ҚАЗАҚСТАН РЕСПУБЛИКАСЫ ҒЫЛЫМ ЖӘНЕ ЖОҒАРЫ БІЛІМ МИНИСТРЛІГІ

«Л.Н. ГУМИЛЕВ АТЫНДАҒЫ ЕУРАЗИЯ ҰЛТТЫҚ УНИВЕРСИТЕТІ» КЕАҚ

Студенттер мен жас ғалымдардың
«ǴYLYM JÁNE BILIM - 2023»
XVIII Халықаралық ғылыми конференциясының
БАЯНДАМАЛАР ЖИНАҒЫ

СБОРНИК МАТЕРИАЛОВ
XVIII Международной научной конференции
студентов и молодых ученых
«ǴYLYM JÁNE BILIM - 2023»

PROCEEDINGS
of the XVIII International Scientific Conference
for students and young scholars
«ǴYLYM JÁNE BILIM - 2023»

2023
Астана

УДК 001+37
ББК 72+74
G99

«ǴYLYM JÁNE BILIM – 2023» студенттер мен жас ғалымдардың XVIII Халықаралық ғылыми конференциясы = XVIII Международная научная конференция студентов и молодых ученых «ǴYLYM JÁNE BILIM – 2023» = The XVIII International Scientific Conference for students and young scholars «ǴYLYM JÁNE BILIM – 2023». – Астана: – 6865 б. - қазақша, орысша, ағылшынша.

ISBN 978-601-337-871-8

Жинаққа студенттердің, магистранттардың, докоранттардың және жас ғалымдардың жаратылыстану-техникалық және гуманитарлық ғылымдардың өзекті мәселелері бойынша баяндамалары енгізілген.

The proceedings are the papers of students, undergraduates, doctoral students and young researchers on topical issues of natural and technical sciences and humanities.

В сборник вошли доклады студентов, магистрантов, докторантов и молодых ученых по актуальным вопросам естественно-технических и гуманитарных наук.

УДК 001+37
ББК 72+74

ISBN 978-601-337-871-8

Список использованных источников

1. Cuellar, J. "Frama-C: A Software Analysis Perspective." Formal Methods for Components and Objects. Springer, Berlin, Heidelberg, 2012. P.233-259.
2. Eigenmann, R. "Towards a Comprehensive Evaluation of Frama-C." Formal Techniques for Distributed Objects, Components, and Systems. Springer, Cham, 2015. P.92-106.
3. "Frama-C User Manual", https://frama-c.com/download/frama-c-user-manual.pdf.
4. Beringer, L. "Verified Low-Level Programming Embedded in Frama-C." Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages. ACM, 2017.
5. Boulmé, S. "Static Analysis of Concurrent C Programs with Frama-C. Formal Techniques for Distributed Objects, Components, and Systems. Springer, Cham, 2016. P.3-19.
6. Marché, C. "Why Frama-C?" Communications of the ACM 61.3. 2018. P.67-75.
7. Griesmayer J. and Grosu R. "Formal Methods for the Verification of Computer Systems", Springer, 2015.
8. "Formal Verification of C Programs with Frama-C", https://hal.archives-ouvertes.fr/hal-01224831/document.
9. Blanchard A. "Introduction to C program proof with Frama-C and its WP plugin", 2020. P. 212.

УДК: 004.855.5

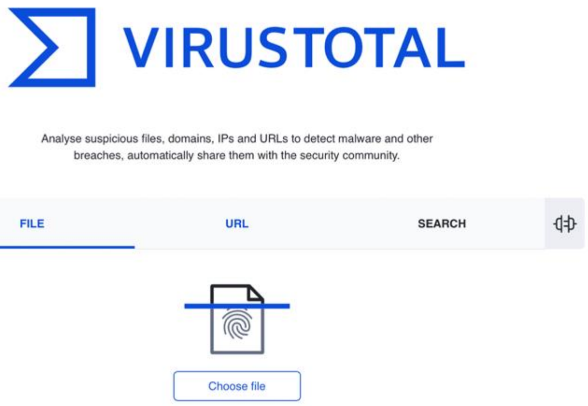
A SURVEY ON STATIC MALWARE ANALYSIS AND DETECTION METHODS

Mukhamadiyev Madiyar
mukhamadiyev_mg_1@enu.kz
MSc Student, L.N. Gumilyov Eurasian National University
Aldosh Balziya
b.nurgaliyeva@astanait.edu.kz
Teacher of Department of Intelligent Systems and Cybersecurity, Astana IT University, Astana, Kazakhstan
Konyrkhanova Assem
konyrkhanova_aa@enu.kz
Associate Professor of the Department of Information Security of the Information technologies faculty, L.N. Gumilyov National Eurasian University

Nowadays the world encounters numerous amounts of malwares, spreading and infecting IT fields, causing business process decline and data loss. To mitigate against malign programs, first must be defined the types of malicious software, which are divided into groups: ransomware, spyware, fileless malware, trojans, adware, worms, rootkits, and more. In this paper, we propose the first steps in automating the static analysis of a malware sample. Analyzing malware involves examining its behavior and characteristics to understand its functionality, purpose, and potential impact on a computer system.

Static malware analysis is a technique used to analyze malware without running it on a computer system. It involves examining the malware's code, structure, and behavior to understand its functionality, purpose, and potential impact on a computer system. Automated instruments such as VirusTotal [1] (Figure 1), which is an internet service and a scanning engine that analyzes suspicious files and accelerates the identification of viruses, worms, trojans and other types of malware detected by antivirus software, can assist in static analysis. The results of scanning files by the service do not depend on any particular antivirus software. In the computer world, a trojan horse is any software that misleads consumers about its true purpose. The story of the trojan horse, which was a trick that led to the destruction of the city of Troy in Ancient Greece, is where the phrase "Trojan Horse" comes from. A computer worm is a separate piece of malicious software that can be found on computers and that is designed to duplicate itself in order to infect other systems. It often spreads over computer networks, betting on security flaws on infected machines to gain access to their systems.

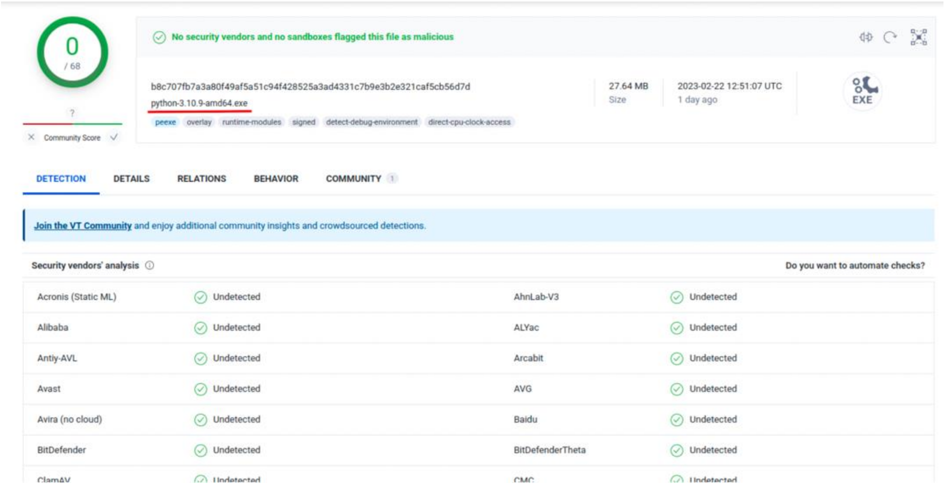
In the beginning of the process as the methodology it was taken a legitimate file [2] (Figure 2) to audit its structure and contagion. As a result, the file was marked clear and unmalicious.



Analyse suspicious files, domains, IPs and URLs to detect malware and other breaches, automatically share them with the security community.

Figure 1. VirusTotal a scanning engine

To proceed with a given task it was determined to load a malicious file from [3] to scan and verify capacity and efficiency of the search engine. As it was mentioned in Figure 3, several automated scanning rules have detected suspicious activities in the attached file. Parameters for examined file are: High risk 1: matches with the rule "Python Initiated Connection" by GitHub, Medium risk 2: matches with rules such as Disable Microsoft Defender Firewall via Registry, Suspicious command lines, and more. 53 out of 70 Antivirus signatures in total have concluded given file with a malicious program as the outcome.



No security vendors and no sandboxes flagged this file as malicious

b8c707fb7a3a80f49af5a51c94f428525a3ad4331c7b9e3b2e321caf5cb56d7d
python-3.10.9-amd64.exe
peexe overlay runtime-modules signed detect-debug-environment direct-cpu-clock-access

27.64 MB
Size

2023-02-22 12:51:07 UTC
1 day ago

Join the VT Community and enjoy additional community insights and crowdsourced detections.

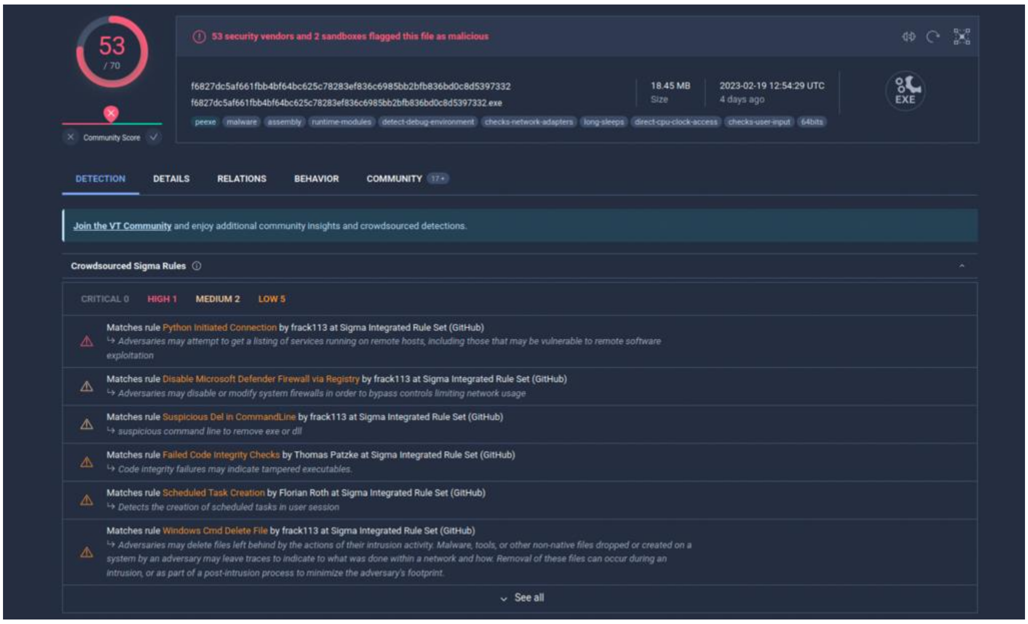
Security vendors' analysis ⓘ

Do you want to automate checks?

Acronis (Static ML)	✓ Undetected	AhnLab-V3	✓ Undetected
Alibaba	✓ Undetected	ALYac	✓ Undetected
Antiy-AVL	✓ Undetected	Arcabit	✓ Undetected
Avast	✓ Undetected	AVG	✓ Undetected
Avira (no cloud)	✓ Undetected	Baidu	✓ Undetected
BitDefender	✓ Undetected	BitDefenderTheta	✓ Undetected
ClamAV	✓ Undetected	CMC	✓ Undetected

Figure 2. Detection for patterns of an unmalicious file

By inspecting the result form Figure 3, VirusTotal reveals that this file is from the MagicRat family. The MagicRat malware is a Remote Access Trojan (RAT) that is mainly distributed by exploiting vulnerabilities such as Log4j in VMware Horizon. This type of malware is software designed to provide its creators with remote access and control over a compromised computer. RAT can infect systems in the same way as other forms of malware. They can be attached to an email, posted on a malicious website, or use a machine that has not been patched. RAT is designed to allow an attacker to remotely control a computer, similar to how Remote Desktop Protocol (RDP) and TeamViewer can be used for remote access and system administration.



53 security vendors and 2 sandboxes flagged this file as malicious

f6827dc5af661fbb4bf64bc625c78283ef836c6985bb2bfb836bd0c8d5397332
f6827dc5af661fbb4bf64bc625c78283ef836c6985bb2bfb836bd0c8d5397332.exe

18.45 MB
Size

2023-02-19 12:54:29 UTC
4 days ago

Join the VT Community and enjoy additional community insights and crowdsourced detections.

Crowdsourced Sigma Rules

CRITICAL 0 HIGH 1 MEDIUM 2 LOW 5

Matches rule Python Initiated Connection by frack113 at Sigma Integrated Rule Set (GitHub)
↳ Adversaries may attempt to get a listing of services running on remote hosts, including those that may be vulnerable to remote software exploitation

Matches rule Disable Microsoft Defender Firewall via Registry by frack113 at Sigma Integrated Rule Set (GitHub)
↳ Adversaries may disable or modify system firewalls in order to bypass controls limiting network usage

Matches rule Suspicious Del in CommandLine by frack113 at Sigma Integrated Rule Set (GitHub)
↳ suspicious command line to remove exe or dll

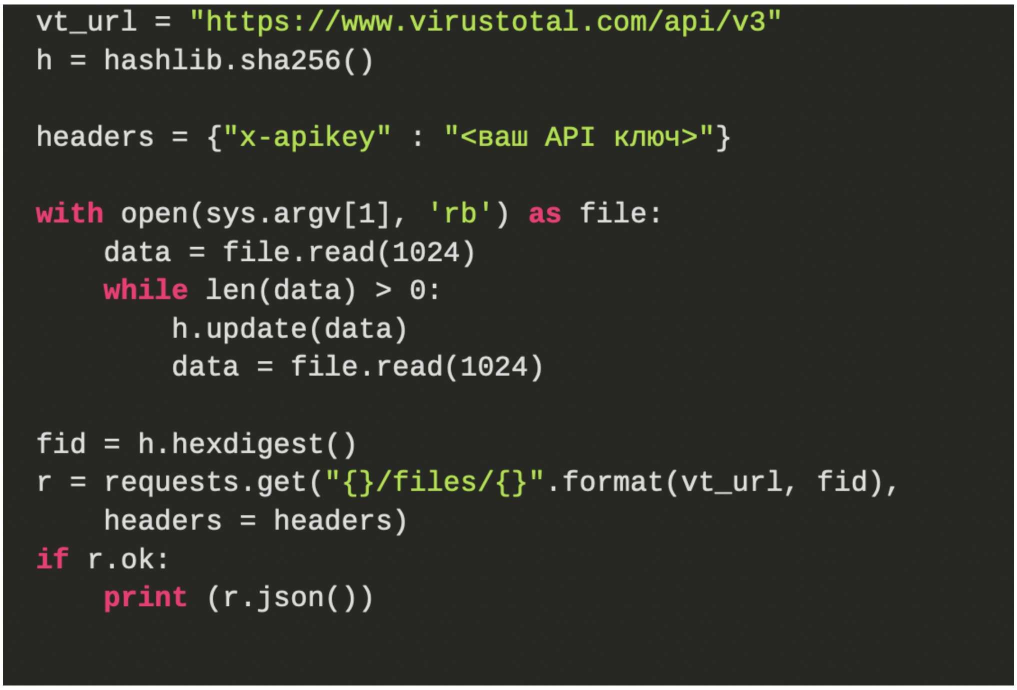
Matches rule Failed Code Integrity Checks by Thomas Patzke at Sigma Integrated Rule Set (GitHub)
↳ Code integrity failures may indicate tampered executables.

Matches rule Scheduled Task Creation by Florian Roth at Sigma Integrated Rule Set (GitHub)
↳ Detects the creation of scheduled tasks in user session

Matches rule Windows Cmd Delete File by frack113 at Sigma Integrated Rule Set (GitHub)
↳ Adversaries may delete files left behind by the actions of their intrusion activity. Malware, tools, or other non-native files dropped or created on a system by an adversary may leave traces to indicate to what was done within a network and how. Removal of these files can occur during an intrusion, or as part of a post-intrusion process to minimize the adversary's footprint.

Figure 3. Detection for patterns of a malicious file

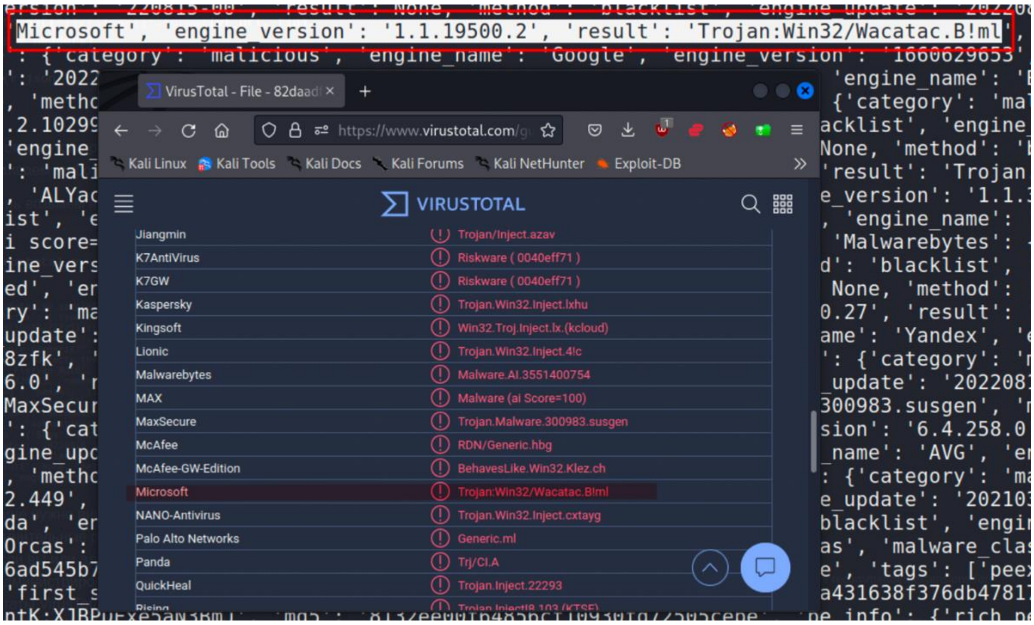
VirusTotal also provides a free API to automate static analysis of malicious programs in more agile way. After registering an account on the web page, user can get an API key to work further. To authenticate using the API, user must include his private API key in the “x-api key” header of all requests, and later start making requests to the public API. In the following examples we will use fragments of python3 code. To get information about a file by its identifier (SHA256, SHA1 or MD5 hash of the file), it is possible write a simple program in python3 with “sys, requests, and hashlib” libraries (Figure 4). The logic of given program is very simple: we calculate the SHA256 hash of the file (our malware sample) and request information from VirusTotal using the API. Next, we need to choose file samples and run the code.



```python
vt_url = "https://www.virustotal.com/api/v3"
h = hashlib.sha256()

headers = {"x-apikey" : "<ваш API ключ>"}

with open(sys.argv[1], 'rb') as file:
    data = file.read(1024)
    while len(data) > 0:
        h.update(data)
        data = file.read(1024)

fid = h.hexdigest()
r = requests.get("{}/files/{}".format(vt_url, fid),
    headers = headers)
if r.ok:
    print (r.json())
```

Figure 4. An automated identifier for files

After execution of the command “python3 virustotalcode.py samplefile.exe” (Figure 5), the exact number of rules and signatures of the VirusTotal is remotely triggered to investigate the uploaded samplefile.exe.



```
Microsoft', 'engine_version': '1.1.19500.2', 'result': 'Trojan:Win32/Wacatac.B!ml',
: {'category': 'malicious', 'engine_name': 'Google', 'engine_version': '1660629653'
```

Jiangmin	Trojan/Inject.azav
K7AntiVirus	Riskware (0040eff71)
K7GW	Riskware (0040eff71)
Kaspersky	Trojan.Win32.Inject.lxhu
Kingsoft	Win32.Troj.Inject.lx.(kcloud)
Lionic	Trojan.Win32.Inject.4!c
Malwarebytes	Malware.AI.3551400754
MAX	Malware (ai Score=100)
MaxSecure	Trojan.Malware.300983.susgen
McAfee	RDN/Generic.hbg
McAfee-GW-Edition	BehavesLike.Win32.Klez.ch
Microsoft	Trojan:Win32/Wacatac.B!ml
NANO-Antivirus	Trojan.Win32.Inject.cxtayg
Palo Alto Networks	Generic.ml
Panda	Trj/CI.A
QuickHeal	Trojan.Inject.22293
Rising	Trojan.Inject!8.103 (KTSE)

Figure 5. The result of API request

This program in python can be modified and improved, for example, by adding the calculation of other hashes of the test sample: MD5 and SHA-1 calculations. We believe that our program, written for educational and research purposes, will help novice Malware Analyst specialists and will also serve as a good foundation for more advanced and complex programs to automate routine processes of static malware sample research in the future.

References

1. https://www.virustotal.com
2. https://www.python.org/downloads/windows/
3. https://samples.vxunderground.org/APTs/2022/2022.09.07(2)/Samples/f6827dc5af661fb b4bf64bc625c78283ef836c6985bb2 bfb836bd0c8d5397332.7z

ӘОЖ 004.056.55

АҚПАРАТТЫҚ ҚАУІПСІЗДІК САЛАСЫНДАҒЫ ИНЦИДЕНТТЕРДІ ТЕРГЕУ ҚҰРАЛДАРЫН САЛЫСТЫРМАЛЫ ТАЛДАУ

Мырзақұл Жансая Нұрматуллақызы
Zhansaya.myrzakul@mail.ru
Л.Н.Гумилев атындағы Еуразия ұлттық университеті, ақпараттық технологиялар факультеті, ақпараттық қауіпсіздік кафедрасының студенті
Астана, Қазақстан
Ғылыми жетекші – Ахметова Ж.Ж., PhD, доцент

Қазіргі әлем цифрлық технологияларға көбірек көшуде, бұл ақпараттық қауіпсіздік инциденттерінің көбеюіне әкеледі. Әртүрлі ұйымдар, соның ішінде мемлекеттік органдар, компаниялар және жеке тұлғалар киберқауіпсіздік қаупіне және оқиғаларды тергеу үшін қажетті әрекеттерге тап болады.

Ақпараттық қауіпсіздік оқиғаларын тергеу үшін осы саланың мамандары қолдана алатын көптеген құралдар бар. Бұл құралдарға қауіпті анықтау және алдын алу бағдарламалық құралы, желілік трафикті талдау құралдары және ақпараттық жүйелердің қауіпсіздігін физикалық тексеруге арналған арнайы жабдық кіреді.

Алайда, ақпараттық қауіпсіздік саласындағы оқиғаларды тергеудің барлық құралдары бірдей емес. Олардың артықшылықтары мен кемшіліктері бар және белгілі бір құралды таңдау