Trojan Attack on the Initialization of Pseudo-Random Bit Generators Using Synchronization of Chaotic Input Sources

MICHAL MELOSIK¹ AND WIESLAW MARSZALEK²
¹Department of Computer Science and Telecommunication, Poznan University of Technology, 60-965 Poznan, Poland
²Department of Computer Science, Opole University of Technology, 45-758 Opole, Poland
Corresponding author: Wiesław Marszałek (w.marszalek@po.edu.pl)

ABSTRACT This paper deals with a safety problem of entropy sources in deterministic random bit generators. An initialization input attack on a set of two pseudo-random generators using analog-digital hardware Trojan is presented. Trojan circuit implementations are discussed following their full classification. Simulation results confirming the effectiveness of the Trojan impact on a repetition of identical seed values are analyzed. Trojan structure is integrated with the deterministic random bit generator functional model recommended by NIST and the task is to impact two seed generators (in two channels), which use the independent Chua circuits as the sources of entropy. The analog Trojan part causes synchronization of the two analog chaotic circuits as explained by the results of LTSpice simulation. The digital Trojan part controls the analog part and modifies the values of optional parameters used in the creation of the seeds. The seed value creation in two cases (with and without Trojan) are simulated by using Xilinx ISIM.

INDEX TERMS Hardware trojans, chaotic circuits, entropy source, input initialization attack, seed generators.

I. INTRODUCTION
Hardware random generators are often designed as a combination of two sub-generators, the true-random, or TRBG, with slow dynamics, and the pseudo-random, or PRBG, with high speed [1]. In such a combination, the TRBG acts as the PRBG’s seed generator that initiates or reinitiates PRBG. This approach is discussed in detail in [2]. In the manufacturing processes of electronic devices, a possibility of using a certain IP Core is often constrained by a limited design budget. Reducing the costs of designing and implementing microelectronic systems requires various design compromises. For security modules, such compromises are the use of pseudo-random generators instead of commercial quantum ones. However, such approach can unfortunately lead to a low bitrate compared to commercially available random generators. In practical application, the bitrate can be increased by parallel implementation of several identical pseudo-random algorithms in one digital system. A variety of random values is ensured by their initialization with seed values from the low cost and easy to built independent entropy sources. Such a demand for independent entropy sources is satisfied by various analog chaotic circuits [1], [3]. Seed values created from two identical uncorrelated chaotic circuits in the same time interval can be used to initiate two identical pseudo-random algorithms. The variety of seed values created in this way satisfies the security requirements of the initialization process as described by the National Institute of Standards and Technology (NIST) [2]. The security of entropy sources is a research topic in the areas of operating system virtualization and cloud services [4]–[10]. However, with respect to the design of deterministic random generators as microelectronic systems, this problem has not been sufficiently studied in the literature. Taking into account the contemporary hardware Trojan threats, this problem should also be considered during the evaluation of printed circuit boards (PCBs), in which various analogue entropy sources are assembled. In research on hardware security and trust, the presence of hardware Trojans is quite often only limited to integrated circuits, thus ignoring the possibility of hacking attacks at the highest level of abstraction that PCBs are for microelectronic systems [11]–[13]. The success of the integration of hardware
Trojan into PCBs is determined by the efficiency of reverse engineering, which identifies the most critical modules. The effectiveness of reverse engineering depends mainly on what kind of protection against hacking or tampering has been applied at the PCB design process [14], [15]. Considering the fact that most of the modern microelectronic systems are only protected by a miniature screw-on housing, therefore, hackers do not encounter significant difficulties in accessing PCB mounted components. Therefore, the question should be asked whether, in the final assembly of a device, such attacks can actually take place. To answer this question, an analysis of the attack described in [16] will be helpful. This attack demonstrates a situation in which the hacker does not encounter any problems in accessing the electronic circuits to perform reverse engineering. The hacker is able to get the information about which communication interface pins should be soldered to the external modification chip device, in order to extract the sensitive data. Such attacks make it necessary to investigate whether the entropy sources (being often analog circuits) exposed to an easy access to their circuit structure can cause a threat in the process of forming seed values. In this article we analyze the threat of synchronizing chaotic entropy sources initiating a set of pseudo-random generators. As shown in [1], such synchronization is not possible by observing bits after post-processing of binary values from chaotic entropy sources. The research presented in this paper shows that such synchronization is possible by designing a dedicated hardware Trojan. We cover the synchronization of circuits by an unauthorized modification – connections between chaotic analog circuits. According to the authors’ knowledge, no attack in which a synchronization between two independent entropy sources occurs because of a hardware Trojan, has been described in the literature. This new type of synchronization Trojan attack is described in sections II, III and IV in this paper and supplemented by a description of measures that can be used to prevent such an attack, as presented in section V.

II. AN INPUT ATTACK ON FUNCTIONAL MODEL OF A DETERMINISTIC RANDOM BIT GENERATOR

As shown in [1], analog chaotic circuits with slow dynamics occupy a minimal area of PCB and the randomness of their generated values was confirmed by a set of NIST tests. The conclusion of the research presented in [1] is that analog chaotic circuits can be used as seed generators for digital PRBG. However, what was not considered is a parallel implementation of random generators for independent cryptographic channels and the consequences of such a case. Given the deterministic nature of PRBG, it is important to make sure that, the seed value from one entropy source can be used to initialize only one PRBG with the same deterministic algorithm, as shown in the top diagram in Fig. 1. The diagram shows two channels (A and B) with pseudo-random generators \( PRBG_{IP1} \), in which different sets of random bits are secured by independent sources of entropy. The lack of correlation of the sources of entropy and a necessity to eliminate bias (through the von Neumann corrector) are achieved by the process of creation of the seed values \( seed_{A1} \) at different time instants \( t_{A1} \) and \( seed_{B1} \) at different time instants \( t_{B1} \). Because of the finite number of bits used to represent random values in successive iterations, it is necessary to re-initialize \( PRBG_{IP1} \), which is done at the instants \( t_{A2} \) and \( t_{B2} \). Introducing the analog-digital Trojan (bottom diagram in Fig. 1) causes synchronization of the independent entropy sources - the seed values become identical in both channels and are created at the same time instant \( t_{HWT-1} \). As a consequence, identical random sequences are appearing in channels A and B, also after a re-initialization at \( t_{HWT-2} \).

Ignoring this type of threat may result in a possibility of attacks being launched on the PRBG input. Special interest for hackers may be the input initialization attack, which is not intended to cause a decrease of entropy in the seed generators, but to cause simultaneous duplication of the initiating values created by two seed generators [17]. As a result of such an attack a set of parallel pseudo-random generators (based on identical deterministic algorithms \( PRBG_{IP1} \)) is initialized with identical seed values resulting in \( PRBG_{IP1}(seed_{A}) = PRBG_{IP1}(seed_{B}) \). The bottom diagram in Fig. 1 shows a situation in which the hardware entropy source is synchronized with each other. Based on their observations identical seed values are created for both cryptographic channels A and B, resulting in a forbidden case [17], [18].

Fig. 2 shows a detailed concept of the initialization input attack in the functional DRBG model defined by NIST in [2]. This attack requires implementation of two modules. The first one is an analog circuit that allows to induce synchronization between two chaotic entropy source. It is not expected that the analog synchronization itself may cause the initialization input attack, because of different parameter values \( nonce \) and \( pers.str \) for channels A and B. Therefore, it is also necessary...
In this case, chaotic Chua circuits, acting as independent seed modules of cryptographic system security were identified. Seed generators, an analog-digital hardware Trojan. Its implementation made it possible to design and integrate with a set of chaotic systems have shown that the engineers’ creativity significantly influences the development and design of projects [19], [20]. The same modularity and creativity can be widely used by a hacker to design a dedicated hardware Trojan, integrated with the first seed generator. The functional DRBG model defined by NIST is appeased by a Trojan consisting of an analog part (HWT Analog), integrated with the sources of entropy and the digital part (HWT Digital), which modifies the optional parameters nonce and pers.str. Activation and de-activation of the analog synchronization is done by controlling of the analog Trojan module by the HWT-FSM, being integrated with the seed generator in channel A. The same FSM is responsible for controlling of the digital module, replacing the nonce and pers.str. and making them identical in both channels.

III. ANALOG-DIGITAL HARDWARE TROJAN SYNCHRONIZING CHAOTIC SEED GENERATORS

In order to prepare the attack from Fig. 1, the hacker must first design a dedicated analog-digital hardware Trojan. Research results in the areas of cognitive psychology and creativity concepts in the modular design of electronic embedded systems have shown that the engineers’ creativity significantly influences the development and design of projects [19], [20]. The same modularity and creativity can be widely used by a hacker to design a dedicated hardware Trojan, integrated with the selected module, as is discussed in this section. Modularity and creativity, preceded by successful reverse engineering, made it possible to design and integrate with a set of chaotic seed generators, an analog-digital hardware Trojan. Its implementation and full classification according to [21] is shown in Fig. 3. In the first stage of the reverse engineering, the key modules of cryptographic system security were identified. In this case, chaotic Chua circuits, acting as independent seed generators, were recognized on the basis of analog circuit topologies. Full-scale PCBs of the analog Chua circuit is not a widespread solution, but it is rather simple in implementation and inexpensive, compared to other commercially available entropy sources. Based on selected voltage signals from these circuits, the $U_4$ and $U_5$ voltage comparators, form input bitstreams passed on ports 23 and 24 of the IC $-1$ integrated circuit (FPGA device). To make an attach on input possible the hardware Trojan must force generation of the same seed values for channels A and B. Synchronization is done by using a voltage follower on a separate, free PCB surface, connecting nodes $nA2$ and $nB2$. Additionally, based on the general classification of Trojans in [21], synchronization will be activated at a fixed point in time. Trojan circuit consists of a trigger that controls the ADG1208 switching unit, which activates synchronization. The ADG1208 operating principle is to switch to output $D$ one of the inputs $S_1 - S_4$, depending on the configuration of the [HWT $-$ const, HWT $-$ trig] two-bit control vector. In parallel to the analog circuit marked in Figure 3 as HWT Payload, Trojan consists of a digital module that is directly implemented in the FPGA device (IC $-1$). Optional values such as nonce or pers.str. are changed by Trojan during an attack. Integration of the digital module must be accomplished during testing and assembly. Having the original digital implementation project created in EDA tools, it is possible to modify the source code of the hardware description language. This will allow the attacker to generate a new file for programming, reprogramming the FPGA device or replacing it with another one.

IV. RESULTS OF THE ATTACK ON INITIALIZATION OF CHAOTIC SEED GENERATORS

Simulation results from LTSpice for the analog module of the discussed Trojan are shown in Fig. 4. For $t < 255ms$ hardware Trojan is inactive. No Trojan activity is maintained until $D = S_1$. The $S_1$ input is in a high impedance state, causing both Chua circuits to work independently. For $t > 255ms$ the Trojan is activated by sending a vector [HWT $-$ const, HWT $-$ trig] = [0, 1] from IC $-1$ to ADG1208. As a result, input $S_2$ is forwarded to output $D$, thus connecting nodes $nA2$ and $nB2$ through a voltage follower built on a $U_3$ operational amplifier. The activation of the Trojan causes the same voltages in nodes $nA2$, $nB2$, thus formation by the comparators $U_4$, $U_5$ identical binary vectors $V(bit_a)$, $V(bit_b)$, which is an unacceptable security situation [17]. The result is the duplication of random values in channels A and B, which is marked as HWT Results in the Fig. 3. The middle row in Fig. 4 for $t > 255ms$ confirms confirms Trojan operation by relation $V(v_{2a}) - V(v_{2b}) \approx 0$.

Values from the bitstreams $V(bit_a)$ and $V(bit_b)$ are passed to a buffer that stores raw random data before they are post-processed by the von Neumann corrector. Post-processing eliminating bias may cause delays between new PRBG initializations.

In order to present a full analysis of the obtained results we will first analyze two cryptographic channels A and B from

![Diagram](image-url)
Fig. 3 without hardware Trojan. For better readability of the simulation results in Fig. 5, it was decided to skip displaying von Neumann corrector output, due to the low bitrate. Instead, the \( \text{shift256} \) vector was presented. This vector is loaded with bits appearing in the randomness extractor output. When a request is sent to the seed generator to create a new value, the \( \text{shift256} \) vector is zeroed. Each of the seed generators is controlled by its own independent FSM. State changes are highlighted with a blue marker in Fig. 5. In post-processing a state, the von Neumann corrector extracts the randomness from the storage buffer data. The number of bits appearing at the corrector output is counted. When 256 new bits are available the \( \text{A-shift256} \) vector is filled with them, which is reported by changing the value of the \( \text{A-ready-shift256} \) (A1)
FIGURE 4. Synchronization of entropy sources as a result of analog-digital hardware trojan triggering. For \( t < 255 \text{ms} \), chaotic circuits from Fig. 2 generate independent signals \( V(v_{2a}) \) and \( V(v_{2b}) \), used to form voltage levels in corresponding bitstreams signals \( V(b_{i,a}) \) and \( V(b_{i,b}) \). At \( t > 255 \text{ms} \) the Trojan is triggered, resulting in \( V(v_{2a}) - V(v_{2b}) = 0 \) and the identical values in \( V(b_{i,a}) \) and \( V(b_{i,b}) \). Synchronization of the two entropy sources in channels A and B is done for \( V(\text{hwt-trig}) = 1 \) (the middle plot). As a consequence, the two analog voltage signals at \( nA2 \) and \( nB2 \) in Fig. 3 (the Chua circuits) are identical (see the middle plot above for \( t > 255 \text{ms} \)). The bit streams \( V(b_{i,a}) \) and \( V(b_{i,b}) \), shown above in the fourth and fifth plots, respectively, are obtained from the signals \( V(v_{2a}) \) and \( V(v_{2b}) \) by using comparators U4 and U5 in Fig. 3.

In point A2 in Fig. 5 it can be seen how the high state of \( A\text{-ready-shift256} \) activates the transition to the state \( A\text{-load-optional} \) and loading the optional values \( A\text{-nonce} \) and \( B\text{-pString} \). The \( A\text{-seed} \) vector is formed as \( A\text{-seed} = A\text{-shift256 XOR A\text{-nonce} XOR A\text{-pString} \), being in the \( A\text{-seed-a} \) state. Point A3 marks the initial fragment of the seed vector which is reported by \( A\text{-ready-seed} \). The process of generating the seed value ends with the transition of the FSM to \( idle \).

An similar process of creating a seed value takes place in channel b (points B1-B3). Comparing channels A and B in Fig. 5, it can be seen that there are no simultaneously vectors \( A\text{-shift256}, B\text{-shift256} \) generation. This is possible by high sensitivity to the initial conditions of chaotic circuits, as shown in figure 4 for \( t < 255 \text{ms} \).

Fig. 6 shows the digital simulation results. The entire Trojan is controlled by an independent \( HWT\text{-FSM} \), in which individual states are highlighted in red. The triggering of the Trojan is done in two stages. In the first one, the analog circuit synchronization is activated by changing the \( V(\text{HWT-trig}) \) value from 0 to 1 as shown in the middle row of Fig. 4. In \( \text{hwt-sniff} \) state, Trojan awaits completion of forming \( A\text{-shift256} \) vector. When this happens, the Trojan triggers its digital module as shown in Fig. 6 with the high state of the \( \text{hwt-en} \) signal. The area of activity of the digital module is highlighted in Fig. 6 with a light red rectangle marker. As mentioned before, in the attack on the initialization of pseudo-random generators, the \( \text{nonce} \) and \( \text{pers. str.} \) values are converted into values selected by the attacker. In this case, the default values were changed to two zero vectors, which was marked with red lines. The values of the seeds will be identical in both channels and equal to the \( \text{shift256} \) vector values. XOR operation will not perform the expected mixing of bits. Completing the attack Trojan goes to the \( \text{hwt-hidden} \) state, hiding evidence of its activity by restoring the original values for nonce and pers.str. Both deterministic algorithms are initialized with the same values, which is the intended target of the attack on PRBGs.

In a quest of continuous access to sources of randomness it can be observed that the NIST tests, such as for example, NIST SP 800-22, can only examine statistical properties of randomness, independently in each separate channel. It is worth recalling that the NIST tests do not examine randomness of the entropy source as a whole, but rather evaluate the statistical (random) properties of a particular binary sequence recorded at a specific time slot and generated under specific conditions, based on an observation of that entropy source [22]. Detection of a lack of randomness by NIST tests can occur when the values from the entropy source appear in an orderly and predictable manner in a bitstream under analysis or in a larger part of it. The simplest situation to obtain negative NIST results is to test consecutive bitstreams composed of regularly incremented values.
Another example of obtaining negative NIST test results is when the values from the entropy source are identical and invariant across successive recorded measurement samples. Furthermore, detections of abnormalities by NIST tests could occur when a chaotic circuit is modified so that it begins to generate periodic signals instead of typically chaotic waveforms - as discussed in detail in [12], [13]. In a synchronization attack, no such disturbance is induced, resulting in

FIGURE 5. Simulation results of digital seed generation in channels A and B using entropy sources without analog-digital hardware Trojan. The A1 is the end time of the pre-processing activity by the von Neumann correction. This eliminates bias from the bit stream \( V(\text{bit}) \) (Fig. 4). The pre-processing ends when buffer \( A_{\text{shift}256} \) is filled up with 256 bits. The preliminary part of buffer \( A_{\text{shift}256} \) after the von Neumann correction is marked in green. The A2 is the state in which the seed generator absorbs the values \( A_{\text{nonce}} \) and \( A_{p\text{String}} \), used optionally in creation of the values initializing PRBG\(_{IP1}\). The A3 is the state in which the seed value is created (marked in brown) for PRBG\(_{IP1}\) in channel A. This is the result of \( A_{\text{shift}256} \) XOR \( A_{\text{nonce}} \) XOR \( A_{p\text{String}} \). The seed is transferred to PRBG\(_{IP1}\) at the time when \( A_{\text{ready}} \) is in the high state. The same process of creation of seed\(_{A}\) occurs in channel B. To secure higher diversity, the values of \( B_{\text{nonce}}, A_{\text{nonce}} \) and \( A_{p\text{String}}, B_{p\text{String}} \) are different, respectively. The time difference in creation of seed\(_{A}\) and seed\(_{B}\) values are the results of high sensitivity of the chaotric Chua generators on initial conditions. Such generators are used as entropy sources in Fig. 3.

FIGURE 6. Simulation results of digital generation of seeds in channels A and B during the interaction of analog-digital hardware Trojan on entropy sources. The A1-HWT and B1-HWT denote the buffers \( A_{\text{shift}256} \) and \( B_{\text{shift}256} \), respectively, which contrasts to A1 and B1 in Fig. 5, and are identical due to synchronization of the chaotic entropy sources done by the analog Trojan. Simulation of the digital Trojan module is marked in red and contains A2-HWT and B2-HWT. The optional parameters \( A_{\text{nonce}}, B_{\text{nonce}} \) and \( A_{p\text{String}}, B_{p\text{String}} \) contain zero vectors. This prevents different seed\(_{A}\) and seed\(_{B}\) values to occur in channels A and B. The zero vectors yield seed\(_{A}\) = seed\(_{B}\) = \( A_{\text{shift}256} \). This is marked by the points A3-HWT and B3-HWT.
the formation of a regular pattern in the sequence of bits ‘0’ and ‘1’. A synchronization attack aims to synchronize chaotic runs in order to duplicate irregular bit sequences in each of the independent cryptographic channels. Note that the NIST tests are useless in detecting a synchronization attack. The test results for two independent channels will indicate that the bitstreams in both channels, A and B, exhibit randomness. The NIST tests will not show that duplication of the same random sequence has occurred in the other parallel channel. The registration of bits from channels A and B for evaluation by NIST tests is done independently for each channel. As shown in [1], a bit sequence obtained from an analog chaotic circuit and then subjected to the von Neumann correction exhibits a satisfactory level of randomness - most NIST tests result in the ‘pass’ results. On this basis, the binary values resulting from monitoring the chaotic entropy source can be used in the initialization of pseudorandom generators.

The attack presented in this paper is, to the authors’ knowledge, novel in that its identification can not be achieved by employing the known methods for detecting randomness in time series, for example, as described above, by the commonly used NIST tests. Moreover, as noted in [1], synchronization of chaotic circuits based on the bitstream formed from observations of their selected chaotic signals is not easy to achieve. The attack presented in this paper shows how such a synchronization can be achieved using a specially designed analog-digital hardware Trojan.

V. TECHNICAL MEASURES TO PREVENT SYNCHRONIZATION ATTACKS
One of the best known hardware security classifications for cryptographic modules is contained in FIPS 140-2 [23], a mandatory standard for the protection of sensitive or valuable data within federal systems. In FIPS 140-2, four generally defined levels of security can be achieved:

- **Level 1**: minimum requirements for ensuring security of cryptographic module. The used defense mechanisms do not guarantee effective hardware security.
- **Level 2**: improvement of the security level from level 1 through the addition of some requirements, including special shielding or sealing to prevent unauthorized modification to the system.
- **Level 3**: increased protection against physical tampering with the system to prevent access to critical components.
- **Level 4**: the highest possible level of security. Resistance to attacks is ensured by using a special shield of the cryptographic module, with a detection of any manipulation attempted in the system.

The analysis of the above-mentioned security levels can be helpful in selecting security features that can significantly hinder or prevent a synchronization attack performed on chaotic entropy sources. Technical prevention for a synchronization attack would include three different solutions. The first one is an implementation of a security patch seal. Cryptographic modules are protected by a basic shield or a dedicated security seal. Verification of the state of breaking such seal should be done by an engineer before a complete activation of the device is performed. Its breach indicates a possibility of unauthorized interference with the device structure on the PCB level. If the device is operated stationary, one can consider developing a dedicated expert system with a miniature camera allowing remote visual inspection of the integrity of the device. The second technical measure is the circuit obfuscation. The use of circuit obfuscation techniques, significantly complicates identification of key modules and recognition of circuit topology implemented directly on the PCB [24], [25]. This technique involves a placement of additional (redundant) components or microcircuit that make it difficult to perform reverse engineering to identify the key modules and functions of specific circuits. However, the use of this approach will involve a greater use of space on the PCB, as well as the use of more electronic components and circuits. As a third engineering measure to protect against synchronization attack, an independent modular implementation with enhanced external enclosures can be considered. In this approach, circuits that are chaotic seed generators for pseudorandom generators should be implemented as independent modules. These modules should be further protected by a shield that prevents any physical tampering with the structure of that module, as well as the X-raying it to evaluate the hardware implementations it protects. The choice of a particular technical method of protection should be determined by the specifics of the device use, as well as the level of hardware security the designers wish to achieve. The attack presented in this paper is for the case when two seed generators form an external analog module. If the entire circuit is designed as a system-on-chip then the Trojan design for seed generators has to be done at the abstraction level, as classified in Fig. 3.

VI. CONCLUSION
Implementations of random generators must ensure that the seeding process is kept safe. The inability to establish a pattern in the process of generating seed values and influence such a process is the key aspect of hardware security and trust. The article discusses the case of two cryptographic channels, with the identical PRBGs. The study illustrates one of the possible risks of such an implementation. The use of reversible engineering for topological analysis of electronic circuits allowed to identify the types of entropy sources used. On this basis, an analog-digital hardware Trojan was designed to synchronize the analog Chua circuits and temporarily change the values of the nonce and pers.str. optional parameters. Activation of a dedicated hardware Trojan causes a forbidden situation in cryptography, where two independently implemented, parallel operating PRBGs were initialized with identical seed values at the same time instant. This shows the importance of the security of entropy sources (in the forms of chaotic analog circuits), and the trust aspect of the design process of random hardware generators as a combination of the true- and pseudo-random modules.
ACKNOWLEDGMENT
The authors would also like to thank the three anonymous reviewers for their constructive and valuable comments on an earlier version of the paper.

REFERENCES
[1] M. E. Yalcin, J. A. K. Suykens, and J. Vandewalle, “True random bit generation from a double-scroll attractor,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 51, no. 7, pp. 1395–1404, Jul. 2004.
[2] E. Barker, L. Feldman, and G. Witte, “Recommendation for random number generation using deterministic random bit generators,” Nat. Inst. Standards Technol., Gaithersburg, MD, USA, Tech. Rep. 800-90A Rev 1, 2015.
[3] W. Marszalek and H. Podhaisky, “2D bifurcations and Newtonian properties of memristive Chua’s circuits,” EPL Europhys. Lett., vol. 113, no. 1, p. 10005, Jan. 2016.
[4] A. Everspaugh, Y. Zhai, R. Jellinek, T. Ristenpart, and M. Swift, “Not-so-random numbers in virtualized Linux and the Whirlwind RNG,” in Proc. IEEE Symp. Secur. Privacy, San Jose, CA, USA, May 2014, pp. 559–574.
[5] H. Corrigian-Gibbs and S. Jana, “Recommendations for randomness in the operating system, or how to keep evil children out of your pool and other random facts,” in Proc. 15th Workshop Hot Topics Oper. Syst., Kartause Ittingen, Switzerland, May 2015, pp. 1–8.
[6] R. Kumari, M. Alimomeni, and R. Safavi-Naini, “Performance analysis of Linux RNG in virtualized environments,” in Proc. ACM Workshop Cloud Comput. Secur. Workshop, Devo, CO, USA, Oct. 2015, pp. 29–39, doi: 10.1145/2808426.2808434.
[7] J. Wang, J. Fan, and X. Wu, “The entropy source of pseudo random number generators: From low entropy to high entropy,” in Proc. IEEE Int. Conf. Intell. Secur. Informat. (ISI), Jul. 2019, pp. 161–163, doi: 10.1109/ISI.2019.8823457.
[8] L. Coppolino, S. D’Antonio, G. Mazzeo, and L. Romano, “A comprehensive survey of hardware-assisted security: From the edge to the cloud,” Internet Things, vol. 6, Jun. 2019, Art. no. 100055.
[9] S. Cohneye, A. Kwong, S. Paz, D. Genkin, N. Heninger, E. Ronen, and Y. Yarom, “Pseudorandom black swans: Cache attacks on CTR_DRBG,” in Proc. IEEE Symp. Secur. Privacy (SP), San Francisco, CA, USA, May 2020, pp. 1241–1258.
[10] T. Gao, T. Li, R. Jiang, M. Yang, and R. Zhu, “Research on cloud service security measurement based on information entropy,” IJ Netw. Secur., vol. 21, pp. 1003–1013, 2019.
[11] M. Tehranipoor and F. Koushanfar, “A survey of hardware Trojan taxonomy and detection,” IEEE Design Test Comput., vol. 27, no. 1, pp. 10–25, Jan. 2010.
[12] M. Melosik and W. Marszalek, “Using the 0-1 test for chaos to detect hardware Trojans,” in Proc. IEEE Design Test Comput., vol. 27, no. 1, pp. 919–921, Mar. 2016.
[13] M. Melosik, P. Sniatala, and W. Marszalek, “Hardware Trojans detection through Obfuscation,” IEEE Design Test Comput., vol. 35, no. 1, pp. 80–109, Jan. 2014.
[14] Z. Esfes and T. B. Ward, “The emergence of novel attributes in concept modification,” Creativity Res. J., vol. 14, no. 2, pp. 149–156, Apr. 2002.
[15] WIESLAW MARSZALEK received the Ph.D. and D.Sc. degrees in electrical engineering from the Warsaw University of Technology, Poland, and the Ph.D. degree in applied mathematics from North Carolina State University, Raleigh, NC, USA. In the past, he held various research and teaching positions at the state and private universities in North Carolina, OH, USA, and NJ, USA. He was a Humboldt Research Fellow in Bochum, Germany, from 1990 to 1992, Hamburg, Germany, in 1996, and Halle, Germany, in 2011, and a Fulbright Research Fellow in Warsaw, Poland, from 2005 to 2006 and Opole, Poland, in 2018. He currently teaches computer science with the Opole University of Technology, Poland and mathematics with Rutgers University, NJ, USA. He has published over 100 journal and conference papers in the areas of 2D discrete equations, differential-algebraic equations, nonlinear circuits and electric arcs, memristors, hardware Trojans, and singularity induced bifurcations and chaos.

MICHAIL MELOSIK received the B.Sc. and M.Sc. degrees in computer engineering from the Poznan University of Technology, in 2007 and 2009, respectively, and the Ph.D. degree (Hons.) in computer science, in 2017. His research is related to the threat of unauthorized insertion of hardware Trojans into integrated circuits and reprogrammable circuits. His Ph.D. thesis being recognized with an award by the Prime Minister of Poland. His research interests include the area of hardware embedded systems and interdisciplinary applications of microcomputer hardware particularly in the field of security and trust, hardware Trojans, random number generators, VLSI systems, and application of chaotic systems in seed generators.

161853