Detection of Cybersecurity Events Based on Entropy Analysis

Andrii Bigdan[1], Tetiana Babenko[1], Hryhorii Hnatiienko[1], Oleksii Baranovskyi[2], and Larysa Myrutenko[1]

[1] *Taras Shevchenko National University of Kyiv, 64/13 Volodymyrska St., Kyiv, 01601, Ukraine*
[2] *Blekinge Tekniska Högskola, 371 79 Karlskrona, Sweden*

Abstract

As a rule, modern approaches to protecting against cyberattacks do not guarantee the impossibility of compromising applications and operating systems. Therefore, detection and identification of vulnerabilities, and actions to avoid or mitigate their impact on businesses and cybersecurity processes are critical for the operation of information systems and the information security management system. To identify a possible attack vector, as a rule, the following methods could be applied: either those that allow detecting abuses or that allow detecting anomalies. This paper investigates the possibility of identifying the alleged attack vector based on the entropy analysis of cybersecurity events. The research results presented in the paper allow us to determine the required width of the sliding window and confirm that such entropy analysis detects exceeding security thresholds and anomalies in the operation of operating systems and applications and, accordingly, probable attack vectors.

Keywords

Entropy, anomaly, event log, information security, intrusion detection, cybersecurity event

1. Introduction

According to the World Economic Forum's report on global risks, cyber-attacks and data theft are expected to remain among the most long-term risks that businesses will face over the next ten years and, accordingly, enterprises of various forms of ownership need to ensure a given level of cyber security service delivery [1]. Cyber-attacks are becoming more complex and destructive, which leads to the disruption of information systems, including critical information infrastructure. Therefore, studies that allow timely identification of deviations in the operation of processes that in the future may lead to the implementation of a certain attack vector and compromise of information systems are relevant [2-9]. One of the promising areas of research is utilizing indicators of entropy to assess various parameters of information systems cybersecurity, as evidenced by a significant number of relevant scientific papers.

To detect attacks, the following groups of methods are used [2]: based on storing behavior examples; frequency; neural network; implementing finite-state machines; other special. These methods are also divided into abuse-detecting and anomaly-detecting methods. Abuses are based on the use of existing flaws in the information system, and an anomaly is a deviation from the normal state of the system, such unusual activity in it may indicate certain attacking actions [3]. The advantage of anomaly detection methods is the ability to identify new attacks without modifying or updating the model parameters in the case when the behavior of the system during the attack is statistically different from the system's normal behavior [4]. Besides anomalies in the network, there are anomalies in the event log on the hosts that can also indicate the occurrence of unauthorized activity. Anomalies in the system operation can be indicated by the number of different records that can be analyzed using entropy

Proceedings of the 7th International Conference on Digital Technologies in Education, Science and Industry (DTESI 2022), October 20-21, 2022, Almaty, Kazakhstan
EMAIL: abigdan@gmail.com (Andrii Bigdan); babenko.tetiana.v@gmail.com (Tetiana Babenko); g.gna5@ukr.net (Hryhorii Hnatiienko); oleksii.baranovskyi@bth.se (Oleksii Baranovskyi); myrutenko.lara@gmail.com (Larysa Myrutenko)
ORCID: 0000-0002-2940-6085 (Andrii Bigdan); 0000-0003-1184-9483 (Tetiana Babenko); 0000-0002-0465-5018 (Hryhorii Hnatiienko); 0000-0001-5629-5205 (Oleksii Baranovskyi); 0000-0003-1686-261X (Larysa Myrutenko)

calculation. Using several methods at the same time, i.e., an integrated approach to identifying and analyzing cybersecurity incidents, it is possible to increase the effectiveness of counteracting abuses in the field of cybersecurity and intrusion [2].

2. Literature review

Entropy is “a measure of the randomness or diversity of a data-generating function. Data with full entropy is completely random and no meaningful patterns can be found.” [10]. Entropy is a degree of uncertainty. The chaos level in the data can be calculated using the entropy of the system [11]. According to the authors of [11–14], the availability and use of high-entropy data sources can theoretically be crucial for ensuring cybersecurity. As a rule, the sources of entropy in information systems are keyboard inputs, mouse movement parameters, reading data from disks, the voltage on structural elements, temperature indicators, etc. The authors of [12,13] utilize entropy in the analysis of encrypted and packed malware to detect malware and malware accident investigation. Malware can easily avoid signature-based detection by using packing or encryption methods. So, packed file detection is also important. As known, detection methods can be divided into signature-based and entropy-based detection [13]. [11] presents the results of using entropy analysis to identify abuses in digital marketing, namely, to determine the subject of interaction with information systems (bot or human). The solution proposed by the authors is a JavaScript detection tag that collects hundreds of parameters from the browser, and it provides a fairly simple way to determine the power of the central processor on various devices (server, laptop, phone), and identifies the same kinds of bots (not new ones). The approach proposed by the authors does not allow determining the type of interaction subject with 100% accuracy, but, in our opinion, it is promising, in particular, for detecting the compromise of end systems and unauthorized use of their computing resources, including for cryptocurrency mining. The authors of [14] used entropy as a statistical indicator of applications’ operation designed to monitor the network, in particular, to detect anomalies in network traffic. They proceeded from the fact that it is quite difficult to estimate the entropy of data streams due to the complexity of entropy calculation.

In [11], the results of research on risk assessment of security systems based on the calculation of the Shannon entropy for calculating systemic risk are presented. According to the authors of [14], study [15] presents the most optimal algorithm for estimating entropy and determining the size of the sliding window, provided that the sample for research is random and corresponds to the specifics of data distribution in the network traffic flow. Accordingly, determining the "correct" size of the sliding window in the process of entropy calculation remains one of the problematic issues in applying entropy indicators to analyze the security of cyber-physical systems and identify information security anomalies. If we calculate the entropy of all sources of events in the information system, then when logging a new security event, it is necessary to calculate the probabilities of all previously logged message types. The method is not suitable for calculating the entropy of a source that constantly generates new message types.

3. Problem statement

This paper investigates the possibility of identifying the alleged attack vector based on the entropy analysis of cybersecurity events in the operating system and application systems. Anomaly detection in the operation of the application process/operating system was performed based on the comparison of the reference entropy with the entropy of the application process/operating system after performing unauthorized actions. The research was conducted in the environment Windows 10 operating system one of the popular user systems [16].

4. Model implementation

As known [17], the Windows event logging service uses the data stored in the "EventLog" registry key, which, in turn, contains subkeys that store system operation logs. The main logging data of the Windows operating system are records of applications, system processes, and security processes.

Processes of application can log events associated with them in the standard log of the corresponding application processes, created when the corresponding program is installed. Within the framework of this study, a detailed analysis of the structure and content of operation logs the following logs were used: application processes, security systems, and PowerShell Operational, which stores, in particular, information about console commands. At the same time, it was considered that most of the records in the event logs during normal system operations are "Microsoft Windows security auditing" messages about logging in and logging out of users, granting them rights, fetching credentials; and also taking into account the fact that messages about security events may not be logged in case an attacker blocked the logging of events security.

Entropy values were calculated based on the PID (process ID) and PPID (parent process ID) of the process and the event code. As known [18], by default, the Windows operating system does not log events about the following activities: creating, modifying, or deleting files; creating processes, or modifying the registry. So, to register these events the Sysmon service was leveraged. Sysmon, once installed on a system, remains resident across system reboots and allows monitoring and logging of cybersecurity events to the Windows log. Also, this tool allows us to get detailed information about the launch of processes, network connections, and changes in the file creation time [19]. The Sysmon service was used in the standard configuration [20]. At this stage of the research, network connections were not analyzed, so the entries of the "NetworkConnect" type were disabled.

The entropy of the event source was calculated using the Shannon formula [21]:

$$H(A) = -\sum_{i=1}^{k} p_i \log_2 p_i, \tag{1}$$

where A – set of messages, k – the number of messages, p_i – the probability of occurrence of each message.

To calculate the probability of a message appearing, the following formula was used:

$$p_i = \frac{n_i}{N}, \tag{2}$$

where n_i – the number of messages of a certain type, N – the total number of messages.

The sliding window method was used to calculate the entropy of the event log [22]. The entropy was calculated for messages within the selected window of size W. For the first W messages, the entropy is calculated by the formula (1), then when new messages appear, the window is shifted by dW.

Figure 1 shows a scheme of the sliding window method with $W = 5$ and $dW = 1$, the different message types are numbered.

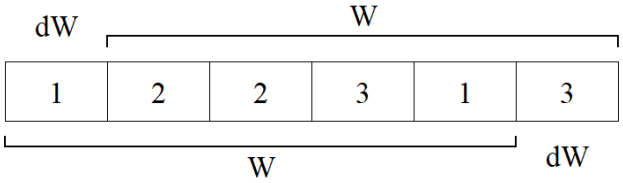

Figure 1: Scheme of the sliding window method

After shifting the window, the current entropy value was calculated:

$$H_i = H_{i-1} + \Delta H, \tag{3}$$

The entropy change was calculated by the formula:

$$\Delta H = H_i - H_{i-1}, \tag{4}$$

Substituting formula (1) into formula (4), we obtained the expression:

$$\Delta H = -\sum_{i=1}^{k} p_i \log_2 p_i + \sum_{j=1}^{l} p'_j \log_2 p'_j, \tag{5}$$

where letters without a stroke are the probability of the value of messages for the current window, with a stroke – for the previous window.

With this approach, elements with no changed probability in the current and previous windows will not affect the change in entropy, therefore, it will depend only on the elements that have entered and left the window. In Figure 1, after shifting the window, the probability of elements 1 and 3 will change, while element 2 will remain unchanged. If we designate the previous values of the probabilities of elements 1 and 3 as p_{11} and p_{31}, and the current values as p_{12} and p_{32}, then the entropy change will be calculated by the formula:

$$\Delta H = -p_{12}\log_2 p_{12} - p_{32}\log_2 p_{32} + p_{11}\log_2 p_{11} + p_{31}\log_2 p_{31}, \quad (6)$$

That is, for elements with changed probability we need to subtract the entropy before the window shift and add the entropy after the shift. To calculate the entropy of Windows event logging according to the described algorithm, a Python program was developed. To receive messages from the event log, the "EvtSubscribe" function [23], which is part of the Win32 API, was used.

Figure 2 shows the entropy during normal system operation. Entropy values were calculated based on the PID of the process, and for Sysmon messages - based on the event code. The size of the sliding window was 100, which made it possible to cover the events to be logged as fully as possible.

The analysis of the obtained research results showed that in the occurrence of a significant number of cybersecurity events of the same type, the entropy value approaches zero. If the number of cybersecurity events is greater than the size of the sliding window, the entropy value will be zero. Therefore, the following heuristics were introduced in further studies.

Heuristic E1. By examining the graph of changes in the entropy of the message source, it is possible to identify an attack with high certainty.

Heuristic E2. For complete identification, there must be several signs of the presence of an attack vector. The prerequisites for an attack are a deep drop in the level of entropy and a large increase in the level of entropy. But if these two features are not related, these features are not sufficient to consider them one by one as mandatory features of entropy.

Heuristic E3. The attack is accompanied by a large range of entropy reduction at the beginning of the attack and a large increase in the entropy level at the end of the attack.

Further, a heuristic will also be introduced to guarantee the capture of the attack vector. Within the framework of this study, several options for expert determination of the window width were considered:

1. Expert survey of information systems security administrators.
2. To determine the width of the window, which is guaranteed to contain information about the attack, several investigations were carried out, in which the beginning and end of the attack were fixed by experts. The window width was determined based on the statistical analysis of the obtained data. The width of the window should be considered in several aspects: when studying trends in entropy change; when detecting anomalous behavior of the function values; for guaranteed localization of the event, while the event consists of related messages/entries.

When studying the behavior of the function that describes the change in entropy, at least several factors were distinguished: the number of events in the window; a variety of different types of events in the window; interval of entropy changes within the window.

The attack detection window (ADW) is considered to be the number of recorded events during which it is possible to reliably determine the signs of the beginning of an attack, the maximum difference in entropy values, and the signs of the completion of an attack.

The window for determining standard events (WDSE) is considered to be the number of recorded events, during which it is possible to determine the beginning, reaching the minimum/maximum entropy, and the end of the event. The width of the window in all cases was measured by the number of recorded events. Even based on visual observations, it can be concluded that almost always ADW>WDSE.

As known, determining the size of a sliding window when calculating entropy remains one of the problematic issues in applying entropy indicators to analyze the security of cyber-physical systems and detect cybersecurity anomalies. Determining the required width of the sliding window makes it possible to automate the analysis of the behavior of a function that reflects the level of entropy. To ensure the further automatic investigation of the behavior of the function that describes the value of entropy in the system, it is necessary to investigate: identifying the behavior of the function graph during attacks; identifying common features of attacks displayed on the graph; finding the boundaries of the beginning and end of the attack.

An attack can be investigated by studying the behavioral trends, in particular, the time series. As known, trends are described using linear, logarithmic, power, and other equations. Let's consider an approach that allows us to identify a rapid change in trend behavior from the very first steps of the emergence of a certain trend. Let a series of events be given, the sequence of which will be denoted by indices $i \in T = \{1, \dots, t\}$. The entropy value for each event will be denoted by

$$x_i, i \in T, \tag{7}$$

To study the patterns of behavior of the values of series (7), we set WDSE τ, for example, in the interval $\tau \in \left[1, \frac{t}{2}\right]$. For each event $i \in T$, we will determine the values of the ratio:

$$r_{ij} = \frac{x_i}{x_{i+j}}, \tag{8}$$

where $j = 1, \dots, \tau$.

In situations where there may be zero values among the entropy values, sufficiently small values $\varepsilon > 0$ can be added to the denominator in formula (8):

$$r_{ij} = \frac{x_i}{\left(x_{i+j} + \varepsilon\right)}, \tag{9}$$

where $j = 1, \dots, \tau$.

Signs of a change in the trend with a sharp decrease in entropy values are the presence within the width window (8) among the values of the form (9) that are significantly greater than the values of the series (7):

$$\exists r_{ij} >> \underset{l \in T}{max}(x_l), \tag{10}$$

Signs of a change in the trend with a sharp increase in entropy values are the presence among the values of the form (9) and those that are significantly lower than the inverse values of the series (7):

$$\exists \left(\frac{1}{r_{ij}}\right) << \underset{l \in T}{min}\left(\frac{1}{x_l}\right), \tag{11}$$

The presence of several values (10) or (11) among the values of the form (9) is the criteria for a sharp change in the trend. Depending on the width of the window, it is possible to determine the trend change at different stages.

Table 1

Example of window width with a trend change

t	1	2	3	4	5	6
1	1,17					
2	0,83	0,83				
3	0,86	0,83	1			
4	1,17	1,2	0,83	1,17		
5	1,20	1,0	1,20	0,56	0,88	
6	0,71	1,00	0,56	0,75	0,25	0,55
7	1,17	0,56	0,88	0,35	0,66	0,20
8	0,67	0,88	0,30	0,66	0,24	0,79
9	1,13	0,55	0,98	0,3	1,18	0,35
10	0,73	1,13	0,48	1,58	0,40	9,45
11	1,38	0,89	2,48	0,89	**17,30**	1,48
12	0,89	2,20	1,19	**19,80**	2,37	**69,30**
13	1,80	1,33	**19,8**	3,56	**89,10**	5,33
14	0,83	9,00	2,22	**49,5**	4,44	**149,00**
15	6,00	1,67	**27,00**	3,33	**99,00**	13,30
16	0,33	3,00	0,42	9,00	1,67	**24,80**

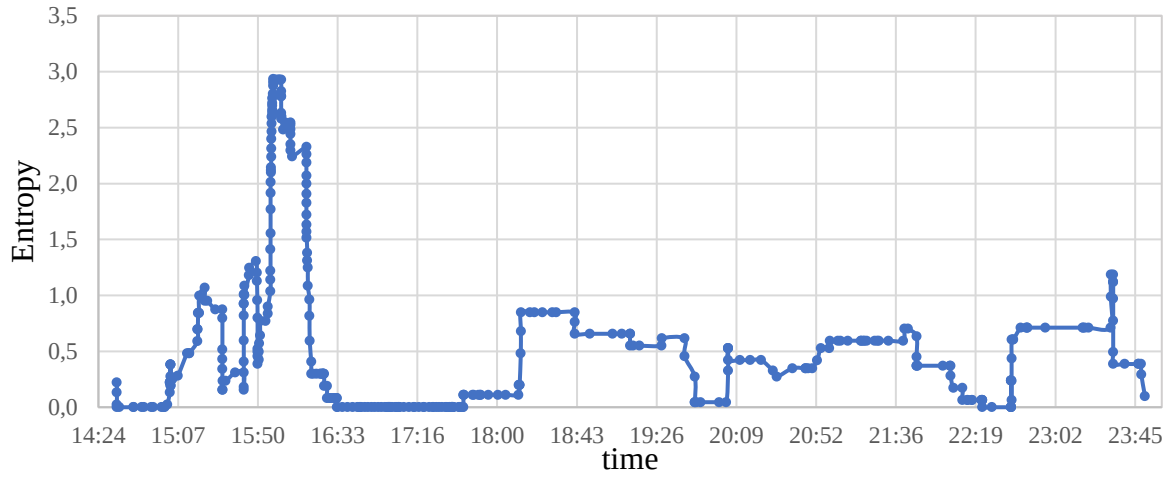

Figure 2: Entropy during normal system operation and security event logging enabled

Figure 3 shows simulation results with cybersecurity event logging disabled. The average value of entropy is 2.04961, and the variance is 2.45429, which is significantly higher than the average value of entropy and variance obtained in the opposite case.

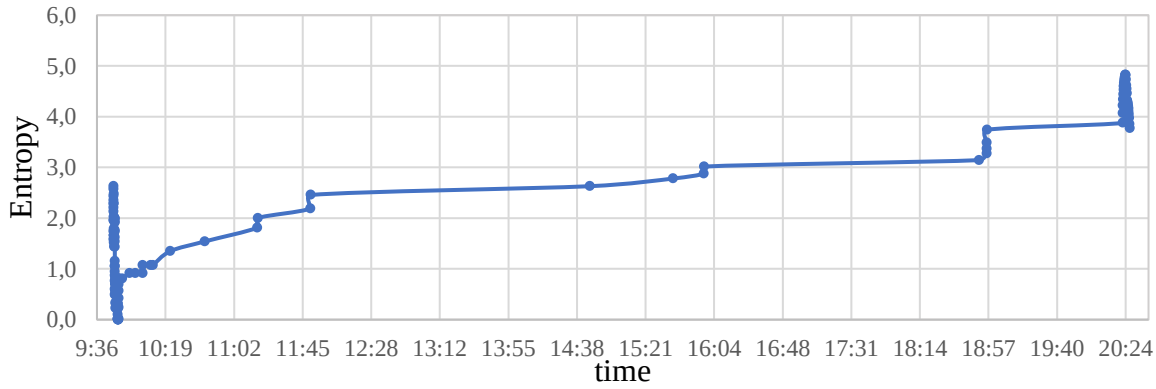

Figure 3: Entropy values when logging cybersecurity events disabled

Figure 4 shows a graph of the entropy changes during the regular operation of the system and the logging of cybersecurity events by the Sysmon process. The average value of entropy is 1.53181, the minimum is 0.58508.

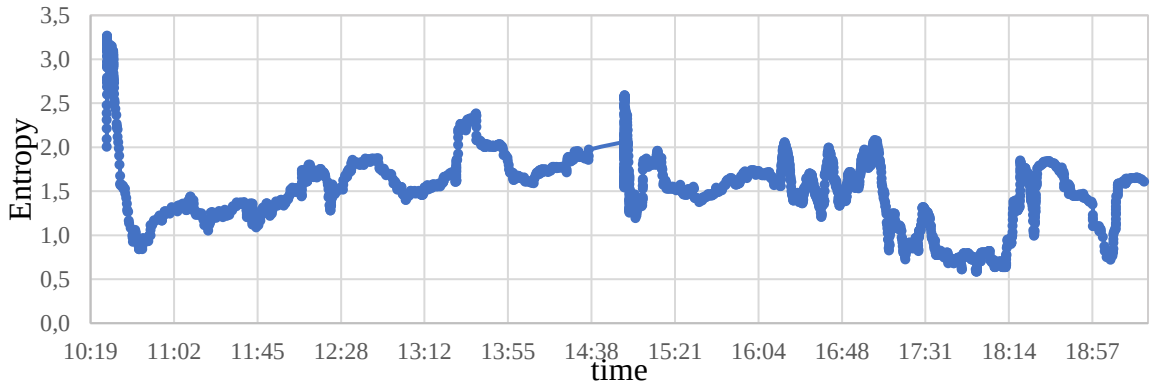

Figure 4: Entropy during normal system operation and logging of cybersecurity events by the Sysmon process

At the next stage of the research, unauthorized actions were simulated in a system with an active Sysmon service. As part of the implementation of this stage, numerous files were written, and many processes were created.

The creation of a significant number of files or processes may indicate a local DoS (Denial of Service) attack aimed at degrading the performance of the hardware and software system, and, as a result, disrupting availability. Simultaneous actions with numerous files may also indicate unauthorized actions, such as the execution of a ransomware virus. To simulate the generation of cybersecurity events in the browser, a significant number of pages were opened, which are typically stored so that they can be restored the next time the browser is launched. Figure 5 shows a graph of the entropy changes when replacing descriptors for numerous files. A similar dependence was obtained for the case of creating and executing many processes.

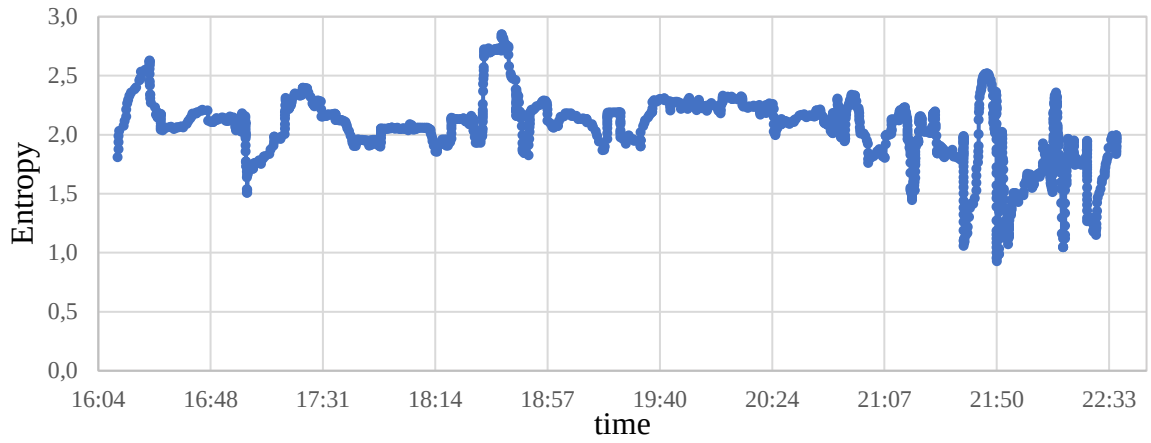

Figure 5: Entropy values when creating a significant number of processes

Analysis of the research results showed that the average value of the entropy of the system, obtained based on the analysis of cybersecurity events, is greater than the entropy value when security events are not recorded. Thus, it was concluded that by comparing the average value of the entropy of the system for a certain period with the value accepted as a reference, it is possible to detect the disabling of logging of cybersecurity events, which may indicate, among other things, a certain phase of the implementation of the attack vector.

Then the ability was tested for immediate detection of the security events logging turned off. To disable logging, the command "auditpol /clear" must be executed. This command is designed to remove the security audit settings for all users and, accordingly, processes running on their behalf. Figure 6 shows the entropy values after disabling cybersecurity event logging.

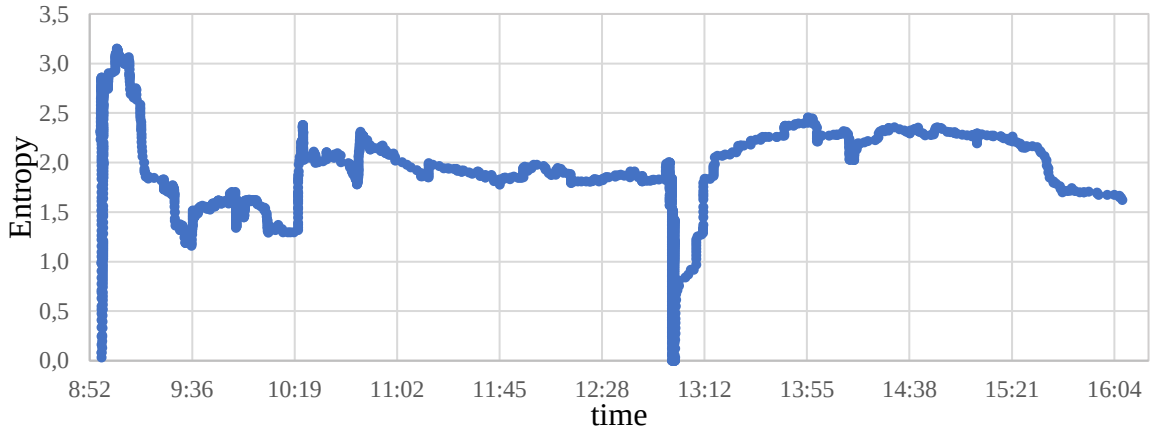

Figure 6: Entropy values after disabling cybersecurity events logging

The first zero value of entropy is due to a large number of records about reading account data and granting rights during user login. The following zero value is caused by a large number of audit policy change records.

We also studied whether the value of the entropy is affected by attacks aimed at obtaining unauthorized access to the system via a reverse TCP connection. An attack was performed using

Metasploit on a Linux virtual machine which involves remotely connecting to the target system and exploiting "UAC bypass" privilege escalation, which in turn would grant "System" user rights:

- "Meterpreter reverse TCP" payload [24-27] was created and uploaded to the target system.
- Once opened, the file establishes a connection to Metasploit, creating a session that provides access to the target system.
- To obtain superuser or "System" rights, privileges were escalated using the "FodHelper UAC bypass" exploit [28], which allows bypassing Windows user access control.

The successful result of the attack is a remote connection to the target system and the execution of privilege escalation. The attack scheme that was used in the simulation process is shown in Figure 7.

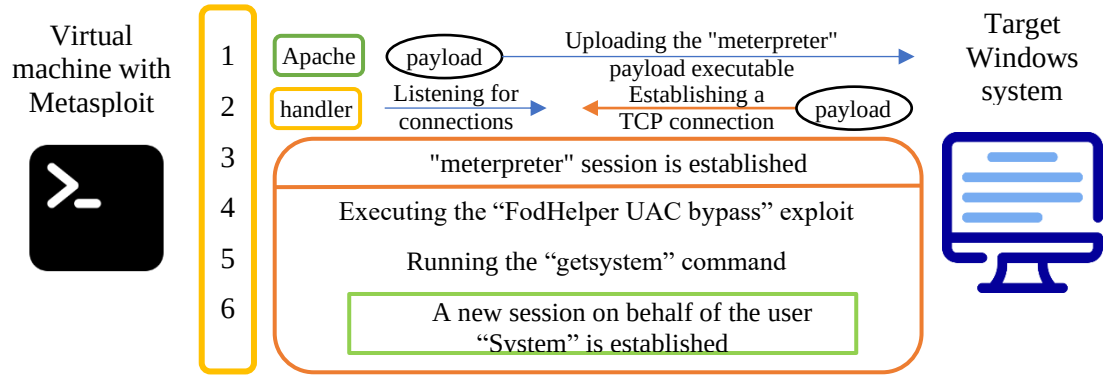

Figure 7: Attack implementation scheme

Figure 8 shows the entropy values at a sliding window size of 300 during normal system operation and the attack execution. Average value – 1.89971, minimum – 0.09664, maximum – 3.14670.

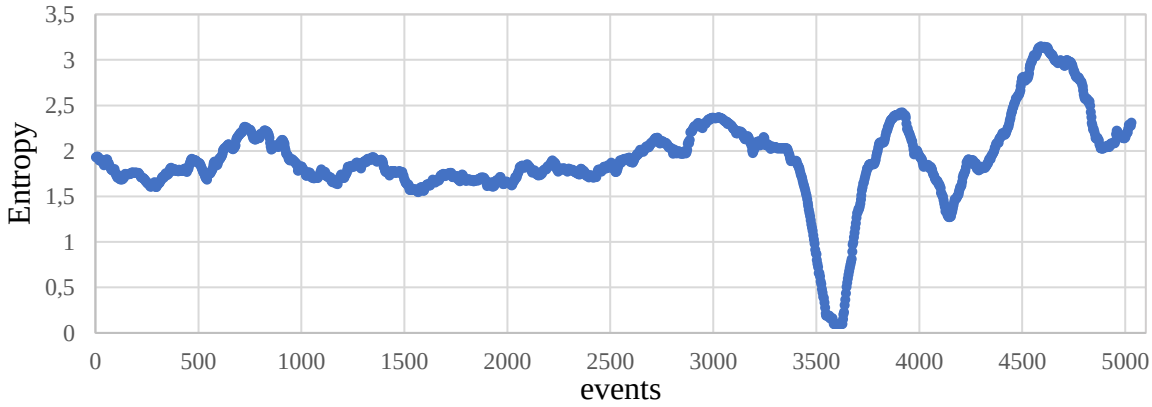

Figure 8: Entropy values during the attack execution

During the attack execution, the entropy value was the highest for the entire observation period, which is due to the occurrence of cybersecurity events such as modifying the registry and executing commands in the console. Entropy values close to zero are caused by cybersecurity events related to reading credentials. Figure 9 shows the deviation of the entropy values from the reference values observed during the implementation of the attack vector.

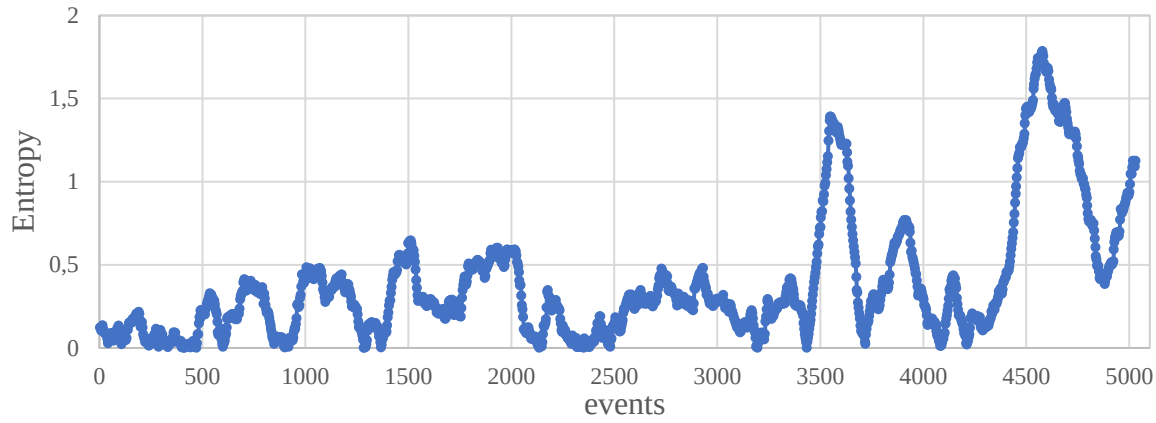

Figure 9: Entropy deviation from reference values

5. Conclusions

The analysis of the obtained results allows us to conclude that such events significantly affect the value of the entropy of the system; accordingly, large values may indicate the execution of unauthorized actions in the system.

Thus, the proposed approaches in this work allow the detection of anomalous conditions, such as one-time recording of a large number of files, launching a large number of processes, stopping security event logging, gaining unauthorized remote access to the system, and privilege escalation. When developing the attack lifecycle, the entropy value differs significantly from the average values (observed during normal system operation), and, accordingly, this method of analyzing security event logging allows you to detect the exceeding of defined security thresholds, which may indicate the presence of anomalies or abuses. The proposed approach to determining the width of the sliding window makes it possible to identify with great accuracy the presence of a deviation from the normal operation of the system and to determine the development trends of incomprehensible events or processes. These solutions can be integrated into intrusion detection systems or other security controls.

6. Reference

[1] The Global Risks Report 2022, 17th Edition. World Economic Forum, 25-26, 11 January 2022. URL: https://www3.weforum.org/docs/WEF_The_Global_Risks_Report_2022.pdf.

[2] O. M. Kolodchak, Suchasni metody vyiavlennia anomalii v systemakh vyiavlennia vtorhnen, Visnyk Natsionalnoho un-t Lvivska politekhnika, Kompiuterni systemy ta merezhi (Ukrainian) 745 (2012) 98-104. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2017/nov/6726/16-98-104.pdf.

[3] D. Palko, L. Myrutenko, T. Babenko, A. Bigdan, Model of Information Security Critical Incident Risk Assessment, in: Proceedings of the IEEE International Conference on Problems of Infocommunications Science and Technology, PIC S and T, 2020, pp. 157-161.

[4] D. Palko, H. Hnatienko, T. Babenko, A. Bigdan, Determining key risks for modern distributed information systems, in: CEUR Workshop Proceedings, 3018, 2021, pp. 81-100.

[5] S. Kazmirchuk, A. Korchenko, T. Parashchuk, Analiz system vyiavlennia vtorhnen, Zakhyst informatsii (Ukrainian) 20.4 (2018) 259-276. URL: https://doi.org/10.18372/2410-7840.20.13425.

[6] V. V. Bierkovskyi, O. S. Bezsonov, Analiz ta klasyfikatsiia metodiv vyiavlennia vtorhnen v informatsiinu system, Systemy upravlinnia, navihatsii ta zviazku (Ukrainian) 3 (2017) 7-62. URL: http://nbuv.gov.ua/UJRN/suntz_2017_3_17.

[7] I. V. Ruban, V. O. Martovytskyi, S. O. Partyka, Klasyfikatsiia metodiv vyiavlennia anomalii v informatsiinykh systemakh. Systemy ozbroiennia i viiskova tekhnika (Ukrainian) 3 (2016) 100-105. URL: http://nbuv.gov.ua/UJRN/soivt_2016_3_24.

[8] O. Panasko, S. Burmistrov, Praktychni aspekty upravlinnia intsydentamy informatsiinoi bezpeky. Hraal nauky (Ukrainian) 5 (2021) 164-166. URL: https://ojs.ukrlogos.in.ua/index.php/grail-of-science/article/download/13105/12042.
[9] T. Radivilova, Vyiavlennia anomalii v telekomunikatsiinomu trafiku statystychnymy metodamy, Kiberbezpeka: osvita, nauka, tekhnika (Ukrainian) 11.3 (2021) 183-194. URL: https://doi.org/10.28925/2663-4023.2021.11.183194.
[10] T. W. Edgar, D. O. Manz, Research Methods for Cyber Security, Syngress, Cambridge, 49, 2017.
[11] A. Subasi, Practical Machine Learning for Data Analysis Using Python, Academic Press, 78, 2020. URL: https://doi.org/10.1016/C2019-0-03019-1.
[12] R. Lyda, J. Hamrock, Using Entropy Analysis to Find Encrypted and Packed Malware, IEEE Security and Privacy Magazine 5.2 (2017) 40-45. URL: https://doi.org/10.1109/MSP.2007.48.
[13] S.-W. Han, S. Lee, Packed PE File Detection for Malware Forensics, The KIPS Transactions Part C 16C.5 (2009). URL: https://doi.org/10.3745/KIPSTC.2009.16C.5.555.
[14] Y. Liu, Y. Guan, Distributed Network and System Monitoring for Securing Cyber-Physical Infrastructure, Handbook on Securing Cyber-Physical Critical Infrastructure, 2012, 455-479. URL: http://dx.doi.org/10.1016/B978-0-12-415815-3.00018-2.
[15] A. Chakrabarti, G. Cormode, A. McGregor, A near-optimal algorithm for computing the entropy of a stream, in: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete algorithms, SODA 07, 2007, pp. 328-335.
[16] A. V. Ilienko, S. S. Ilienko, T. M. Kulish, Perspektyvni metody zakhystu operatsiinoi systemy Windows, Kiberbezpeka: osvita, nauka, tekhnika (Ukrainian) 4 (2020) 124-134. URL: https://doi.org/10.28925/2663-4023.2020.8.124134.
[17] Eventlog Key - Win32 apps. Developer tools, technical documentation, and coding examples. Microsoft Docs, 20 August 2021. URL: https://docs.microsoft.com/uk-ua/windows/win32/eventlog/eventlog-key.
[18] A. S. Tanenbaum, H. Bos, Modern Operating Systems, 4th. ed, Pearson Education, 2015.
[19] M. Russinovich, T. Garnier, Sysmon - Windows Sysinternals. Developer tools, technical documentation and coding examples, Microsoft Docs, 11 May 2022. URL: https://docs.microsoft.com/uk-ua/sysinternals/downloads/sysmon.
[20] SwiftOnSecurity. Sysmon configuration file template with default high-quality event tracing. GitHub, 17 October 2021. URL: https://github.com/SwiftOnSecurity/sysmon-config.
[21] Y. P. Zhurakovskyi, V. P. Poltorak, Teoriia informatsii ta koduvannia: Pidruchnyk, Vyshcha shk (Ukrainian) 32 (2001).
[22] O. Gudkov, Calculation Algorithm for Network Flow Parameters Entropy in Anomaly Detection. IT Security for the Next Generation, International Round, Delft University of Technology, May 2012.
[23] EvtSubscribe function (winevt.h) - Win32 apps. Developer tools, technical documentation and coding examples. Microsoft Docs, 13 October 2021. URL: https://docs.microsoft.com/en-us/windows/win32/api/winevt/nf-winevt-evtsubscribe.
[24] Metasploit-Framework. Kali Linux Tools. Kali Linux, 05 August 2022. URL: https://www.kali.org/tools/metasploit-framework/.
[25] Working with payloads. Metasploit documentation. Rapid7 Docs, July 2022. URL: https://docs.rapid7.com/metasploit/working-with-payloads.
[26] About the Metasploit meterpreter. Offensive security. Infosec Training & Penetration Testing. Offensive Security, 02 November 2019. URL: https://www.offensive-security.com/metasploit-unleashed/about-meterpreter/.
[27] Meterpreter getsystem. Metasploit Documentation. Rapid7 Docs, July 2022. URL: https://docs.rapid7.com/metasploit/meterpreter-getsystem/.
[28] Windows UAC protection bypass (via fodhelper registry key). Rapid7 Docs, 30 May 2018. URL: https://www.rapid7.com/db/modules/exploit/windows/local/bypassuac_fodhelper/.