Security
of Infocommunication Systems
and Internet of Things

2024 Vol 2, No 1
https://doi.org/10.31861/sisiot2024.1.01004

Received 20 June 2024; revised 29 August 2024; accepted 30 August 2024; published 30 August 2024

Research the Level of Chaotic and Reliability in Webcam-generated Random Number Sequences

Rostyslav Diachuk, Yuriy Dobrovolsky, Dmytro Hanzhelo, Heorhii Prokhorov* and Denis Trembach

Software Engineering Department, Yuriy Fedkovych Chernivtsi National University, Chernivtsi, Ukraine

*Corresponding author (E-mail: g.prokhorov@chnu.edu.ua)

ABSTRACT Engineering and software solutions in the sphere of software engineering, in particular cryptography, constantly require the use of random sequences in their algorithms. Usually, standard methods of frameworks, software platforms, libraries and programming languages do not provide the necessary level of randomness of generated number sequences. Basically, the randomness of software generators of random sequences is based on a value of a certain system parameter, for example, the value of the current date and time. It is obvious that with simple cyber-attacks it is possible to put the crypto-resistance of the system at risk. To solve this problem, the property of the webcam to generate the same image from a state frame is used. It is shown that the fact of changing values of illumination created by pixels differs by at least 63% for two consecutive frames 100 milliseconds gap in complete darkness, the level of the avalanche effect exceeds the crypto-resistance requirement by 13%. Thus, we can talk about a high level of chaos and randomness of the generated numerical sequences. Frame generation was carried out both in complete darkness with an illumination of 10-4 lux, and in a uniformly illuminated (200 lux) white surface. Testing the camera under extreme conditions gives a complete picture of the unpredictability and chaos in the generation of random sequences. It is hypothesized that this approach theoretically allows a generation of random number sequences at a speed of 1.25 Gbit/s, and a mixed software-hardware solution is able to provide up to 10 Gbit/s. The approach built on this property of a webcam can provide a way to solve the problem of designing an affordable low-cost, crypto-resistant high-speed hardware random number generator in laboratory conditions without the involvement of a special equipment.

KEYWORDS software engineering, chaos, crypto-resistance, software reliability, random number generator.

I. INTRODUCTION

At the current level of the development of information technologies the issues of supporting the protection of the vital interests of a human, a state and the whole society, the national interests of Ukraine in cyberspace from the point of view of cyber security are of crucial importance [1].

One of the component methods of cyber security based on software engineering is the generation of random sequences – one of the mandatory elements of guaranteeing cyber security of data. The computer for generation can use as some of its own pseudo-random value, for example, the amount of used/free stack/heap memory or the current time value. So are data from peripheral devices – USB, Keyboard, Mouse and many other sources, called external sources of entropy.

These numbers are not overall random, because they have a predictable nature of changes. In order to transform such a set of numbers into a really random set, cryptotransformations can be applied to them, for example, cellular automata [2-5] to get uniform distribution of random values from unevenly distributed ones of the chaos source. The resulting number sets are declared as pseudo-random ones, because they are not random in fact, but deterministically produced from entropy. A modern crypto-algorithm, encrypting data, produces ciphertexts that should not statistically distinguish from a trully random sequence. Namely, so that for the production of random sequences it is possible to take a source of entropy, which ensures satisfactory level of unpredictability and randomness of values even in a short range.

Generated random number sequences are widely used in some secured connections in various network communications, for producing crypto keys, establishing load balance, control of integrity, and for some other applications [6].

Thus, from the point of view of software engineering, there are relevant works devoted to the investigation of new approaches to the generation of random sequences at the hardware level.

II. ANALYSIS OF CURRENT LITERARY DATA AND FORMULATION OF THE PROBLEM

Three approaches are used to generate random numbers.

The first approach – a software one – is based on specialized mathematical algorithms of software engineering. Unfortunately, software generators are somewhat predictable. Mathematical proofs of unsatisfactory cryptoresistance of pseudorandom sequences are given in [7]. The pseudorandom sequence generation algorithm is publicly available, for instance, for the Java language version 17 [8], which creates a theoretical possibility to crash the encryption algorithm. And in the publication [9], the author describes the hacking process in detail, although with the help of huge computing power. However, with the development of computing power, for example, quantum computing [10], where the high speed of calculations inclines the possibility of an attack to the practical plane. In December 2022, a publication by a group of Chinese scientists appeared, which demonstrated the possibility of breaking long RSA keys using modern quantum computers. Work [11] describes the first ever hacking of a 48-bit key.

Thus, it can be said that the software method of generating random numbers is not completely crypto-resistant.

The second approach – hardware – is related to the construction and usage of special devices that use some physical sources of chaos. For example, in [12], a beta radiation counter is used to generate random values, which makes relevant research dependent on additional equipment. In works [13, 14], the authors use the noise of an analog video camera, but subsequent digitization of the video signal reduces the processing speed to low level.

These approaches, although completely crypto-resistant, require additional expensive and exotic equipment.

Taking this into account, the third approach is often more relevant, which involves the usage of events from standard components of a computer system. The most popular method of random number generation in such a way is random number generation using the CPU clock counter. However, the high sensitivity of a phase noise of generators to external influence was investigated in [15], which means the possibility of influencing the random number generator from the outside.

CPU clock counters allow you to obtain uniformly distributed random numbers. For most modern encryption systems, this is an advantage, since these are the numbers that modern encryption systems work with. However, in some papers [16, 17] a data encryption method using unevenly distributed random numbers was proposed, in which the direct application of a random number generator based on the processor clock counter is unsatisfactory.

In [18], a method of generation using an optical manipulator "mouse" is proposed, which allows obtaining unevenly distributed random numbers. The disadvantage of this method is that the speed of generating random numbers does not exceed 1 kbit/s, which does not allow a creation of a high-speed encryption system based on it.

In our opinion, the most effective way to create an unpredictable sequence of numbers is the use of a webcam, namely, the change of image pixels from frame to frame of the same image, which was considered in [19], but at that time (2014) it was theoretically possible to reach a speed of 200 Mbps, and the maximum resolution of the webcam did not exceed VGA. But at the current stage, it is recommended to have a speed of 1 Gbit/s.

The systematization of the above shortcomings of the existing approaches allows us to formulate the general problem of the lack of a hardware crypto-resistant, affordable, inexpensive laboratory high-speed random number sequence generator.

III. THE PURPOSE AND OBJECTIVES OF THE RESEARCH

The purpose of the research is to create a hardware crypto-resistant, affordable, inexpensive laboratory high-speed RNS generator based on the extraction of pixel values of the webcam matrix.

To solve the set goal, the following tasks were solved:

- development of a method of extracting image pixel values from a webcam frame;
- examine the obtained RNS for randomness of values and avalanche effect.

IV. MATERIALS, CONDITIONS AND RESEARCH METHOD

A. Research equipment. ASUS Z97K desktop computer: CPU Intel® Core™ i3-4170 CPU @ 3.70 GHz × 4, 32 Gb of RAM, SDD Kingston 240 Gb. Web Digital Camera: FULL HD 1080P, TrueColor, QQVGA (176×144), QVGA (320 × 240), VGA (640 × 480), SVGA (800 × 600). In the selected webcam, the upper resolution limit, according to the specification, is 800 × 600 pixels (VGA), and the default mode is 176 × 144 (QQVGA), the Quarter-QVGA resolution. If desired, this size can be expanded to HXGA resolution (4096 × 3072) – it depends on the resolution of the selected camera [20].

Software: OS Ubuntu 22 LTS 64 bit, Java Amazon Corretto 17.0.5, IntelliJ IDEA 2023.3.4 (Ultimate Edition), package com.github.sarxos.webcam version 0.3.12 – frame capture, javax.imageio package – video image processing, package java.security.SecureRandom – software generation of a random sequence.

B. Research methods. To simplify the investigation process, frame capture was performed in the minimum QQVGA (176×144) resolution mode.

The block diagram of the experiment algorithm is shown in Fig. 1. When the camera is in the required conditions, serial two pictures are taken in BMP or TIFF format (1). The default size of each of these snapshots is 176 × 144 pixels, for a total of 25344 pixels. This size is the default for the Webcam class of the webcam-capture package of the Java programming language version 17.

Obviously, the shorter the time interval between frames, the less likely that something can change on them and the better for the experiment. The minimum time interval depends on the computing power of the equipment and was 100 milliseconds. Then the received photos are sent to the server (2).

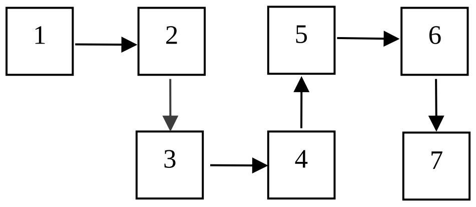

FIG. 1. Block diagram of the algorithm for generating random numbers from the pixels of the image formed by the webcam.

Next, it is necessary to obtain a so-called bitmap from each photo – a matrix (3) in which the values of image elements (pixels) are stored. For a 24-bit bitmap, each pixel contains values of three colors (RGB color mode) – red, green, blue respectively. Each color is allocated 1 byte = 8 bits, that is, the maximum possible number in decimal format is equal to 255, and the minimum one is 0. The number 255 corresponds to the maximum intensity of a color, and the number 0 corresponds to the minimum intensity.

From the point of view of the Java language, the data range of the byte type is the numeric interval of integer numbers [–128 ... +127], but this is overcome by arithmetic conversion.

In this way, a matrix (three-dimensional array) of random numbers in the range [0 .. 255] is obtained:

$$\boldsymbol{A} = (a_{i,j,k})^{m,n,p},\ m=176,\ n=144,\ p=3. \qquad (1)$$

At the next step (4), after a yime interval (100 ms), another frame of the same image is taken and another matrix is formed:

$$\boldsymbol{B} = (b_{i,j,k})^{m,n,p},\ m = 176,\ n =144,\ p = 3. \qquad (2)$$

Next step (5), an elementary matrix subtraction operation is performed:

$$\boldsymbol{C} = \boldsymbol{B} - \boldsymbol{A} = (b_{i,j,k} - a_{i,j,k})^{m,n,p}, \qquad (3)$$

where the appropriate values of the differences in the color components for the two images of each pixel are recorded.

Next, the matrix is flattened into a linear vector:

$$\boldsymbol{E} = (e_i)^t, \qquad (4)$$

where $t = m \times n \times l = 176 \times 144 \times 3 = 76032$.

The resulting values differences can vary in a diapason from –255 to +255.

Next step (6) – grouping and sorting are carried out and the resulting vector of random values is further investigated for compliance with the requirements of speed, cryptoresistance, avalanche effect, etc. (7).

C. Conditions of experiment. To avoid extraneous influence, the webcam that was connected to the computer was located in a dark, deaf box. The frame of darkness is not given, because on it the human eye will not notice inhomogeneities in the distribution of points of different color gamuts. The level of illumination was 10-4 lux, which is considered as a complete darkness. The temperature in the room is 20 ºC.

For the contrast of the investigation, as the opposite side of the limit tests, an image of a completely white homogeneous wall was taken under uniform daytime lighting on a cloudy day exactly at noon (Fig. 2).

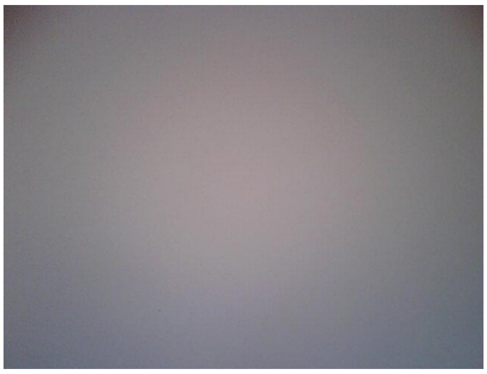

FIG. 2. Webcam image of a white wall.

In the frame, you can see that, despite the homogeneity and uniform lighting of the image, the picture was not homogeneous. It is brighter in the center than on the periphery, the interspersion of other colors, in particular red and green, is clearly visible. This allows us to assume that there are elements of randomness in this byte array. To what extent this randomness satisfies the crypto-resistance requirements given in [21] will be clarified later.

V. THE RESULTS OF THE INVESTIGATION OF THE GENERATED SEQUENCES

The flexible methods of the Webcam and Buffered Image class of the Java programming language made it possible to quickly and optimally extract the number sequence from an instance of a frame immediately without complex matrix transformations. The sequence itself has been placed into a simple Array data structure and is ready for further investigation.

The basic program code of the program is:

```
   Webcam webcam = Webcam. getWebcams() .get(0);
   Dimension dimension = WebcamResolution. VGA.
getSize();
   webcam.setViewSize(dimension);
   webcam.open()
```

First, the system webcam is initialized. By default, the default resolution mode is QQVGA. The camera is opened for work, and access to it by other programs is blocked. With this setting, you can choose the camera itself (if there are several of them) and the frame capture mode (frame resolution). The following code retrieves a snapshot object from the camera, converts the image to a TIFF byte stream, and stores it in a one-dimensional byte array.

```
   BufferedImage image = webcam.getImage();
   ByteArrayOutputStream stream = new
ByteArrayOutputStream()
   ImageIO.write(image, "tiff", stream);
   byte[] bytes = stream.toByteArray();
```

The thus created method of extracting pixel values from a frame using the process of transforming a three-dimensional array into an one-dimensional one is greatly simplified with the help of Java capabilities. When two consecutive frames from the webcam were received with a delay of 100 milliseconds, they further investigated for discrepancies in the values of the corresponding pixels in the TrueColor color gamut. The values of the differences in illumination for all points of the two frames are listed.

```
   list.add(bytes1[i] – bytes2[i])
```

Thus, for each pixel in the image matrix of the first frame, its difference with the corresponding pixel of the second image is entered in the list. To estimate the percentage of chaos between two frames, you need to calculate the amount of all pixels that have changed their values and divide by the total amount.

From two consecutive frames of the webcam in QQVGA mode, 76032 numbers between -255 and +255 were generated as the difference of the corresponding pixel values. The percentage of pixels that has changed its value is 63% for complete darkness, for a white illuminated surface (200 lux) – 82%, for an ordinary office space – 96%. For the completeness of the experiment, the spectrum of disagreements, i.e. the number of points corresponding to each disagreement, was further calculated. This was possible thanks to the aggregation methods of the Java language.

The histogram of the distribution of discrepancies (Δx) by number for two frames is presented in Fig. 3.

In this way, the RNS generated by the webcam were examined for randomness, which showed that, as seen in Figure 3, for each pixel, the difference in pixel values between the first and second image was calculated. As can be seen from the histogram, the difference was mostly within ± 10. Approximately 28000 pixels (or 37%) have zero deviation. And 9.5K pixels have a deviation of ± 1, etc. The maximum deviation of -50 has a single point out of 75K. As for the avalanche effect, the proposed method shows significantly greater resistance to it, and is at least 63% for complete darkness, which is 13% better than the standard requirements.

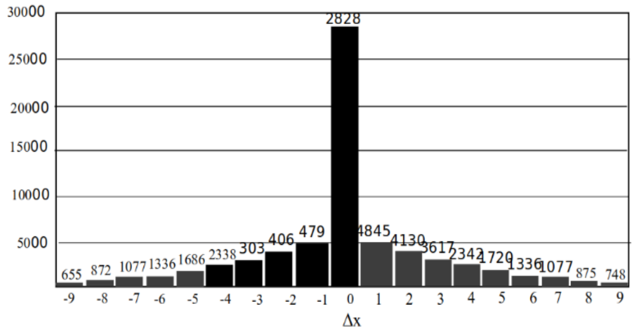

FIG. 3. Histogram of the distribution of differences Δx of two camera frames.

VI. DISCUSSION OF RESEARCH RESULTS

The obtained results confirmed the high level of randomness and unpredictability of RNS. The number of pixels that did not change their value on two consecutive frames is approximately 28000 (Fig. 3), or 37%.

The obtained solutions (chaos value of 63% under the most unfavorable conditions – complete darkness) indicate a high level of the avalanche effect – according to the requirements of crypto-resistance, it is enough to ensure a minimum level of 50%. This is explained by the confirmation of the hypothesis about the chaotic nature and unpredictability of digital noise in the camera matrix due to the noise having the origin from the stochastic nature of the photons interaction with the atoms of the photo-diode material of the sensor. The obtained results are provided due to the development of the method described in [19], namely, the generation of pixels in the dark at an illumination of about 10-4 lux. This is a simple solution, in contrast to, for example, work [12], where the chaos generator was a source of beta radiation.

The disadvantage of the study is the impossibility of testing RNS with generally accepted NIST tests due to insufficient size [21]. The development of the research has a perspective, namely, when the limitations are removed, for example, not complete darkness, but a bright colorful background (200 lux), the level of chaos will be even higher (up to 100%). Theoretically possible to concatenate RNS of sequential frames and obtain a generation if high performance up to 1.25 Gbit/s.

VII. CONCLUSION

Based on the results of the research, it was established that the value of the intensity of the webcam pixels is unpredictable and random, which is not visible to the naked eye, but is clearly recorded by hardware and software tools. Namely, in two pictures in complete darkness with an interval of 100 milliseconds, 63% of the pixels changed their value, which is 13% better than the standard requirements of crypto-resistance to the avalanche effect.

AUTHOR CONTRIBUTIONS

Y.D. – conceptualization, G.P. – methodology, investigation; R.D. – software, experiment; D.H. – statistics, D.T. – conceptualization, writing (original draft preparation).

COMPETING INTERESTS

The authors declare no competing interests.

REFERENCES

[1] UKAZ PREZYDENTA UKRAINY №37/2022 "Pro rishennia Rady natsionalnoi bezpeky i oborony Ukrainy vid 30 hrudnia 2021 roku "Pro Plan realizatsii Stratehii kiberbezpeky Ukrainy". URL: https://www.president.gov.ua/documents/372022-41289. [in Ukrainian]

[2] H. Fukś, "Four State Deterministic Cellular Automaton Rule Emulating Random Diffusion," In: Chopard, B., Bandini, S., Dennunzio, A., Arabi Haddad, M. (eds) *Cellular Automata. ACRI 2022.* Lecture Notes in Computer Science, vol. 13402, 2022. Springer, Cham. https://link.springer.com/chapter/10.1007/978-3-031-14926-9_13.

[3] Dobrovolsky, Y. "Development of a Hash Algorithm Based on Cellular Automata and Chaos Theory." *Eastern-European Journal of Enterprise Technologies,* 5/9 (113) 2021. P. 48-55. DOI: 10.15587/1729-4061.2021.242849 https://journals.uran.ua/eejet/article/view/242849/241487.

[4] A. Cicuttin, L. De Micco, M. L. Crespo, et al., "Looking for Suitable Rules for True Random Number Generation with Asynchronous Cellular Automata," *Nonlinear Dynamics,* vol. 111, pp. 2711-2722, 2022. https://doi.org/10.1007/s11071-022-07957-8.

[5] L. Li, Y. Luo, S. Qiu, et al., "Image Encryption Using Chaotic Map and Cellular Automata," *Multimed Tools Appl,* vol. 81, pp. 40755–40773, 2022. https://doi.org/10.1007/s11042-022-12621-9.

[6] Asia Othman Aljahdal, "Random Number Generators Survey," *International Journal of Computer Science and Information Security (IJCSIS),* Vol. 18, No. 10, October 2020. [Online]. Available: https://zenodo.org/records/4249407.

[7] F. Martinez, "Attacks on Pseudo Random Number Generators Hiding a Linear Structure," in *Topics in Cryptology – CT-RSA 2022,* S. D. Galbraith, Ed. Cham: Springer, 2022, vol. 13161, *Lecture Notes in Computer Science*. [Online]. Available: https://doi.org/10.1007/978-3-030-95312-6_7.

[8] Class SecureRandom. All Implemented Interfaces. URL: https://docs.oracle.com/javase/8/docs/api/java/security/SecureRandom.html.

[9] M. Cornejo, S. Ruhault, "(In)Security of Java SecureRandom Implementations," *Journées Codage et Cryptographie,* 2014. [Online]. Available: https://www-fourier.ujf-grenoble.fr/JC2/exposes/ruhault.pdf.

[10] Ostapov, S.E., Dobrovolskyi, Yu.H. "Kvantova informatyka ta kvantovi obchyslennia." Chernivtsi: ChNU, 2021. - 99 s. https://archer.chnu.edu.ua/xmlui/handle/123456789/2830. [in Ukrainian]

[11] B. Yan, Z. Tan, S. Wei, H. Jiang, W. Wang, H. Wang, *et al.*, "Factoring Integers with Sublinear Resources on a Superconducting Quantum Processor," *arXiv,* vol. 2212.12372v1, Dec. 2022. [Online]. Available: https://arxiv.org/pdf/2212.12372.pdf.

[12] Seongmo Park, Byoung Gun Choi, Taewook Kang, Kyunghwan Park, Youngsu Kwon, Jongbum Kim, “Efficient Hardware Implementation and Analysis of True Random-Number Generator Based on Beta Source,” *ETRI*, Volume 42, Issue 4, Special Issue on SoC and AI Processors, August 2020, Pages 518-526. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.4218/etrij.2020-0083.

[13] V. Barannik, S. Sidchenko, N. Barannik, and A. Khimenko, "The method of masking overhead compaction in video compression systems," *Radioelectron. Comput. Syst.*, no. 2, pp. 51-63, 2021. [Online]. Available: https://doi.org/10.32620/reks.2021.2.05.

[14] S. Yevseiev, O. Milov, N. Zviertseva, O. Lezik, O. Komisarenko, A. Nalyvaiko, V. Pogorelov, V. Katsalap, Y. Pribyliev, and I. Husarova, "Development of the concept for determining the level of critical business processes security," *East.-Eur. J. Enterp. Technol.*, vol. 1, no. 9(121), pp. 21–40, 2023. [Online]. Available: https://doi.org/10.15587/1729-4061.2023.274301.

[15] Agata Kaźmierczyk, Andrzej Ł. Chojnacki, Kornelia Banasik. "Pseudorandom Number Generators as Applied in Reliability Analysis." Kielce University of Technology, Faculty of Electrical Engineering, Automatic Control and Computer Science, Department of Power Engineering, Power Electronics and Electrical Machines, doi:10.15199/48.2022.12.44. [Online]. Available: http://pe.org.pl/articles/2022/12/44.pdf.

[16] V. Barannik, N. Barannik, and O. Slobodyanyuk, "Indirect information hiding technology on a multiadic basis," *Informatyka, Automatyka, Pomiary w Gospodarce i Ochronie Środowiska*, vol. 11, no. 4, pp. 14–17, 2021. doi:10.35784/iapgos.2812.

[17] V. Barannik, N. Barannik, O. Ignatyev, and V. Himenko, "Method of indirect information hiding in the process of video compression," *Radioelectronic and Computer Systems*, vol. 0, no. 4, pp. 119–131, 2021. doi:10.32620/reks.2021.4.10.

[18] S. Ostapov, B. Diakonenko, M. Fylypiuk, K. Hazdiuk, L. Shumyliak, and O. Tarnovetska, "Symmetrical cryptosystems based on cellular automata," International *Journal of Computing*, vol. 22, pp. 15–20, Mar. 2023. https://doi.org/10.47839/ijc.22.1.2874.

[19] R. Li, "A true random number generator algorithm from digital camera image noise for varying lighting conditions," in SoutheastCon 2015, Fort Lauderdale, FL, USA, 2015, pp. 1–8. doi: 10.1109/SECON.2015.7132901. Available: https://ieeexplore.ieee.org/document/7132901.

[20] "Webcam-capture Resolution," GitHub. [Online]. Available: https://github.com/sarxos/webcam-capture/blob/master/webcam-capture/src/main/java/com/github/sarxos/webcam/WebcamResolution.java.

[21] L. Afflerbach, "Criteria for the assessment of random number generators," *Journal of Computational and Applied Mathematics*, vol. 31, no. 1, pp. 3–10, 1990. doi: 10.1016/0377-0427(90)90330-3.

Rostyslav Diachuk

Had received BS and MS degrees in Software Engineering Department from Yuriy Fedkovych Chernivtsi National University. He worked as an assistant of the Software Engineering Department, but currently studying Ph.D on the same department. His research interests include cryptography, pseudorandom sequence systems.

ORCID ID: 0000-0002-6259-3302

Yuriy Dobrovolsky

Graduated from the Faculty of Physics and Mathematics in 1984. Received the degree of Doctor of Technical Sciences. Currently, he is a professor at the department of software engineering at Yuri Fedkovich Chernivtsi National University. His research interests include software reliability engineering, cryptography, coding theory, hardware random number sequence generation.

ORCID ID: 0000-0003-0626-0594

Dmytro Hanzhelo

Now is studying on a Ph.D. in Computer Science, Yuriy Fedkovych Chernivtsi National University. He is currently a security practitioner, mentor, and part-time lecturer. His research interests include cybersecurity, cryptography, random number sequences generation.

ORCID ID: 0000-0002-0836-4568

Heorhii Prokhorov

He had received a Ph.D. in physics and mathematics in 2006. Now is a Assistant Professor of Software Engineering Department, Yuriy Fedkovych Chernivtsi National University. His research interests include cryptography, coding theory, hardware random number sequences generation.

ORCID ID: 0000-0001-7810-2785

Denis Trembach

Had received BS and MS degrees in Information Security from Evropejs'kij Universitet Financiv, Ukraine. Now is studying on a Ph.D. in Computer Science, Yuriy Fedkovych Chernivtsi National University. He is currently a security practitioner, mentor, and part-time lecturer. His research interests include cybersecurity, applied AI, chaotic systems dynamics.

ORCID ID: 0000-0001-8095-4186

Дослідження рівня хаосу та надійності у послідовностях випадкових чисел, що згенеровані вебкамерою

Ростислав Дячук, Юрій Добровольський, Дмитро Ганжело, Георгій Прохоров[*], Денис Трембач

Кафедра програмного забезпечення комп'ютерних систем , Чернівецький національний університет ім. Юрія Федьковича

*Автор-кореспондент (Електронна адреса: g.prokhorov@chnu.edu.ua)

АНОТАЦІЯ Захист інформації, збереження цілісності, надійність програмного забезпечення для передачі даних є сьогодні важливою складовою інформаційних технологій. Інженерні та програмні рішення у сфері програмної інженерії, зокрема криптографії та надійності, постійно потребують використання випадкових послідовностей у своїх алгоритмах. Зазвичай, стандартні методи фреймворків, програмних платформ, бібліотек та мов програмування не забезпечують необхідний рівень випадковості згенерованих послідовностей. В основному випадковість програмних генераторів послідовностей чисел базується на значенні стану певного системного параметру, наприклад значенні поточної дати та часу. Очевидно, що при нескладних кібер-атаках можливо досягти передбачуваного результату і поставити криптостійкість системи під загрозу. Для розв'язання цієї проблеми застосовано властивість вебкамери генерувати послідовності випадкових чисел з одного і того ж зображення. Показано, що факт зміни значень освітленості, створеної пікселями, відрізняється, щонайменше на 63 % для двох послідовних кадрів з інтервалом у 100 мілісекунд у повній темряві, що на 13 % перевищує вимоги до криптостійкості. Таким чином, можна говорити про високий рівень випадковості згенерованих числових послідовностей. Генерація кадрів здійснювалась як при повній темряві при освітленості 10-4 люкса, так і рівномірно освітленої (200 люкс) білої поверхні. Випробування камери на граничних умовах дають повну картину непередбачуваності та хаосу при генеруванні випадкових послідовностей. Висловлюється гіпотеза, що даний підхід теоретично дозволяє генерувати випадкові послідовності з швидкістю 1.25 Гбіт/сек, а програмно-апаратне рішення до 10 Гбіт/сек. Підхід, побудований на цій властивості вебкамери, дозволить вирішити проблему проектування доступного недорогого лабораторного криптостійкого надійного швидкісного апаратного генератора випадкових чисел у лабораторних умовах без залучення спеціального обладнання.

КЛЮЧОВІ СЛОВА програмна інженерія, хаос, криптостійкість, надійність програмного забезпечення, генератор випадкових чисел.