Definition of Cryptojacking Indicators

Tetiana Babenko[1,2], Kateryna Kolesnikova[1], Rostyslav Lisnevskyi[1,2], Shakirt Makilenov[1,3] and Yuriy Landovsky[2]

[1]*International Information Technology University, Manas St. 34/1, Almaty, 050040, Kazakhstan*
[2]*Taras Shevchenko National University of Kyiv, Volodymyrska St. 64/13, Kyiv, 01601, Ukraine*
[3]*Al-Farabi Kazakh National University, al-Farabi Avenue 71, Almaty, 050000, Kazakhstan*

Abstract
This article explores and defines cryptojacking indicators, which are necessary to detect and suppress illegal activities in the field of cryptocurrency mining. The study includes the analysis of characteristic signs and properties of cryptojacking, as well as a review of modern detection methods and research methodologies. Application of the obtained indicators allows the detection of cybersecurity anomalies related to unauthorized cryptojacking on end systems.

Keywords
Cryptomining, cryptojacking, information security risk, distributed information systems, risk factors, security controls, risk control techniques, risk modeling, risk management

1. Introduction

The increasing volume of information processed and transmitted between different information systems, organisations and individual users increasingly depends on the continuity and accuracy of these processes. With each passing year, cybercriminals are becoming more adept at exploiting software vulnerabilities. According to Rapid7's 2021 Vulnerability Report, the time to known exploitation (TTKE) has decreased by 71%. The average time to exploit a vulnerability has dropped significantly from 42 days in 2020 to just 12 days in 2021[1].

In order to respond to new threats in information systems in a timely manner, it is necessary to have tools that allow you to analyze security risks in real-time and mitigate their impact on assets. Each individual asset has a large attack surface, as it is constantly exposed to a large number of external and internal risk factors [2,3]. In addition, within the framework of modern distributed systems, such assets are not isolated, but are in constant interaction with other network nodes, end users, or are exposed to the external environment. Another aspect of the situation is the presence of a large number of known vulnerabilities, or zero-day vulnerabilities, which also significantly affects the profile of potential threats.

From a technical perspective, cryptocurrency mining involves continuous computation of specific hash functions. A cryptographic hash function (CHF) is a mathematical algorithm that maps data of arbitrary size (often referred to as a "message") into a fixed-size binary array ("hash value," "hash," or "message digest"). It is a one-way function, meaning it cannot be inverted. Ideally, the only way to find such a message is to use a rainbow table of corresponding hashes [4].

Very often, attacks on endpoint systems are realized through the exploitation of vulnerabilities in system and application software. Starting from attacks on browser vulnerabilities when a user loads an infected web page, and ending with the delivery of a malicious payload to an end system by exploiting vulnerabilities in network protocols and operating systems [5,6].

DTESI 2023: Proceedings of the 8th International Conference on Digital Technologies in Education, Science and Industry, December 06–07, 2023, Almaty, Kazakhstan

t.babenko@iitu.edu.kz (T. Babenko); kkolesnikova@iitu.edu.kz (K. Kolesnikova); r.lisnevskyi@iitu.edu.kz (R. Lisnevskyi); sh.makilenov@iitu.edu.kz (Sh. Makilenov); mr.lavandovskiy@gmail.com (Y. Landovsky)
0000-0003-1184-9483 (T. Babenko); 0000-0002-9160-5982 (K. Kolesnikova); 20000-0002-9006-6366 (R. Lisnevskyi); 0009-0006-4786-0149 (Sh. Makilenov); 0009-0006-8709-2383 (Y. Landovsky)

Despite the wide range of capabilities of modern EPP platforms, the number of information security incidents related to the compromise of end systems and their subsequent use for crypto mining (Cryptojacking) has increased significantly.

Cryptojacking is the process of using the victim's computing power without consent for cryptocurrency mining. Cryptojacking usually leads to a decrease in system performance and an increase in energy costs [6-9].

Detection and Identification of Cryptojacking is a challenging task, as perpetrators conducting Cryptojacking attacks often attempt to conceal their activities. Currently, there are several indicators and methods that can be employed to detect Cryptojacking activity, including [9-20]:

- Detecting anomalies in resource usage, including monitoring the central processing unit (CPU) and graphics processing unit (GPU) load, as increased CPU or GPU load may indicate the execution of mining scripts.
- Studying processes that use an abnormally large amount of computing resources since Cryptojacking scripts can consume a significant amount of RAM and energy.
- Monitoring network activity to detect suspicious activity, such as communication with known mining pools. High network traffic associated with mining can be an indicator of Cryptojacking.
- Checking system processes and services on the end system to identify unknown or suspicious processes running in the background.
- Monitoring the browser to detect unauthorized script loading and execution or identifying suspicious browser extensions for the detection of extensions that could run Cryptojacking scripts.
- Monitoring blockchain transactions.
- Utilizing integrated solutions. Antivirus and anti-malware software can sometimes detect and block malicious scripts and programs associated with Cryptojacking.
- Analyzing system logs and cybersecurity event logs.

Based on the above, a combination of appropriate methods and solutions is required to solve the problem of detecting and identifying Cryptojacking activity. Since the task of detecting Cryptojacking activity, especially in distributed information systems, contains uncertainties and cannot be well described by analytical methods, solutions based on artificial intelligence, in particular neural networks, are increasingly used in scientific works on this topic. In this paper, we will study the possibility of finding indicators of unauthorized crypto activity based on the analysis of operations performed by the application and network activity.

2. Approaches review

In the field of cryptojacking, most of the proposed detection methods use dynamic analysis. The main reason for this is that mining scripts use a set of known instructions. For example, miners use cryptographic hash libraries and repeatedly increment the value of a static variable (i.e. nonce), or connect to some known service providers to keep downloading computation results and receive new tasks. These typical actions of cryptojacking malware create a pattern that allows them to be detected by dynamic analysis. An analysis of research in the area of detecting Cryptojacking activity showed that there are only a few studies that use static features, such as operation codes [21] and WebAssembly (Wasm) instructions. WebAssembly [21] is a low-level instruction format that allows programs to run closer to a machine-level language and deliver higher performance using stack-based virtual machines. This low-level instruction model allows WebAssembly to execute code more efficiently, and this feature provides more profit because the cryptojacking script runs faster. All major browsers on the market now support WebAssembly. The detection system proposed in [21] uses transaction codes for static analysis, where transaction codes are extracted using IDA Pro. The system proposed by the authors of [21] detects cases of compromise of the end system for cryptomining with an accuracy of 95%.

Analysis of CPU-related events [9-24] are the most commonly used parameters for detecting cryptojacking based on dynamic analysis, since cryptojacking scripts need to receive CPU instructions to perform mining, regardless of the hardware used. Although CPUs are a typical feature of cryptocurrency mining, using CPU events alone can lead to high false positive rates (FPRs), as flash game websites or online rendering websites also heavily use CPUs for their operations. The analysis of memory usage for Cryptojacking is widely presented in and is another commonly used feature of cryptojacking

In [21], system calls were used to detect the process of unauthorized Cryptojacking. System calls are executed with level 0 privileges to make calls and request services from the OS kernel. The proposed detection system in [21] uses system calls that are then used to train deep learning models, and they achieve 99% accuracy.

It should be noted that despite the widespread use of neural networks to detect and identify various types of attacks on endpoints or intermediate elements of information and communication systems, there are few published scientific studies on their use to detect and identify cryptojacking. For example, in [15], a solution is proposed, but in a rather unrealistic scenario that does not take into account encrypted traffic, a reduced set of traditional machine learning models is proposed, using a set of uninformative features as input data.

Paper [16] proposes a method for detecting cryptojacking based on signs of provider network behavior. The authors identified cryptojacking features from the first four packets in the stream and developed a neural network classifier that accurately and efficiently detects signs of cryptojacking traffic. The authors of [17] proposed to use neural networks to analyze assembly code that is platform- and high-level programming language-independent. The paper presents the results of numerical experiments with different neural network architectures, which show that the neural network apparatus is a highly effective tool for detecting cryptojacking attacks and allows achieving high classification accuracy even with limited training data. The research work [18] focuses on providing a solution for detecting malicious cryptomining activity by passively monitoring the network and identifying cryptomining characteristics. The authors of [9] deployed a complex and realistic cryptomining scenario for training and testing machine learning models, in which clients interact with real servers via the Internet and use encrypted connections. The analysis of the research results led to the conclusion that the use of machine learning models can detect cryptomining attacks even if the traffic is encrypted.

In their study [19], the authors evaluated the effectiveness of machine learning methods on the task of real-time traffic classification and concluded that decision tree-based classifiers are the most effective for classifying network traffic. The authors of [19] modeled real-time classification using traffic features obtained from the first few packets of each stream.

It should be noted that most research works are focused on detecting unauthorized Cryptojacking processes. However, there are several solutions to combat cryptojacking, namely, open-source browser extensions such as NoCoin [25] and MinerBlock are used in browsers. These extensions are based on blacklists that are updated when new mining domains are discovered. The browser, in turn, warns the user that a blacklisted website is being accessed. Blocking based on a blacklist is an ineffective way to combat cryptojacking, as attackers can change target domains and, accordingly, the effectiveness of blacklists. A method of dynamic blacklisting was proposed in [24], but as shown in [25], this solution also has limited effectiveness.

It should be noted that most research works are focused on detecting unauthorized Cryptojacking processes. However, there are several solutions to combat cryptojacking, namely, open-source browser extensions such as NoCoin [25] and MinerBlock. These extensions are based on blacklists that are updated when new mining domains are discovered. The browser, in turn, warns the user that a blacklisted website is being accessed. Blacklist-based blocking is an ineffective way to combat cryptojacking, as attackers can change the target domains and, consequently, the effectiveness of blacklists.

In this study, the analysis of operations performed by the application process and the analysis of network traffic will be used to find identifiers of unauthorized cryptomining.

3. Cryptojacking scenarios

As known from the literature [6-24], mining is the process of creating new blocks and recording transactions in a cryptocurrency blockchain. It is a fundamental component of the operation of most cryptocurrencies and ensures network security and decentralization. The mining process can vary depending on the specific cryptocurrency and the algorithm employed. However, in general terms, it can be described as follows:

- Transaction Collection: Network users send transactions (cryptocurrency transfers) using their wallets. These transactions are gathered in pools (mempools), where they await processing by miners.
- Block Creation: Miners compete for the right to create a new block in the blockchain. To achieve this, they solve a complex mathematical problem known as "Proof of Work" (PoW) or, in some cases, "Proof of Stake" (PoS).
- Task Solving: Miners solve a task that requires significant computational effort. In PoW systems, this may involve finding a value (hash) for a new block that meets specific conditions. This process is called "finding a nonce." In PoS systems, miners prove their stake in the cryptocurrency based on the number of coins they hold as collateral.
- Adding a Block: The first miner who successfully solves the task gains the right to create a new block. This block includes the collected transactions from the mempool, information about the previous block, and the results of solving the task (nonce or other proofs).
- Transaction Confirmation: The created block is broadcasted throughout the cryptocurrency network. Network participants (nodes) verify the block and its contents. If everything is correct, the block is confirmed and added to the blockchain.
- Reward: The miner who successfully creates a new block receives a reward in the form of cryptocurrency. This includes transaction fees from the transactions included in the block and, in the case of Bitcoin, newly minted bitcoins issued as a block reward.
- Repetition: The mining process repeats itself, creating new blocks and processing new transactions continuously.

Accordingly, crypto miners have their technological specificities and distinctions. To conduct a comprehensive study, this work examined several examples of crypto miners. The research also took into account that different cryptocurrencies use different hash functions.

During the initial phase of this research, patterns related to Shift, XOR, and Rotate operations were detected. Their identification is essential for the following reasons:

- Code encryption and obfuscation: Malicious actors employ these operations to conceal malicious code and complicate detection. It is crucial to uncover these operations as they might be part of encryption and obfuscation mechanisms that ensure anonymity and hinder the comprehension of malicious code.
- Compromise indicator identification: Analyzing Shift, XOR, and Rotate operations can reveal specific patterns that indicate the use of malicious code. The presence of such operations in malicious software code can serve as an indicator of system compromise.
- Detection of specific attack variations: In some cases, Shift, XOR, and Rotate operations may point to particular variations of cryptojacking or methods used by malicious actors. Revealing these characteristics allows for the analysis and prevention of new attack types.

In this context, it should be noted that various hash functions are used for obtaining cryptocurrencies, namely: SHA-256, Scrypt, Cryptonight, Ethash, X11. When examining hash functions from a machine code perspective, it primarily involves bit operations, such as bit permutations and shifts. Therefore, it can be hypothesized that applications involved in cryptocurrency mining will exhibit an abnormal number of operations related to bit manipulation. Within the entire set of operations on the 'x86' architecture, the following operation categories are identified: Rotate, Shift, Logical Exclusive OR.

Rotate shifts the bits of the first operand (destination operand) by the number of bit positions specified in the second operand (count operand) and stores the result in the destination operand. The destination operand can be a register or a memory location, and the count operand is an unsigned integer that can be either immediate or the value in the CL register. The processor restricts the count to a range from 0 to 31, masking all bits in the count operand except the 5 least significant bits. The following operations fall into this category: RCL, RCR, ROL, ROR.

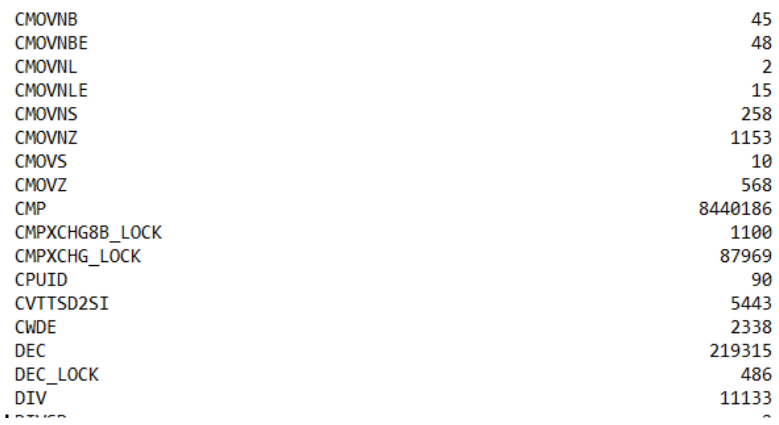
Shift moves the bits in the first operand (destination operand) to the left or right by the number of bits specified in the second operand (count operand). Bits that are shifted beyond the bounds of the destination operand are initially shifted into the CF flag and then discarded. At the end of the shift operation, the CF flag contains the last bit shifted out of the destination operand. The destination operand can be a register or a memory location, and the count operand can be an immediate value or the CL register. The count is masked to 5 bits, limiting the count range from 0 to 31. Special operation code encoding is provided for counting by 1. The following operations are part of this category: SHL, SHR, SHRD, SHLD.

Logical Exclusive OR performs a bitwise exclusive OR (XOR) operation on the destination (first) and source (second) operands and stores the result in the destination operand. The source operand can be an immediate value, a register, or a memory location; the destination operand can be a register or a memory location. (However, two memory operands cannot be used in a single instruction.) Each bit in the result is set to 1 if the corresponding bits of the operands are different; each bit is set to 0 if the corresponding bits are the same.

4. Preparing data for modelling

For the investigation and determination of the number of operations performed during the execution of a process (software product), Intel Software Development Emulator (Intel SDE) was utilized. Intel SDE is an emulator that enables running code with instruction sets on systems that do not support these instructions. It's worth noting that SDE is useful for assessing functionality but not performance since it executes programs much slower than if they were run on actual hardware. SDE only takes into account the instructions that are executed dynamically during the program's operation. It is a very effective way to debug a program. In this regard, if a specific block of addresses corresponding to certain parts of the code doesn't exhibit the expected instructions during execution, it can serve as an indicator of unexpected/unintended branching conditions. Running the program with different parameters or inputs also contributes to dynamic execution, so it cannot be assumed that a single program run reproduces the entire history.

After executing the SDE operation, a file named "out_notepad.txt" is generated, which contains information about loaded libraries, most common instruction blocks, executed functions, and performed operations. An example of this information can be seen in the illustration provided in Figure 1.



```
CMOVNB                                                     45
CMOVNBE                                                    48
CMOVNL                                                      2
CMOVNLE                                                    15
CMOVNS                                                    258
CMOVNZ                                                   1153
CMOVS                                                      10
CMOVZ                                                     568
CMP                                                   8440186
CMPXCHG8B_LOCK                                           1100
CMPXCHG_LOCK                                            87969
CPUID                                                      90
CVTTSD2SI                                                5443
CWDE                                                     2338
DEC                                                    219315
DEC_LOCK                                                  486
DIV                                                     11133
```

Figure 1: A fragment of the SDE source file with a list of operations

The key information for analysis is contained at the end of the file, where a list of all executed instructions during the program's operation is provided. An example of such a list can be seen in Figure 1. SDE allows you to filter a significant amount of information, so the resulting command looked like this: sde.exe -mix_omit_per_function_stats 1 -top_blocks 0 -mix_max_cumulative 0 -mix_omit_per_hread_stats 1 -omix out_notepad.txt -- notepad.exe.

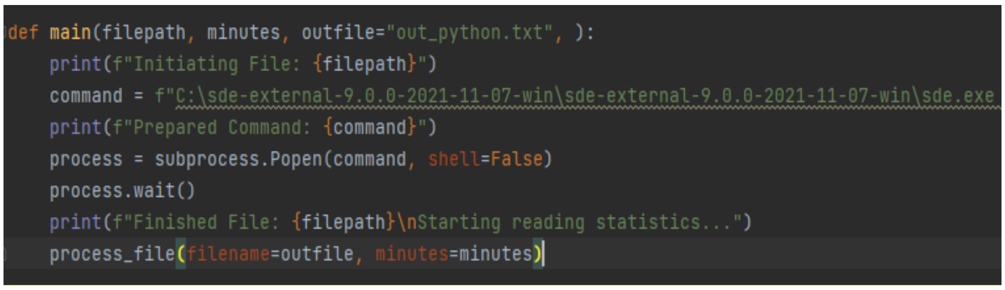
To conduct these studies, SDE was set up to collect information only on global transactions, and therefore did not collect statistics for individual flows.

In the process of further research, we developed an application that performs such operations:

- Launch the SDE program together with the target research software.
- Open the file where the SDE results are stored.
- Analyzing and parsing the results.
- Visualize the information obtained as a result of the analysis.

First, the script receives information about the application to be examined, and then sets a name for the output file. Next, it prepares a command for SDE, taking into account all the optimization flags, and executes this command directly. The program will not continue execution until the running process is completed. The subsequent phase involves opening the generated file, extracting information from it, and conducting parsing. During this phase, specific operations were identified, encompassing RCL, RCR, ROL, ROR, SHR, SHL, SHRD, SHLD, and XOR. The program's "main" function utilized in this research is illustrated in Figure 2.

To illustrate the results of the study, we used the Matplotlib library [91]. Matplotlib is a powerful tool for data visualization in Python. This library allows you to create graphs and charts, is open source and available for use. Matplotlib is an alternative to other software solutions, such as MATLAB, and provides the ability to integrate graphical representations into user program interfaces through APIs (application programming interfaces). The Matplotlib library helped us visualize the results of our research and present them in an understandable way.

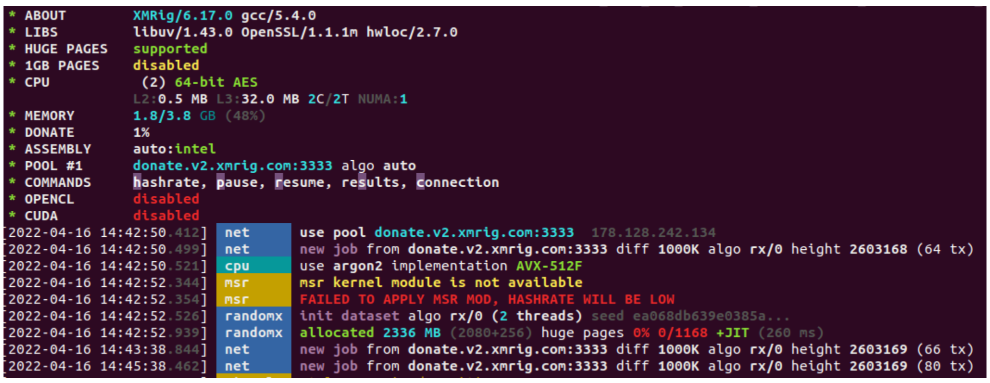


```python
def main(filepath, minutes, outfile="out_python.txt", ):
    print(f"Initiating File: {filepath}")
    command = f"C:\sde-external-9.0.0-2021-11-07-win\sde-external-9.0.0-2021-11-07-win\sde.exe
    print(f"Prepared Command: {command}")
    process = subprocess.Popen(command, shell=False)
    process.wait()
    print(f"Finished File: {filepath}\nStarting reading statistics...")
    process_file(filename=outfile, minutes=minutes)
```

Figure 2: The "main" function of the program

5. Analysis of software behavior

At the initial stage of research, we analyzed a cryptominer called XmRig. XmRig is a high-performance open-source software tool that is cross-platform and designed to mine cryptocurrency using the computing power of processors (CPU) and graphics processing units (GPU). The tool is available for use on operating systems such as Windows, Linux, macOS, and FreeBSD. Figure 3 shows the console interface of the XMrig miner.



```
 * ABOUT        XMRig/6.17.0 gcc/5.4.0
 * LIBS         libuv/1.43.0 OpenSSL/1.1.1m hwloc/2.7.0
 * HUGE PAGES   supported
 * 1GB PAGES    disabled
 * CPU           (2) 64-bit AES
                L2:0.5 MB L3:32.0 MB 2C/2T NUMA:1
 * MEMORY       1.8/3.8 GB (48%)
 * DONATE       1%
 * ASSEMBLY     auto:intel
 * POOL #1      donate.v2.xmrig.com:3333 algo auto
 * COMMANDS     hashrate, pause, resume, results, connection
 * OPENCL       disabled
 * CUDA         disabled
[2022-04-16 14:42:50.412]  net      use pool donate.v2.xmrig.com:3333  178.128.242.134
[2022-04-16 14:42:50.499]  net      new job from donate.v2.xmrig.com:3333 diff 1000K algo rx/0 height 2603168 (64 tx)
[2022-04-16 14:42:50.521]  cpu      use argon2 implementation AVX-512F
[2022-04-16 14:42:52.344]  msr      msr kernel module is not available
[2022-04-16 14:42:52.354]  msr      FAILED TO APPLY MSR MOD, HASHRATE WILL BE LOW
[2022-04-16 14:42:52.526]  randomx  init dataset algo rx/0 (2 threads) seed ea068db639e0385a...
[2022-04-16 14:42:52.939]  randomx  allocated 2336 MB (2080+256) huge pages 0% 0/1168 +JIT (260 ms)
[2022-04-16 14:43:38.844]  net      new job from donate.v2.xmrig.com:3333 diff 1000K algo rx/0 height 2603169 (66 tx)
[2022-04-16 14:45:38.462]  net      new job from donate.v2.xmrig.com:3333 diff 1000K algo rx/0 height 2603169 (80 tx)
```

Figure 3: XMRig console interface

XmRig was executed for 2 minutes. The ratio of Shift, XOR, and Rotate operations to all operations in the XMRig cryptominer is shown in Figure 4.

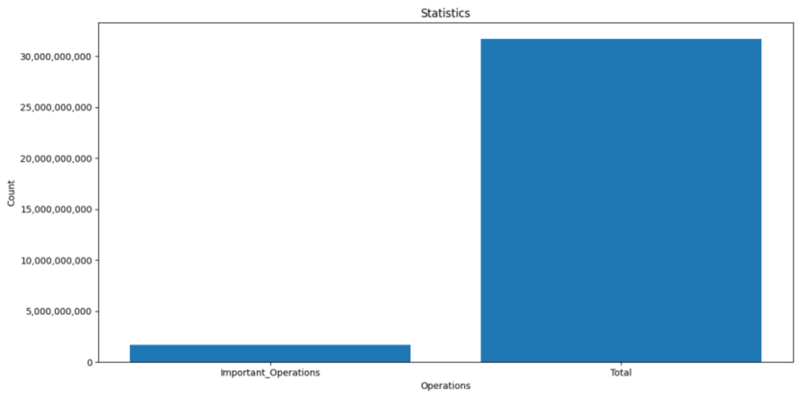


Figure 4: The ratio of Shift, XOR, and Rotate operations to all operations in the XMRig crypto miner

Over 30 billion operations were performed in a period of 2 minutes. About 5% of all operations were allocated to Shift, Rotate, and XOR operations The distribution of operations is shown in Figure 5.

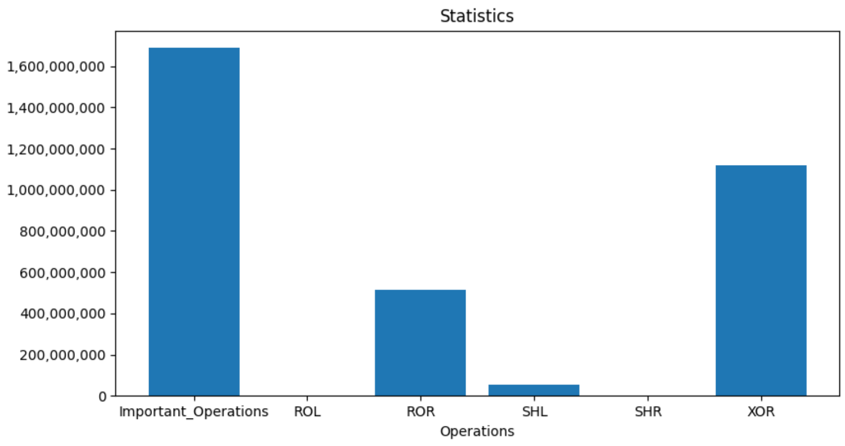


Figure 5: Distribution of operations in the XMRig crypto miner

As a result of the analysis, it was found that the vast majority of operations are Rotate (1.62% of the total number of operations) and XOR (3.53% of the total number of operations). Although Shift operations can be used to calculate hash functions, in this context, they account for only 0.18% of the total number of operations.

Similarly, we analyzed the work of a miner known as cpuminer. Cpuminer is a multi-threaded, highly optimized miner aimed at using central processing units (CPUs) to mine cryptocurrencies, including Litecoin, Bitcoin, and others. The program implements the SHA-256d and scrypt(N, 1, 1) algorithms, which allow mining using different algorithms. Cpuminer also supports the get block template and Stratum mining protocols, which makes it possible to use it both for individual mining and for participating in community mining on the network. The analysis of the cpuminer program execution indicates its uniqueness and high optimization for cryptocurrency mining on central processing units (CPUs). One of the characteristic features is a large number of XOR operations, which make up a significant percentage of the total number of operations, namely 6.71%. On the other hand, Shift and Rotate operations are very limited in cpuminer, accounting for only 0.01% of all operations. This disproportion indicates a difference in data processing approaches compared to other cryptocurrency mining programs.

It is also important to analyze the overall ratio of all operations to important operations, which indicates the efficiency and specific characteristics of cpuminer in the context of cryptocurrency mining. The results of the analysis are shown in Figure 6. This analysis helps to understand how the program uses computing resources to achieve optimal results in cryptocurrency mining.

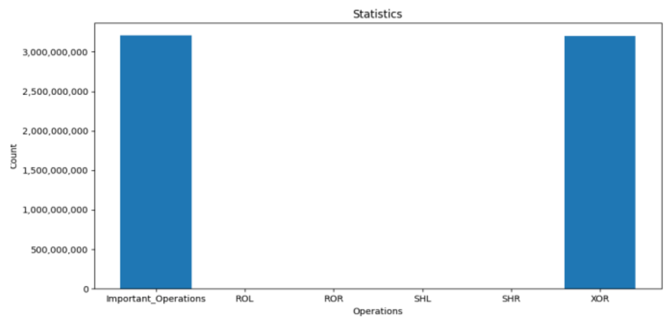


Figure 6: Distribution of operations in cpuminer

The study found that all three types of operations (Rotate, Shift, and XOR) are typical for the cryptomining process. It is difficult to identify clear patterns, as certain cryptominers may actively use Rotate operations, while others prefer Shift or XOR operations. However, the key aspect of the analysis is to relate all operations to those that play an important role in the context of cryptomining. In pure applications, the highlighted operations in total do not exceed 4% of all operations.

While in cryptominers, this ratio is always at least 5%. This indicates the specific nature of the operations used in cryptomining and their important role in the functioning of cryptominers. to identify suspicious activity, the following formula was used:

$$K = \frac{\sum O_{imp}}{\sum O_{all}} * 100,$$

where K is the percentage of Shift, Rotate, and XOR operations to all operations;

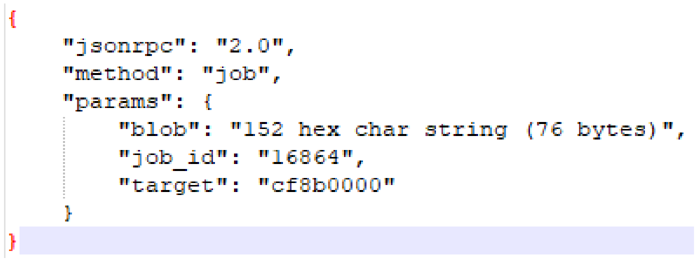
O_{imp} - an array of Shift, Rotate and XOR operations;

O_{all} - an array of all operations.

Given the results, we can conclude that the K-ratio is important in the context of application analysis. If the value of the K-ratio exceeds the threshold of 5%, it is considered an indicator that requires further in-depth consideration and analysis of the relevant application. This practice is justified by the fact that exceeding the specified threshold may indicate a potential anomaly in the application's operation. In particular, it is known that safe and harmless applications that do not require significant computing resources of the end system usually do not demonstrate such a high level of resource utilization. Therefore, a value of the K-ratio that exceeds 5% can be considered as a potential indicator of anomalies in the application's functioning. This approach helps to detect and identify applications that may have suspicious behavior or a negative impact on the end system.

6. Analysis of crypto miners' network activity

At this stage, we investigated network traffic related to cryptomining operations. As a rule, in the process of cryptomining, the Stratum protocol is used to interact between the client and the miner's pool server. The Stratum protocol is based on the use of a regular TCP socket and is used to exchange JSON-RPC messages. In practice, this means that the client establishes a connection to the server via a TCP socket and sends requests to the server in the form of JSON messages that end with the newline character "\n". Each received line on the client side is a valid JSON-RPC fragment containing a response to the sent request Stratum communication is initiated by the miner (client), who interacts with the mining pool server and goes through the authentication procedure. After successfully logging in to the mining pool system, the miner receives a task through notifications about new tasks, which have a common format, as illustrated in the figure 7.



```
{
    "jsonrpc": "2.0",
    "method": "job",
    "params": {
        "blob": "152 hex char string (76 bytes)",
        "job_id": "16864",
        "target": "cf8b0000"
    }
}
```

Figure 7: Stratum notification of a new job

Wireshark was used to analyze network traffic [26]. The analysis of network traffic based on the statistics obtained using Wireshark allowed us to conclude that the size of most packets sent by the cryptominer is 54 bytes long. For example, during the study, XMRig was launched for 25 minutes, during which time 194 requests were made. Among them, 67 requests were 54 bytes long, which is almost 35% of all requests made. Figure 8 shows the most common packet sizes for the XMRig cryptominer during 25 minutes of execution.

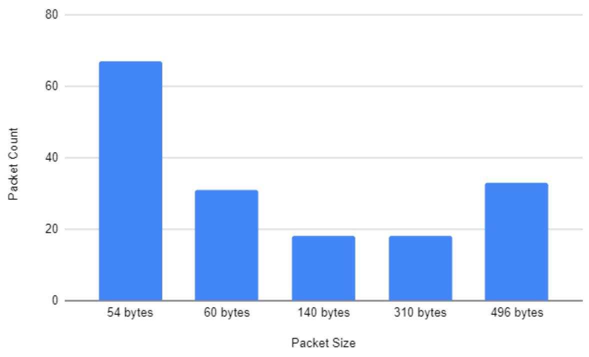


Figure 8: Example of packet length distribution of the XMRig crypto miner

The results of the traffic analysis of other cryptominers allow us to conclude that the presence of packets with a length of 54 bytes should be considered as an indicator of cryptomining.

7. Conclusion

In summary, the results of this study provide an important contribution to the field of cybersecurity and the detection of unauthorised cryptomining on endpoint systems. The identified indicators, such as the use of the Stratum protocol, packet length, Rotate, Shift and XOR operations, as well as their combination, allow us to identify potential threats and act proactively to protect systems from unauthorised cryptomining. These results can be useful for organisations and individual users seeking to improve the security of their computers and network infrastructures in the face of growing interest in crypto mining and cyber threats.

8. References

[1] M. Alvarez, D. Bales and J. Chung (2020). Ibm x-force threat intelligence index. Accessed: May, vol. 12, pp. 2020.
[2] John McCumber (2004). Assessing and Managing Security Risk in IT Systems: A Structured Methodology, New York, 288 p.

[3] Palko, D., Babenko, T., Bigdan, A., Gorbovy, O.,Borusiewicz, A. (2023). Cyber Security Risk Modeling in Distributed Information Systems. Applied Sciences (Switzerland), 13(4), 2393.

[4] F. Tschorsch, B. Scheuermann (2016). Bitcoin and beyond: A technical survey on decentralized digital currencies, IEEE Commun. Surv. Tutor. 18 (3): 2084–2123, http://dx.doi.org/10.1109/COMST.2016.2535718.

[5] S. Eskandari, A. Leoutsarakos, T. Mursch, J. Clark (2018). A first look at browserbased cryptojacking, in: 2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW), pp. 58–66, http://dx.doi.org/10.1109/EuroSPW. 2018.00014.

[6] S. Kethineni and Y. Cao (2020). The rise in popularity of cryptocurrency and associated criminal activity. International Criminal Justice Review, vol. 30, no. 3, pp. 325-344.

[7] R.K. Konoth, E. Vineti, V. Moonsamy, M. Lindorfer, C. Kruegel, H. Bos, G. Vigna (2018,). Minesweeper: An in-depth look into drive-by cryptocurrency mining and its defense. in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, in: CCS '18, ACM, New York, NY, USA, pp. 1714–1730, http://dx.doi.org/10.1145/3243734.3243858, URL: http://doi.acm. org/10.1145/3243734.3243858.

[8] M. Musch, C. Wressnegger, M. Johns, K. Rieck, Thieves in the browser: Web-based cryptojacking in the wild, in: Proceedings of the 14th International Conference on Availability, Reliability and Security, in: ARES '19, ACM, New York, NY, USA, 2019, pp. 4:1–4:10, http://dx.doi.org/10.1145/3339252. 3339261, URL: http://doi.acm.org/10.1145/3339252.3339261.

[9] A. Zareh and H. R. Shahriari, "Botcointrap: detection of bitcoin miner botnet using host based approach", 2018 15th International ISC (Iranian Society of Cryptology) Conference on Information Security and Cryptology (ISCISC), pp. 1-6, 2018.

[10] M. Conti, L.V. Mancini, R. Spolaor, N.V. Verde, Can't you hear me knocking: Identification of user actions on Android apps via traffic analysis, in: Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, in: CODASPY '15, ACM, New York, NY, USA, 2015, pp. 297–304, http://dx. doi.org/10.1145/2699026.2699119, URL: http://doi.acm.org/10.1145/2699026. 2699119.

[11] S. Zander, T. Nguyen, G. Armitage, Automated traffic classification and application identification using machine learning, in: The IEEE Conference on Local Computer Networks 30th Anniversary (LCN'05)L, 2005, pp. 250–257, http://dx.doi.org/10.1109/LCN.2005.35.

[12] R. Alshammari, A.N. Zincir-Heywood, Machine learning based encrypted traffic classification: Identifying ssh and skype, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, 2009, pp. 1–8, http://dx.doi.org/10.1109/CISDA.2009.5356534.

[13] Subhan Ullah, Tahir Ahmad, Rizwan Ahmad, Mudassar Aslam, "Prevention of Cryptojacking Attacks in Business and FinTech Applications", Handbook of Research on Cybersecurity Issues and Challenges for Business and FinTech Applications, pp.266, 2022.

[14] Ke Ye, Meng Shen, Zhenbo Gao, Liehuang Zhu, "Real-Time Detection of Cryptocurrency Mining Behavior", Blockchain and Trustworthy Systems, vol.1679, pp.278, 2022.

[15] J. Z. I. Munoz, J. Suarez-Varela and P. Barlet-Ros, "Detecting cryptocurrency miners with NetFlow/IPFIX network measurements", Proc. IEEE Int. Symp. Meas. Netw. (M&N), pp. 1-6, Jul. 2019.

[16] S.K. UmaMaheswaran, Dipesh Uike, K K Ramachandran, Tharangini A, T. Suba, Devvret Verma, "The Critical Understanding on the Emerging Threats and Defensive Aspects in Cryptocurrencies using Machine Learning Techniques", 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), pp.1938-1942, 2022.

[17] Mani; Myeongsu Kim; Bharat Bhargava; Pelin Angin; Ayça Deniz; Vikram Pasumarti, Malware Speaks! Deep Learning Based Assembly Code Processing for Detecting Evasive Cryptojacking Ganapathy IEEE Transactions on Dependable and Secure Computing Year: 2023 | Early Access Article | Publisher: IEEE.

[18] A. Pastor; A. Mozo; S. Vakaruk; D. Canavese; D.R. López; L. Regano; S. Gómez-Canaval; A. Lioy, Detection of Encrypted Cryptomining Malware Connections With Machine and Deep Learning. Access Article | Publisher: IEEE Year: 2020 | Volume: 8 .

[19] Y. Wang and S.-Z. Yu, "Machine learned real-time traffic classifiers", Proc. 2nd Int. Symp. Intell. Inf. Technol. Appl., vol. 3, pp. 449-454, Dec. 2008.

[20] Ke Ye, Meng Shen, Zhenbo Gao, Liehuang Zhu, "Real-Time Detection of Cryptocurrency Mining Behavior", Blockchain and Trustworthy Systems, vol.1679, pp.278, 2022.

[21] J. Cabrera-Arteaga, M. Monperrus, T. Toady, B. Baudr, WebAssembly diversification for malware evasion/ Computers & Security18 May 2023.

[22] Guangquan Xu, Wenyu Dong, Jun Xing, Wenqing Lei, Jian Liu, Lixiao Gong, Meiqi Feng, Xi Zheng, Shaoying Liu Delay-CJ: A novel cryptojacking covert attack method based on delayed strategy and its detection, Digital Communications and Networks Volume 9, Issue 5, October 2023, pp 1169-1179/ https://doi.org/10.1016/j.dcan.2022.04.030.

[23] R. Ning, C. Wang, C. Xin, J. Li, L. Zhu, H. Wu, Capjack: Capture in-browser crypto-jacking by deep capsule network through behavioral analysis, in: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 1873–1881, http://dx.doi.org/10.1109/INFOCOM.2019.8737381.

[24] Carlin, P. O'Kane, S. Sezer, J. Burgess, Detecting cryptomining using dynamic analysis, in: 2018 16th Annual Conference on Privacy, Security and Trust (PST), 2018, pp. 1–6, http://dx.doi.org/10.1109/PST.2018.8514167.

[25] Emad Badawi Guy-Vincent Jourdan Iosif-Viorel Onut. The “Bitcoin Generator” Scam Blockchain: Research and Applications14 April 2022.

[26] Nagendra Kumar Nainar, Ashish Panda./ Wireshark for Network Forensics: An Essential Guide for IT and Cloud Professionals. ISBN-13 (pbk): 978-1-4842-9000-2. https://doi.org/10.1007/978-1-4842-9001-9.