experiment did not take an action on selecting potential candidates for prime numbers. As mentioned earlier, there is no specific strategy for choosing such candidates. The following theorems were formulated and proven in the research [28].

Theorem 1. The sum (difference) of the members of two sets of prime numbers that do not intersect is a coprime number with each of the elements of these sets.

Theorem 2. Within the interval between $p_n\#+1$ and the next prime number, all numbers will be composite except those obtained by addition, where the maximum prime number added must be less than the square of the first added prime number.

Following these theorems allows for the construction of a sufficiently large pool of potential prime number candidates. Let's consider the example presented earlier. If we examine the primorial of the first 93 prime numbers, it becomes quite straightforward to compute the first candidate for a prime number. This would be the number equal to $p_{93}\#+1$. More precisely, these are two numbers, and the second one can be correctly expressed as per Theorem 1, $p_{93}\#-1$. Given that the maximum prime number in the presented primorial is 487, the next number, which will be a candidate for primality, is calculated similarly to the first one, by adding to the primorial the next prime number after 487 will be $p_{93}\#+491$. By adding the next prime numbers up to 487^2 -1, we obtain 20904 candidates for primality. Thus, in the interval from $p_{93}\#+1$ to $p_{93}\#+273168$, we need to check less than 8.82% of the numbers. Similarly, we can construct the second interval from $p_{93}\#-1$ to $p_{93}\#-273168$. This way, it is possible to build a pool of composite numbers with a capacity of over half a million.

Moreover, all of them are guaranteed not to be divisible by 94 prime numbers.

If necessary, to expand the pool, one can utilize squares and higher powers of added prime numbers and their products.

For the construction of all possible prime numbers over large intervals, it is possible to multiply the primorial by all natural numbers from 2 to 490. In this case, one can make use of prime and computed numbers that have been utilized in previous iterations.

Conclusions and prospects for further development

The study proposes a method for testing numbers for primality. The idea of this method is based on the use of forward and reverse numbers. As known, these numbers are used for building the RSA system. The proposed method consists of three stages.

The first stage check for the presence of small factors in the number being tested for primality. At this stage, using the first ten prime numbers eliminates more than 80% of the numbers being tested. This stage excludes Carmichael numbers and other composite numbers that have small factors.

The second stage involves computing a random number raised to the power of the multiplication of forward and reverse numbers taken modulo the number being tested. At this stage, all composite numbers except for the remaining Carmichael numbers after the first stage are filtered out.

The third stage involves raising the number being tested to the power of the private key. A hypothesis was proposed: primality of a number can be determined by considering the number of steps required to obtain the desired result.

To test this hypothesis, the number 101101 was used. This number is a Carmichael number, meaning that according to the results of the second stage, the number is identified as a potential pseudoprime number. The verification was carried out for numbers from 2 to 150 using public keys (forward numbers) ranging in prime numbers from 7 to 23, excluding numbers 2, 3, and 5. The hypothesis was confirmed for the proposed number. For two other numbers that were tested, the results of the computations indicate that 410041 (a Carmichael number) is composite, while 223 092 907 is prime number.

For a more comprehensive verification of the proposed hypothesis, a large number of experiments were conducted, testing the correctness of the hypothesis on various Carmichael numbers, keys, and arbitrary numbers. The study presents generalized results of testing the number 29341 with different public keys and numbers from 2 to 8. The first ten numbers coprime to 29340 were chosen as public keys. In each implementation, it was established that the number under testing is composite.

According to the authors, the proposed method for testing numbers for primality is sufficiently simple and effective. The first two stages allow filtering out all composite numbers except for Carmichael numbers. However, Carmichael numbers consist of at least three factors, and the sizes of these factors vary. Since testing process assumes the primality of numbers, if this assumption is correct, the GCD will be equal to one in all cases. The test is polynomial, deterministic, and unconditional.

Using a strategy for selecting prime number candidates significantly reduces the number of tested numbers. For numbers of size 200 digits, the number of tested numbers decreases to 8.82%. The strategy proposed in the study offers a relatively simple solution to this problem. It should be noted that as the size of the tested numbers increases, their quantity will decrease.

The use of factorization algorithms to determine primality is a sufficiently but poorly studied area. When a number has more than two factors, it will be possible to search not only for individual factors, but more precisely for at least one of them, as well as for all possible combinations.

This simplifies the search task, significantly reducing the number of iterations.

References

1. Miller, C. and Valasek, C. (2015), *Remote Exploitation of an Unaltered Passenger Vehicle*, available at: https://illmatics.com/Remote%20Car%20Hacking.pdf
2. Tweney, D. (2015), *FBI says this hacker took over a plane through its in-flight entertainment system*, VentureBeat, available at: https://venturebeat.com/security/fbi-says-this-hacker-took-over-a-plane-through-its-in-flight-entertainment-system
3. Herping, S. (2019), "Securing Artificial Intelligence", *Stiftung Neue Verantwortung*, available at: https://www.stiftung-nv.de/de/publikation/securing-artificial-intelligence
4. Mozhaev, O., Kuchuk, H., Kuchuk, N., Mykhailo, M. and Lohvynenko, M. (2017), "Multiservice network security metric", *2nd International Conference on Advanced Information and Communication Technologies*, AICT 2017 – Proceedings, pp. 133–136, doi: https://doi.org/10.1109/AIACT.2017.8020083
5. Pupillo, L., Fantin, S., Ferreira, A. and Polito, C. (2021), *Artificial Intelligence and Cybersecurity*, CEPS Task Force Report, available at: https://www.ceps.eu/wp-content/uploads/2021/05/CEPS-TFR-Artificial-Intelligence-and-Cybersecurity.pdf
6. Yaloveha, V., Podorozhniak, A. and Kuchuk, H. (2022), "Convolutional neural network hyperparameter optimization applied to land cover classification", *Radioelectronic and Computer Systems*, No. 1(2022), pp. 115–128, DOI: https://doi.org/10.32620/reks.2022.1.09
7. Cao, Weiling, Kosenko, V. and Semenov, S. (2022), "Study of the efficiency of the software security improving method and substantiation of practical recommendations for its use", *Innovative Technologies and Scientific Solutions for Industries*, No. 1(19), pp. 55–64, doi: https://doi.org/10.30837/ITSSI.2022.19.055
8. Semenov, S., Sira, O. and Kuchuk, N. (2018), "Development of graphicanalytical models for the software security testing algorithm", *Eastern-European Journal of Enterprise Technologies*, Vol 2, No 4 (92), pp. 39-46, doi: https://doi.org/10.15587/1729-4061.2018.127210
9. (2021), *OSCE SMM Spot Report 8/2021: Forced emergency landing of long-range unmanned aerial vehicle due to dual GPS signal interference*, Organization for Security and Co-operation in Europe, available at: https://www.osce.org/special-monitoring-mission-to-ukraine/483149
10. Knyazev, V., Lazurenko, B. and Serkov, A. (2022), "Methods and Tools for Assessing the Level of Noise Immunity of Wireless Communication Channels", *Innovative Technologies and Scientific Solutions for Industries*, No. 1 (19), pp. 92–98, doi: https://doi.org/10.30837/ITSSI.2022.19.092
11. Merlac, V., Smatkov, S., Kuchuk, N. and Nechausov, A. (2018), "Resourses Distribution Method of University e-learning on the Hypercovergent platform", *Conf. Proc. of 2018 IEEE 9th Int. Conf. on Dependable Systems, Service and Technologies. DESSERT'2018*, Kyiv, May 24-27, pp. 136–140, doi: http://dx.doi.org/10.1109/DESSERT.2018.8409114
12. Petrovska, I., Kuchuk, H., Kuchuk, N., Mozhaiev, O., Pochebut, M. and Onishchenko, Yu. (2023), "Sequential Series-Based Prediction Model in Adaptive Cloud Resource Allocation for Data Processing and Security", *2023 13th International Conference on Dependable Systems, Services and Technologies, DESSERT 2023*, 13–15 October, Athens, Greece, code 197136, doi: http://dx.doi.org/10.1109/DESSERT61349.2023.10416496
13. Pevnev, V., Frolov, A., Tsuranov, M. and Zemlianko, H. (2022), "Ensuring the Data Integrity in Infocommunication Systems", *International Journal of Computing*, vol. 21(2), pp. 228–233, doi: https://doi.org/10.47839/ijc.21.2.2591
14. Kovalenko, A. and Kuchuk, H. (2022), "Methods to Manage Data in Self-healing Systems", Studies in Systems, Decision and Control, Vol. 425, pp. 113–171, doi: https://doi.org/10.1007/978-3-030-96546-4_3
15. Trubchaninova, K., Serkov, A., Tkachenko, V., Kharchenko, V., Pevnev, V. and Doukas, N. (2020), "Method of Increasing Security of Spatial Intelligence in the Industrial Internet of Things Systems", *24th International Conference on Circuits, Systems, Communications and Computers* (CSCC'2020), Platanias Chania Grete Island, Greece, July 19-22, 2020. pp. 283–289, doi: https://doi.org/10.1109/CSCC49995.2020.00058
16. Kovalenko, A. and Kuchuk, H. (2022), "Methods to Manage Data in Self-healing Systems*", Studies in Systems, Decision and Control,* vol. 425, pp. 113–171, doi: https://doi.org/10.1007/978-3-030-96546-4_3
17. Diffie, W. and Hellman, M. E. (1976), "New Directions in Cryptography", *IEEE Transactions On Information Theory*, vol. 22(6), pp. 644–654, doi: https://doi.org/10.1109/TIT.1976.1055638
18. Rivest, R.L., Shamir, A. and Adleman, L. (1978), "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Communications of the ACM*, Vol. 21(2), pp. 120–126, doi: https://doi.org/10.1145/359340.359342
19. Dixon, J. D. (1984), "Factorization and Primality Tests", *The American Mathematical Monthly*, 3vol. 91, no. 6 , pp. 333–352, doi: https://doi.org/10.1080/00029890.1984.11971425
20. Couvreur, C. and Quisquater, J. J. (1982), "An Introduction to Fast Generation of Large Primes Numbes", *Philips Journal of Research*, vol. 37, pp. 231–264.
21. Brillhart, J., Lehmer, D. H. and Selfridge, J. L. (1975), "New Primality Criteria and Factorizations of $2^m \pm 1$",*Mathematics of Computation*, vol. 29, no. 130, pp. 620–647, doi: https://doi.org/10.2307/2005583
22. Plaisted, D. A. (1979), "Fast Verification, Testing and Generation of Large Primes", *Theoretical Computer Science*, vol. 9, pp. 1–16, doi: https://doi.org/10.1016/0304-3975(79)90002-1
23. (2018), *Largest Known Prime Number Has Almost 25 Million Digits*, SCI News, News Staff, available at: https://www.sci.news/othersciences/mathematics/largest-prime-number-06751.html
24. Crandall, R. and Pomerance, C. (2005), *Prime Numbers. A computational perspective*, Springer, New York, 598 p., available at: http://thales.doa.fmph.uniba.sk/macaj/skola/teoriapoli/primes.pdf
25. Mimoso, M. (2015), *Prime Diffie-Hellman Weakness May Be Key to Breaking Crypto*, Threat post, available at: https://threatpost.com/prime-diffie-hellman-weakness-may-be-key-to-breaking-crypto/115069

26. Agrawal, M., Kayal, N. and Saxena, N. (2004), "PRIMES is in P", *Annals of Mathematics*, vol. 160, no. 2, pp. 781–793, doi: https://doi.org/10.4007/annals.2004.160.781
27. Lenstra, Jr. H. W. and Pomerance, C. (2019), "Primality testing with Gaussian periods", *Journal of the European Mathematical Society*, vol. 21, no. 4, pp. 1229–1269, doi: https://doi.org/10.4171/jems/861
28. Pevnev, V. (2017), "Pseudoprime Numbers: Basic Concepts and the Problem of Security", *ICT in Education, Research and Industrial Applications: Integration, Harmonization and Knowledge Transfer*, Proc. of 13th Int. Conf. Kyiv, Ukraine, May 15-18. 2017. Kyiv. 2017. pp. 583-593, available at: https://ceur-ws.org/Vol-1844/10000583.pdf
29. Pevnev, V. (2018), "Investigation of the algorithm for the numbers primality determining", *Dependable Systems, Services and Technologies: Proc. 9th IEEE Int. Conf.*, Kyiv, pp. 243–247, doi: https://doi.org/10.1109/DESSERT.2018.8409137

Надійшла (received) 15.02.2024
Прийнята до друку (accepted for publication) 22.05.2024

Відомості про авторів/ About the Authors

Пєвнєв Володимир Яковлевич — доктор технічних наук, доцент, професор кафедри комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.
Vladimir Pevnev – Doctor of Sciences (Engineering), Associate Professor, Professor at the Department of Computer Systems, Networks and Cybersecurity, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine.
e-mail: v.pevnev@csn.khai.edu; ORCID ID: https://orcid.org/0000-0002-3949-3514;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57194525720.

Юдін Олесь Вікторович – аспірант кафедри комп'ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.
Oles Yudin – PhD student of the Department of Computer Systems, Networks and Cybersecurity, National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine.
e-mail: o.yudin@csn.khai.edu; ORCID ID: https://orcid.org/0009-0006-8079-0488;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=58882520600.

Седлачек Петер – доктор філософії (комп'ютерні науки), доцент кафедри інформатики, факультет управлінських наук та інформатики, Жилінський університет, Жиліна, Словаччина;
Peter Sedlaček – PhD in Computer Sciences, Assistant Professor of the Department of Informatics, Faculty of Management Science and Informatics, University of Žilina, Žilina, Slovakia;
e-mail: Peter.Sedlacek@fri.uniza.sk; ORCID ID: https://orcid.org/0000-0002-7481-6905;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57194525720.

Кучук Ніна Георгіївна – доктор технічних наук, професор, професор кафедри обчислювальної техніки та програмування, Національний технічний університет "Харківський політехнічний інститут", Харків, Україна;
Nina Kuchuk – Doctor of Technical Sciences, Professor, Professor of Computer Engineering and Programming Department, National Technical University "Kharkiv Polytechnic Institute", Kharkiv, Ukraine;
e-mail: nina_kuchuk@ukr.net; ORCID ID: http://orcid.org/0000-0002-0784-1465;
Scopus ID: https://www.scopus.com/authid/detail.uri?authorId=57196006131.

Методика перевірки великих чисел на простоту

В. Я. Пєвнєв, О. В. Юдін, П. Седлачек, Н. Г. Кучук

Анотація. Сучасний етап розвитку науки та техніки передбачає забезпечення інформаційної безпеки у всіх сферах людської діяльності. Найбільш чутливими до різноманітних атак є конфіденційні дані та бездротові канали систем дистанційного управління. Для захисту інформації у цих випадках найчастіше використовуються різноманітні системи шифрування, серед яких широко використовуються прості числа великої розмірності. **Предметом досліджень** є методи побудови простих чисел, які полягають у виборі кандидатів на простоту та визначенні простоти чисел. **Метою роботи** є розробка та теоретичне обґрунтування методу визначення простоти чисел і надання результатів його тестування. У статті передбачається вирішити такі основні завдання: проаналізувати найбільш уживані та найновіші алгоритми, методи, підходи та засоби визначення кандидатів на простоту серед чисел великої розмірності, запропонувати та теоретично обґрунтувати метод визначення простоти для великих чисел, провести його тестування. Для досягнення мети застосовано загальнонаукові методи: аналіз предметної області та математичний апарат, використано теорії множин, чисел та полів, планування експерименту для організації та проведення експериментальних досліджень. **Здобуто такі результати:** проаналізовано сучасні методи вибору кандидатів на перевірку великих чисел на простоту, розглянуті варіанти генерації великих простих чисел, виявлені основні недоліки цих методів для практичного застосування побудованих таким чином простих чисел. Запропоновано та теоретично обґрунтовано метод визначення кандидатів для перевірки великих чисел на простоту та триетапний метод для перевірки чисел на простоту. Проведене тестування запропонованого методу визначення простоти показало правильність теоретичних висновків про можливість застосування запропонованого методу для вирішення поставленої задачі. **Висновки:** Використання стратегії вибору кандидата на простоту дозволяє значно зменшити кількість перевіряних чисел. На числах розміром у 200 десятинних знаків кількість чисел для перевірки зменшується до 8,82%. Зі зростанням розміру чисел для перевірки їх кількість буде зменшуватися. Запропонований метод перевірки чисел на простоту достатньо простий та ефективний. Перші два етапи дозволяють відсіяти всі складені числа, за винятком чисел Кармайкла. При цьому на першому етапі при використанні перших десяти простих чисел відсівається більше 80% чисел для перевірки. На другому етапі проводиться відсів складених чисел з складниками більше 29. На третьому етапі відсіюються числа Кармайкла. Тест є поліноміальним, детермінованим та безумовним.

Ключові слова: прості числа; генерація великих простих чисел; метод визначення простоти; числа Кармайкла.