

Charakteristiky moderního malwaru

Bc. Václav Hess

Diplomová práce
2022



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav informatiky a umělé inteligence

Akademický rok: 2021/2022

ZADÁNÍ DIPLOMOVÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení: Bc. Václav Hess
Osobní číslo: A20613
Studijní program: N0613A140022 Informační technologie
Specializace: Kybernetická bezpečnost
Forma studia: Kombinovaná
Téma práce: Charakteristiky moderního malwaru
Téma práce anglicky: Features of Modern Malware

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Provedte analýzu současných softwarových nástrojů, kterým lze využít na analýzu malwaru. Zhodnoťte jejich silné a slabé stránky.
3. Provedte analýzu vybraných malwarů a částí malwarů s cílem zjistit významné charakteristiky, které by mohly sloužit k jejich detekci.
4. Výstupem práce bude zdokumentovaná množina charakteristik současného malwaru, které mají detekční potenciál, proveďte odborné vyhodnocení.

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. KLEYMENOV, Alexey a Amr THABET. Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks. Birmingham: Packt Publishing, 2019. ISBN 1789610788.
2. MONNAPPA, K A. Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware. Birmingham: Packt Publishing, 2018. ISBN 9781788392501.
3. KIM, Peter. The Hacker Playbook 3: Practical Guide To Penetration Testing. USA: Secure Planet, 2018. ISBN 978-1980901754.
4. MALIN, Cameron, Eoghan CASEY a James AQUILINA. Malware Forensics: Investigating and Analyzing Malicious Code [online]. United States of America: Elsevier, Inc., 30 Corporate Drive, Burlington, MA 01803, 2008 [cit. 2021-12-06]. ISBN 9780080560199. Dostupné z: <https://www.elsevier.com/books/malware-forensics/malin/978-1-59749-268-3>
5. MALIN, Cameron, Eoghan CASEY a James AQUILINA. Malware Forensics Field Guide for Windows Systems: Digital Forensics Field Guides [online]. United States of America: Elsevier, Inc., 30 Corporate Drive, Burlington, MA 01803, 2012 [cit. 2021-12-06]. ISBN 9781597494731.

Vedoucí diplomové práce:

Ing. Milan Oulehla, Ph.D.
Ústav informatiky a umělé inteligence

Datum zadání diplomové práce: **3. prosince 2021**

Termín odevzdání diplomové práce: **23. května 2022**

doc. Mgr. Milan Adámek, Ph.D. v.r.
děkan



prof. Mgr. Roman Jašek, Ph.D., DBA v.r.
ředitel ústavu

Ve Zlíně dne 24. ledna 2022

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně;
- byl/a jsem seznámen/a s tím, že na moji diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování diplomové práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně, dne

Václav Hess v.r.
podpis studenta

ABSTRAKT

Tato diplomová práce řeší charakteristiky moderního malwaru. V teoretické části byla vypracována literární rešerše na dané téma. Dále byla provedena analýza softwarových nástrojů vhodných pro analýzy malwaru, kdy byly zhodnoceny jejich slabé a silné stránky. V praktické části byla provedena statická a dynamická analýza dvou rodin malwaru, pro zjištění jejich významných charakteristik. Výstupem práce bylo zadokumentování množin charakteristik současného malwaru, které mají detekční potenciál.

Klíčová slova: malware, charakteristiky, statická analýza, dynamická analýza, obfuskace, YARA

ABSTRACT

This thesis addresses the characteristics of modern malware. In the theoretical part, a literature research on the topic was conducted. Furthermore, an analysis of software tools suitable for malware analysis was performed, where their weaknesses and strengths were evaluated. In the practical part, static and dynamic analysis of two malware families was performed to identify their significant characteristics. The output of the work was the documentation of sets of characteristics of current malware that have detection potential.

Keywords: malware, characteristics, static analysis, dynamic analysis, obfuscate, YARA

Děkuji Ing. Milanu Oulehlovi, Ph.D. za pomoc při konzultacích a při vedení této diplomové práce.

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD	10
I TEORETICKÁ ČÁST	12
1 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY	13
1.1 KNIHA MASTERING MALWARE ANALYSIS: THE COMPLETE MALWARE ANALYST'S GUIDE TO COMBATING MALICIOUS SOFTWARE, APT, CYBERCRIME, AND IOT ATTACKS	13
1.2 KNIHA LEARNING MALWARE ANALYSIS: EXPLORE THE CONCEPTS, TOOLS, AND TECHNIQUES TO ANALYZE AND INVESTIGATE WINDOWS MALWARE	14
1.3 KNIHA THE HACKER PLAYBOOK: PRACTICAL GUIDE TO PENETRATION TESTING.	15
1.4 KNIHA MALWARE FORENSICS: INVESTIGATING AND ANALYZING MALICIOUS CODE.....	16
1.5 KNIHA MALWARE FORENSICS FIELD GUIDE FOR WINDOWS SYSTEMS: DIGITAL FORENSICS FIELD GUIDES	17
2 MALWARE	18
2.1 DRUHY MALWARU.....	19
2.2 ANALÝZA MALWARU	20
3 TYPY ANALÝZ MALWARU	21
3.1 STATICKÁ ANALÝZA VZORKU MALWARU	21
3.1.1 Antivirové systémy	21
3.1.2 Identifikace typu souboru.....	22
3.1.3 Vyhledání vložených řetězců	24
3.1.4 Vyhledání použitých funkcí a knihoven	25
3.1.4.1 Struktura PE (Portable Execution).....	25
3.2 DYNAMICKÁ (BEHAVIORÁLNÍ) ANALÝZA	26
3.2.1 Síťové služby.....	26
3.3 ANALÝZA KÓDU	27
3.4 ANALÝZA PAMĚTI RAM	27
4 MASKOVACÍ TECHNIKY MALWARU	29
4.1 ZABALENÍ OBSAHU SOUBORU (PACKING)	29
4.2 OBFUSKACE	30
4.3 ŠIFROVÁNÍ A KÓDOVÁNÍ.....	30
5 SOFTWAREOVÉ NÁSTROJE	34
5.1 REMNUX.....	34
5.2 NÁSTROJE PRO STATICKOU ANALÝZU VZORKU MALWARU	34
5.2.1 Nástroj file.....	35
5.2.2 Nástroj TrID	35
5.2.3 Nástroj Yara-Rules.....	36
5.2.4 Nástroj Detect-It-Easy.....	37
5.2.5 Nástroj ExifTool.....	38
5.2.6 Nástroj signsrch.....	39
5.2.7 Nástroj ssdeep	40
5.2.8 Nástroj bulk_extractor.....	41

5.2.9	Nástroj manalyze.....	42
	Nástroj strings.....	43
5.2.10	Nástroj stringsifter.....	44
5.2.11	Nástroj floss	44
5.2.12	Nástroj PEframe	46
5.2.13	Nástroj PE Tree	47
5.2.14	Nástroj PEDump	48
5.2.15	Nástroj pecheck.....	49
5.2.16	Nástroj base64dump.....	50
5.2.17	Nástroj xorsearch	51
5.2.18	Nástroj capa.....	52
5.2.19	PEStudio.....	52
5.2.20	YARA	53
5.3	HEXADECIMÁLNÍ EDITORY	55
5.3.1	Hexadecimální editor xxd	55
5.3.2	Hexadecimální editor wxHexEditor.....	56
5.4	NÁSTROJE PRO DYNAMICKOU ANALÝZU VZORKU MALWARU	57
5.4.1	Síťové nástroje	57
5.4.1.1	Nástroj tcpdump.....	58
5.4.1.2	Nástroj Wireshark.....	59
5.4.1.3	Nástroj INetSim	60
5.4.1.4	Nástroj accept-all-ips	62
5.4.2	Nástroje pro monitoring spuštěného malwaru	62
5.4.2.1	Nástroj Process Monitor	62
5.4.2.2	Nástroj Process Hacker.....	63
5.4.2.3	Nástroj RegShot.....	64
5.4.2.4	Nástroj ProcDot	65
5.5	NÁSTROJ PRO ZACHYCENÍ VÝPISU OBSAHU PAMĚTI FTK IMAGER.....	65
5.6	NÁSTROJ PRO ANALÝZU VÝPISU PAMĚTI RAM VOLATILITY3	66
5.7	DISASSEMBLERY/DEBUGGERY	67
5.7.1	Nástroj GDB.....	68
5.7.2	Nástroj cutter	69
5.7.3	Nástroj Ghidra.....	70
5.7.4	Nástroj IDA	71
5.7.5	Nástroj ILSpy	72
5.8	ZHODNOCENÍ NÁSTROJŮ PRO ANALÝZU MALWARU	72
II PRAKTICKÁ ČÁST		74
6	ANALÝZA PRVNÍ RODINY MALWARU	75
6.1	STATICKÁ ANALÝZA PRVNÍ RODINY MALWARU.....	75
6.1.1	Informace o souborech první rodiny malwaru	76
6.1.2	Analýza vložených řetězců.....	78
6.1.3	Analýza hlaviček vzorků malwaru.....	84
6.1.4	Antivirová kontrola ClamAV.....	86
6.2	DISASSEMBLING/DEBUGGING	87
6.3	DYNAMICKÁ ANALÝZA PRVNÍ RODINY MALWARU.....	88
6.3.1	Vzorek malwaru sample01.exe	89
6.3.2	Vzorky malwaru sample02.exe až sample08.exe	92

6.3.3	Detekce první rodiny malwaru.....	93
7	ANALÝZA DRUHÉ RODINY MALWARU.....	96
7.1	STATICKÁ ANALÝZA DRUHÉ RODINY MALWARU	96
7.1.1	Informace o souborech druhé rodiny malwaru	96
7.1.2	Analýza vložených řetězců.....	99
7.1.3	Analýza hlaviček vzorků druhé rodiny malwaru	101
7.1.4	Antivirová kontrola ClamAV.....	104
7.2	DISASSEMBLING/DEBUGGING	104
7.3	DYNAMICKÁ ANALÝZA DRUHÉ RODINY MALWARU	104
7.3.1.1	Vzorek malwaru 2sample01.exe.....	105
7.3.1.2	Vzorky malwaru 2sample02.exe až 2sample05.exe.....	110
7.3.1.3	Detekce druhé rodiny malwaru.....	110
8	MNOŽINA CHARAKTERISTIK SOUČASNÉHO MALWARU, KTERÉ MAJÍ DETEKČNÍ POTENCIÁL.....	114
	ZÁVĚR	116
	SEZNAM POUŽITÉ LITERATURY.....	119
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	125
	SEZNAM OBRÁZKŮ	127
	SEZNAM TABULEK.....	130
	SEZNAM PŘÍLOH.....	131

ÚVOD

Bez počítačů si v dnešní době nedovedeme představit náš současný život. Různé druhy procesorů jsou všude kolem nás. Ať už je to klasický stolní počítač nebo notebook, ale i mobilní telefon, televize, automobil, vše má nějaký druh procesoru. Vše je závislé na výpočetní technice. Bohužel ne všichni vývojáři vytvářejí software s dobrými úmysly. Díky tomu je velké množství škodlivého softwaru, který se obecně označuje jako malware. Nejčastěji se proti malwaru bráníme pomocí různých druhů antivirů, které ovšem nemusejí být dostupné na všech zařízeních. Obecně se malware vyvíjí stejně jako jakýkoliv jiný software a díky tomu jsou novější verze více zákeřné a mohou lépe uniknout antivirové detekci. Jen pro představu, aktuální verze open source antivirového řešení ClamAV¹ zná ke dni 28.3.2022 celkem 8 608 974 signatur různých druhů malwaru viz.Obrázek 1. a pořád se jejich počet rozrůstá.

```
----- SCAN SUMMARY -----
Known viruses: 8608974
Engine version: 0.104.2
Scanned directories: 1
Scanned files: 0
Infected files: 0
Data scanned: 0.00 MB
Data read: 0.00 MB (ratio 0.00:1)
Time: 22.537 sec (0 m 22 s)
Start Date: 2022:03:28 18:38:49
End Date: 2022:03:28 18:39:11
```

Obrázek 1. Počet signatur antiviru ClamAV [zdroj vlastní]

V dokumentu NÚKIB „Zpráva o stavu kybernetické bezpečnosti České republiky za rok 2020“ jsou uvedeny tři nejvýznamnější hrozby roku 2020 a to jsou ransomware, DDoS útoky a spear-phishing.

„Organizace jako nejzávažnější hrozby roku 2020 hodnotily ransomware a DDoS útoky (Graf 17). Ransomware za nejzávažnější útok označila téměř třetina respondentů ze sektoru zdravotnictví a 25 % respondentů z

¹ Dostupný na <https://clamav.net>

finančního i veřejného sektoru. Třetí nejzávažnější hrozbu představoval podle respondentů spear-phishing. V minulosti bylo možné phishingové útoky odhalit poměrně jednoduše skrze špatnou češtinu a odlišné domény v e-mailech. V posledních letech se však potvrzuje trend jejich vyšší propracovanosti v používání lepších formátů e-mailů i obsáhlého portfolia motivů útočníků od žádosti o zaplacení faktury po obnovení přihlašovacích údajů k účtu. „ [1]

Pokud se podíváme do historie, tak první virus s názvem „BRAIN“ vznikl již v roce 1986 v Pákistánu. Byl to vir, který napadal bootovací sektor disket DOS FAT. Původně měl chránit software, před nelegálním šířením. [2]

A od té doby se vyvíjí malware, který nám všem znepříjemňuje život.

I. TEORETICKÁ ČÁST

1 SOUČASNÝ STAV ŘEŠENÉ PROBLEMATIKY

Všechny níže uvedené knihy patří k těm nejlepším, co lze nalézt na současném trhu a poskytují ucelený přehled o analýze vzorků malwaru, včetně statické a dynamické analýzy vzorků malwaru.

1.1 Kniha **Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks**

Knihu „Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks“ nejlépe vystihuje níže uvedený text.

„Kompletní průvodce analytika malwaru pro boj proti škodlivému softwaru, APT, kybernetické kriminalitě a útokům na IoT.“² [3]

V úvodu knihy jsou obecně vysvětleny pojmy jako procesor, registry, paměť, virtuální paměť, zásobník další informace, které jsou podstatné pro pochopení činnosti malwaru.

„Processor je srdce každého zařízení nebo počítače a vdechuje mu život.“³ [3]

V další části knihy je bližší seznámení s různými architekturami procesorů jako jsou x86 (včetně IA-32 a x64), ARM, MIPS, PowerPC, SuperH a SPARC. Jsou zde popsány podrobně typy registrů a instrukční sady. Jedná se o ucelený přehled architektur procesorů, které se v dnešní době používají v nejrůznějších zařízeních. [3]

Další část knihy je věnována statické a dynamické analýze pro architekturu x86/x64 na platformě MS Windows, tj. spustitelné soubory PE (Portable Execution). Do všech podrobností je v této knize popsána hlavička souboru, rozdíly mezi x86 a x64 a další. Dále je v této knize popsán způsob využití dynamických knihoven DLL a jakým způsobem je malware dokáže zneužívat. Jako příklad mohou posloužit techniky injektáže DLL knihoven. Každá uvedená technika je následně rozebrána na několika názorných příkladech a je v knize uvedeno, jakým způsobem je možné provádět analýzu takovýchto souborů. [3]

² The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks

³ A processor is like a heart of any smart device or computer in that it keeps them alive.

V této knize je velká pozornost věnována komprimaci, dešifrování a obfuskaci souborů, která zajišťuje ochranu malwaru před analýzou. Jsou zde uvedeny i techniky, jak tyto ochrany obejít. [3]

Další část knihy je věnována rootkitům, exploitům jako např. přetečením zásobníku (stack overflow) a to nejen na platformě MS Windows, ale i např. na platformě GNU Linux. Dále je věnována pozornost i malwaru, který se šíří v souborech MS Office a PDF. [3]

Tato kniha se také zaměřuje i IoT platformu, kde popisuje strukturu spustitelných souborů ELF a metody analýzy podezřelých souborů. [3]

Ve svém závěru se kniha zabývá malwarem pro macOS, iOS, Android a možnostmi analýzy tohoto malwaru. [3]

Tato kniha je velmi dobrá pro pokročilejší analytiku malwaru a je pro ně velkým přínosem. Její největší předností je, že přináší ucelený přehled různých chování malwaru pro různé platformy a možnosti analýzy těchto malwarů.

1.2 Kniha Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware

Kniha Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware se zaměřuje na analýzu malwaru pro MS Windows.

„Prozkoumejte koncepty, nástroje a techniky pro analýzu a zkoumání malwaru Windows.“⁴[4]

V úvodu tato kniha přibližuje čtenáři základní informace jako co je to malware, jaké se používají analýzy malwaru a také se věnuje nastavení laboratorního prostředí pro vyšetřování chování malwaru při dynamické analýze. [4]

Podrobně se autor věnuje statické analýze a jaké informace lze touto analýzou získat, včetně klasifikace rodiny malwaru pomocí různých typů statické analýzy. V další části se autor věnuje dynamické analýze, kdy krok za krokem provede čtenáře všemi aspekty dynamické analýzy. [4]

⁴ Explore the concepts, tools, and techniques to analyze and investigate Windows malware.

Dále kniha ukazuje příklady disassembleru a jejich využití při analýze malwaru. Vše je doplněno ukázkami disassemblingu a debugingu malwaru pomocí nástroje IDA Pro. [4]

V další kapitole jsou popsány základní funkce malwaru a metody perzistence v operačním systému. Také se tato kniha zaměřuje na injektáž kódu a techniky zavěšení (hooking). Dále jsou v této knize dostupné informace o technikách obfuskace, komprimace a šifrování malwaru. Kniha je také unikátní v popisu metod vyšetřování zachycené (dump) paměti RAM za použití nástroje Volatility Framework. [4]

Tato kniha autora diplomové práce oslovila nejvíce ze všech knih. Nabízí ucelený přehled jak pro začátečníky, tak i pro profesionály při analýze malwaru. Vše je srozumitelně popsáno a vysvětleno na příkladech z praxe.

1.3 Kniha The Hacker Playbook: Practical Guide To Penetration Testing.

Kniha The Hacker Playbook: Practical Guide To Penetration Testing se zaměřuje na seznámení čtenářů s penetračním testováním. I když se toto téma může zdát, že nemá s malwarem spojitost, opak je pravdou. Je důležité znát možnosti infiltrace hackerů do počítačového systému a nejlépe se tato problematika pochopí právě na penetračním testování. Na začátku knihy jsou obecné informace, co to je penetrační testování, kdy a jak se provádí a další informace. Dále autor knihy seznamuje čtenáře s nástroji jako je Metasploit Framework, Cobalt Strike, PowerShell Empire, PoshC2, Merlin a dalšími, které se používají na penetrační testování. Na začátku všech penetračních testů je potřeba získat co nejvíce informací o cílovém systému. To se provádí pomocí mnoha technik skenování, které jsou v této knize velmi podrobně popsány. [5]

Dále autor knihy popisuje webové zranitelnosti jako je Cross Site Scripting (XSS), SQL injektáže, a další. Velice zajímavá je i kapitola o síťové bezpečnosti a o tom, jak penetrační tester postupuje při úspěšném proniknutí do lokální sítě, včetně získání přístupových oprávnění k serverům. Kniha popisuje postupy nejen pro síť založenou na platformě MS Windows, ale i na platformě GNU Linux. [5]

Kniha se také věnuje možnostem penetračního testování za pomoci phishingu a tím získání přihlašovacích údajů do informačních systémů. [5]

Na konci knihy se lze dočíst o možnostech crackingu hesel, exploitace systémů a různé triky na usnadnění práce. [5]

Tato kniha je dobrým zdrojem informací k problematice penetračních testů.

1.4 Kniha Malware Forensics: Investigating and Analyzing Malicious Code

„Tato kniha byla navržena tak, aby pomohla analytikům identifikovat malware v počítačovém systému a odhalit jeho funkčnost, účel a vyhodnotit škody, které malware způsobil.“⁵[6]

Kniha Malware Forensics: Investigating and Analyzing Malicious Code poskytuje celkovou metodologii, pro řešení malwarové infekce. Je rozdělena do několika částí: analýza dočasných dat, analýza paměti, forenzní analýza pevných disků, statická analýza a dynamická analýza malwaru. [6]

V této knize je také velmi dobře popsána analýza smazaných souborů, která může objasnit jinak nedostupné střípky informací o malwaru. Mohou napovědět například o infiltraci informačního systému. Dále jsou zde informace o nástrojích, které lze využít pro účely analýzy. Jelikož je kniha staršího data, neodpovídají nástroje dnešním potřebám a je proto nutné vyhledat jejich alternativy, avšak základní principy a metody analýzy malwaru zůstávají stále aktuální. [6]

Na konci této knihy je unikátní modelový příklad analýzy malwaru, včetně statické i dynamické analýzy a disassemblování binárního souboru. [6]

⁵ This book is designed to help digital investigators identify malware on a computer system, pull malware apart to uncover its functionality and purpose, and determine the havoc malware wreaked on a subject system.

1.5 Kniha Malware Forensics Field Guide for Windows Systems: Digital Forensics Field Guides

Kniha Malware Forensics Field Guide for Windows Systems: Digital Forensics Field Guides je pokračování předchozí knihy Malware Forensics: Investigating and Analyzing Malicious Code. Uvedený literární pramen je jedinečný tím, že obsahuje zpracované formuláře pro forenzní analýzu vzorků malwaru. Součástí formulářů jsou zaškrtačovací pole, které analytik vyplňuje a díky tomu pomáhá se všemi aspekty a součásti analýzy. Tyto formuláře mohou sloužit pro společnou analýzu několika analytiků na jednom vzorku malwaru. Kniha také obsahuje mnoho pokročilých tipů, které mohou pomoci forezním analytikům.[7]

2 MALWARE

„Termín malware je kombinací dvou slov – malicious, což znamená škodlivý a software. Do kategorie malware řadíme veškerý škodlivý kód, přičemž nezáleží na způsobu, jakým napadá počítače, ani na chování nebo na výsledku jeho činnosti.“[8]

Malware je tedy jakýkoliv škodlivý software, který může být jako spustitelný soubor (například .exe, .dll, sys, .drv apod.) nebo je součástí jakéhokoliv jiného legitimního softwaru. Jeho činnost je převážně skrytá, aby malware nevzbudil nežádoucí pozornost, která by mohla vést ke zjištění infekce a k následnému odstranění. Obvykle bývá malware spuštěn bez vědomí uživatele anebo s jeho pomocí, ale vždy skrytě. Malware je vytvářen vždy s úmyslem nějakého zisku. Ten může být duševního anebo finančního charakteru. Může se zaměřovat na získávání citlivých informací, tj. osobních dat, obchodních dat, finančních údajů nebo na špehování oběti. Další možností je zapojení do distribuovaných útoků DDoS, odesílání spamu, těžbu kryptoměn, uzamčení souborů v počítači nebo narušení jeho provozu a následného vydírání. Jak již bylo uvedeno v úvodu této práce, ve zprávě NÚKIB za rok 2020 byl zmíněn nárůst počtu útoků ransomware. Dne 25.2.2022 vydal NÚKIB varování na výskyt nového destruktivního malwaru HermeticWiper, který úmyslně nenávratně poškozuje zájmové soubory tak, aby nebyly čitelné, bez možnosti obnovy. [9]

Do počítačového systému se může malware dostat různými způsoby jako jsou například infikované přenosné USB zařízení (flash disky), dále přes web nebo přes emailovou přílohu. V dřívější době bylo běžné infikování operačního systému pomocí softwaru, který měl odstraněnou vestavěnou ochranu softwaru, proti nelegálnímu kopírování (cracknutá verze). Tento nelegální software měl většinou přídavek v podobě skrytého malwaru.

Další způsob napadení počítačového systému je s využitím zranitelnosti neaktualizovaného softwaru. Pro popis těchto chyb byl vytvořený systém Common Vulnerabilities and Exposures zkratka CVE. Tento systém poskytuje referenční metody pro veřejně známé zranitelnosti. CVE je provozovaná neziskovou organizací The Mitre Corporation.[10]

„Posláním programu CVE je identifikovat, definovat a katalogizovat zveřejněné zranitelnosti kybernetické bezpečnosti“[11]

Zranitelnost je chyba v počítačovém systému, která umožňuje uplatnění kybernetické hrozby. [12]

Malware není jen záležitostí počítačových systémů, ale může se vyskytnout i v zařízeních jako jsou IoT, nebo scada. IoT je zkratka používaná pro „Internet of Things“, tedy v překladu „internet věcí“.

„Síť fyzických zařízení, vozidel, domácích spotřebičů a dalších zařízení, která jsou vybavena elektronikou, softwarem, senzory/čidly a hlavně síťovou konektivitou. Ta umožňuje těmto zařízením se navzájem propojit a vyměňovat si data.“ [13]

2.1 Druhy malwaru

Malware je široký pojem, který označuje různé typy škodlivých programů jako jsou trojské koně, viry, červy, rootkity a podobně. Při analýze malwaru lze narazit na jeho různé typy. Některé z nich jsou kategorizovány na základě jejich funkčnosti a vektoru útoku. [4]

- **Virus nebo červ (worms):** je malware, který se dokáže kopírovat a šířit na jiné počítače. Virus vyžaduje určitou činnost uživatele, zatímco červ se může šířit bez činnosti uživatele. [4]
- **Trojan:** malware, který se maskuje jako běžný užitečný program. Po instalaci může provádět krádeže citlivých dat, nahrávání souborů na server útočníka nebo například sledování webových kamer. [4]
- **Backdoor/Remote Access Trojan (RAT):** jedná se o typ trojského koně, který útočníkovi umožňuje získat přístup k napadenému systému a vzdáleně ho ovládat. [4]
- **Adware:** malware, který uživateli vkládá nevyžádanou reklamu. Obvykle je přidáván k aplikacím, které lze bezplatně stáhnout. [4]
- **Botnet:** jedná se o skupinu počítačů infikovaných stejným malwarem (nazývaným boti), kteří čekají na obdržení pokynů z C&C serveru ovládaného útočníkem. Útočník pak může vydat příkaz těmto napadeným zařízením, která mohou provádět škodlivé aktivity jako jsou útoky DDoS nebo rozesílání spam e-mailů. [4]
- **Information stealer:** malware navržený ke krádeži citlivých dat jako jsou bankovní přihlašovací údaje nebo keyloggery, které zachytávají stisky kláves. Některé vzorky těchto malwarů v sobě zahrnují keyloggery, spyware, sniffery a nástroje pro zachycení formulářů. [4]

- **Ransomware:** malware, který zablokuje přístup uživatelům k jejich počítači nebo zašifruje jejich soubory a následně žádá výkupné. [4]
- **Rootkit:** malware, který poskytuje útočnickovi privilegovaný přístup k infikovanému systému a skrývá jeho přítomnost nebo přítomnost jiného softwaru. [4]
- **Downloader nebo dropper:** malware určený ke stažení nebo instalaci dalších součástí malwaru. [4]

Mezi nejpoužívanější operační systém na platformě PC je MS Windows. Proto se na něj nejčastěji zaměřují autoři malwaru. Jde to i v ruku v ruce s tím, že MS Windows používají i lidé, kteří nejsou v práci s operačním systémem nijak zblhlí a jsou to lidé v oblasti bezpečnosti nevzdělaní. Tím mají útočníci nejjednodušší cestu k infiltraci malwaru do operačního systému. To ovšem neznamená, že pro ostatní platformy jako GNU Linux, macOS a další malware neexistuje. Dále nesmíme zapomenout na různá zařízení IoT, na která se v poslední době útočníci více zaměřují. Přesto je platforma MS Windows nejčastějším cílem útoku.

2.2 Analýza malwaru

Analýza malwaru je studiem chování malwaru. Cílem analýzy je porozumět fungování malwaru a jak jej detekovat a eliminovat. Zahrnuje analýzu podezřelého souboru v bezpečném prostředí za účelem identifikace jeho charakteristik a funkcí. [4]

Analýza je důležitá k určení úmyslu a motivu útočníka.

3 TYPY ANALÝZ MALWARU

Pro pochopení činnosti a funkce malwaru a posouzení dopadu na počítačový systém se začaly používat různé analytické techniky. Níže je uvedena klasifikace těchto analytických technik. [4]

3.1 Statická analýza vzorku malwaru

Statická analýza je proces analýzy binárního souboru bez jeho spuštění. Jedná se o nejjednodušší techniku analýzy. Umožňuje extrahovat metadata z podezřelého souboru. Tato analýza nemusí odhalit všechny funkce a činnost malwaru, ale může poskytnout podstatné informace, které mohou pomoci s dalšími typy analýz.[14]

Jedná o bezpečnou analýzu bez většího rizika nákazy operačního systému. Přesto je dobré pracovat v izolovaném prostředí jako je například virtuální počítač. Jako další ochranu před nechtěným spuštěním vzorku souboru lze využít jiný druh operačního systému než ten, pro který je soubor určený.

3.1.1 Antivirové systémy

Nejjednodušším způsobem identifikace, zda vzorek souboru obsahuje malware, je pomocí několika různých antivirových systémů, které ale nejsou spuštěny současně.

„Používání několika antivirových programů v jednom počítači důrazně nedoporučujeme. Z technického hlediska se dva a více antivirů v počítači bude "přetahovat" o soubory, které kontrolují. Zjednodušeně řečeno je to "jako by se dvě uklízečky praly o jedno koště". Běžné antivirové aplikace totiž kontrolují soubory v reálném čase. To znamená, že např. při otevření webového prohlížeče je antivirem zkontrolován soubor iexplore.exe, který prohlížeči patří. Pokud bude v počítači několik antivirových programů, budou chtít soubor iexplore.exe v jeden okamžik zkontrolovat všichni najednou. V konečném důsledku tak soubor zkontrolován nebude, jelikož může být v jeden okamžik poskytnut pouze jedné aplikaci.“ [15]

Jen jeden antivirový systém většinou nepostačuje, protože každý výrobce antivirového řešení využívá jiné metody identifikace malwaru, jiné signatury apod. Žádný antivir není

dokonalý. Díky tomu může dojít k detekční chybě označované jako false positive nebo false negative.

- False positive je chyba, kdy antivirový systém označí vzorek souboru jako malware, ale ve skutečnosti se jedná o neškodný soubor.
- False negative je chyba, kdy antivirový systém neoznačí vzorek souboru jako malware, i když jim ve skutečnosti je.

Pro antivirovou kontrolu lze využít i lokálně nainstalovaného antiviru do operačního systému.

Také je možné využít online služeb jako je například VirusTotal⁶ nebo VirScan⁷, které umožňují analyzovat daný soubor desítkami různých antivirových systémů.

Tím, že nahrajeme takový soubor k analýze, musíme vzít v úvahu i fakt, že v licenčním ujednání online služeb souhlasíme se zpřístupněním takového souboru bezpečnostní komunitě, která je do projektů zapojená. [16]

To nemusí být vždy žádoucí. Pokud takovýto podezřelý soubor nechceme nahrávat do online služeb, můžeme využít hash vzorku malwaru k jeho identifikaci ve službě VirusTotal⁶. Můžeme využít hash MD5, SHA1 a SHA256. Musíme mít však na paměti, že vyhledáním hodnoty hash v online službách, hledáme pouze výsledky již provedených antivirových skenů. Z toho vyplývá, že pokud se jedná o hodnotu hash souboru, který ještě nebyl analyzován pomocí online služeb, nedostaneme odpověď na základní otázku, zda se jedná o malware či nikoli.

3.1.2 Identifikace typu souboru

Prvním krokem statické analýzy by měla být identifikace typu souboru. Pro účely analýzy je podstatné vědět, zda analyzovaný vzorek je spustitelný soubor a pro jaký operační systém. Identifikace typu souboru se nesmí omezit na kontrolu přípony zkoumaného vzorku souboru, protože přípona je lehce modifikovatelná.

Identifikovat typ souboru lze například pomocí magického čísla (magic number), které se nachází na začátku souboru a lze jej přečíst hexadecimálním prohlížečem. V operačním

⁶ Dostupný na <https://www.virustotal.com>

⁷ Dostupný na <https://www.virscan.org>

systemu GNU Linux je dostupný například hexadecimální editor xxd. Jeho prostředí je vidět na Obrázek 2. Záhloví PE souboru v hexadecimálním editoru xxd [zdroj vlastní] Práce přímo v hexadecimálním prohlížeči je časově náročná a vyžaduje rozsáhlé znalosti analytika.

Magické číslo spustitelných souborů MS Windows je 0x5A, 0x5A hex, tj. v ASCII MZ. Linuxové spustitelné soubory mají magic number ELF, tj. 7F 45 4C 46.⁸

Pro usnadnění práce lze využít několika různých nástrojů, které budou uvedeny v kapitole Softwarové nástroje.

```
remnux@remnux:~$ xxd -g 1 /media/sf_MALWARE/7z2107-x64.exe | head
00000000: 4d 5a 90 00 03 00 00 00 04 00 00 00 ff ff 00 00  MZ.....
00000010: b8 00 00 00 00 00 00 00 40 00 00 00 00 00 00  ..>.....07.....
00000020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  @.....
00000030: 00 00 00 00 40 00 38 00 0d 00 40 00 1e 00 1d 00  ....@.8...@.....
00000040: 06 00 00 00 04 00 00 00 40 00 00 00 00 00 00 00  .....@.....
00000050: 40 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  @.....@.....
00000060: d8 02 00 00 00 00 00 00 d8 02 00 00 00 00 00 00  .....
00000070: 08 00 00 00 00 00 00 00 03 00 00 00 04 00 00 00  .....
00000080: 18 03 00 00 00 00 00 00 18 03 00 00 00 00 00 00  .....
00000090: 18 03 00 00 00 00 00 00 1c 00 00 00 00 00 00 00  .....
remnux@remnux:~$
```

Obrázek 2. Záhloví PE souboru v hexadecimálním editoru xxd [zdroj vlastní]

Pro srovnání je na dalším obrázku zobrazeno záhloví spustitelného souboru pro Linux.

```
00000000: 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00  .ELF.....
00000010: 03 00 3e 00 01 00 00 00 30 37 00 00 00 00 00 00  ..>.....07.....
00000020: 40 00 00 00 00 00 00 00 f8 e1 00 00 00 00 00 00  @.....
00000030: 00 00 00 00 40 00 38 00 0d 00 40 00 1e 00 1d 00  ....@.8...@.....
00000040: 06 00 00 00 04 00 00 00 40 00 00 00 00 00 00 00  .....@.....
00000050: 40 00 00 00 00 00 00 00 40 00 00 00 00 00 00 00  @.....@.....
00000060: d8 02 00 00 00 00 00 00 d8 02 00 00 00 00 00 00  .....
00000070: 08 00 00 00 00 00 00 00 03 00 00 00 04 00 00 00  .....
00000080: 18 03 00 00 00 00 00 00 18 03 00 00 00 00 00 00  .....
00000090: 18 03 00 00 00 00 00 00 1c 00 00 00 00 00 00 00  .....
remnux@remnux:~$
```

Obrázek 3. Záhloví ELF souboru v hexadecimálním editoru xxd [zdroj vlastní]

⁸ Zdroj pro určení souboru pomocí magických čísel je k dispozici na <https://filesignatures.net/>

Pro usnadnění existují různé nástroje, pomocí kterých lze jednoduše identifikovat typ souboru. Mezi nejznámější nástroj pro identifikaci souboru patří linuxový program file.

3.1.3 Vyhledání vložených řetězců

Řetězec je posloupnost znaků, které lze nalézt v samotném souboru a může být ve formátu ASCII nebo Unicode (dva bajty na znak). Spustitelný soubor tyto řetězce obsahuje, pokud má zobrazit nějakou zprávu nebo se např. připojit k dané URL adrese a podobně.

Extrahování řetězců může poskytnout vodítko o funkčnosti programu a indikátorech spojených s podezřelým binárním souborem. Přestože řetězce neposkytují jasný obrázek o účelu a schopnostech souboru, mohou napovědět, co je malware schopen dělat. [4]

Aby tvůrci znesnadnili analýzu malwaru, využívají různé druhy antireverzních technik pro schování vložených řetězců. Tyto postupy budou uvedeny v kapitole 4 Maskovací techniky malwaru.

Například při vyhledání řetězců v malwaru RAT Sakula lze mimo jiné nalézt následující řetězce:

```
http://  
HTTP/1.1  
POST  
SOFTWARE\Microsoft\Windows\CurrentVersion\Run\  
@Microsoft Visual C++ Runtime Library  
Program Files (x86)  
cmd.exe /c
```

Z toho lze vyvodit, že malware RAT Sakula může komunikovat s C&C serverem přes HTTP/1.1 metodou POST, persistence pravděpodobně bude zajištěna vytvořením klíče v registru SOFTWARE\Microsoft\Windows\CurrentVersion\Run a tento malware byl napsán pomocí Microsoft Visual C++. Při analýze je nutné se zaměřit na adresář „Program Files (x86)“, kde bude pravděpodobně malware uložený. Ke své činnosti bude využívat příkazový interpret cmd.exe. Z výše uvedeného je zřejmé, že analýza textových řetězců může hodně napovědět o funkcích malwaru.

3.1.4 Vyhledání použitých funkcí a knihoven

Pro pokročilou analýzu malwaru je nutné pochopit strukturu spustitelných souborů. Je velký rozdíl mezi strukturou spustitelných souborů na platformě MS Windows anebo linuxových spustitelných souborů. Jelikož většina malwaru cílí na uživatele platformy MS Windows, je nutné znát strukturu spustitelných souborů PE.

3.1.4.1 *Struktura PE (Portable Execution)*

Spustitelné soubory Windows musí odpovídat formátu PE/COFF (Portable Executable/Common Object File Format). Tento formát používají nejen spustitelné soubory, ale i soubory dynamických knihoven (.dll), ovladače (.drv) a další. [3]

Odděluje spustitelný kód a data do sekcí. Každá sekce má své paměťová oprávnění. [3]

Na začátku .exe souboru je z důvodu kompatibility s DOS je hlavička DOS MZ (DOS MZ header). Pokud se spustí tento soubor pod DOSem, je vyhodnocena tato hlavička jako korektní a dojde ke spuštění části nazvané DOS stub. Tam je většinou uloženo hlášení „This program cannot be run in DOS mode.“ Pokud je spuštěn soubor na platformě Windows, dojde k vyhledání PE loader v DOS MZ hlavičce a tím přeskočení DOS stub. [3]

Záhlaví PE header popisuje strukturu souboru. Hlavička PE obsahuje informace jako je místo, kde je třeba spustitelný soubor načíst do paměti, adresa, kde spuštění začíná, seznam knihoven/funkcí, na které aplikace spoléhá a zdroje používané binárním souborem. Zkoumání hlavičky PE poskytuje množství informací o binárním systému a jeho funkcích. [17]

Když má spustitelný soubor vytvořit soubor na disku, využije pravděpodobně k tomu rozhraní API CreateFile(), které je součástí kernel32.dll, tzn. musí se nejprve načíst kernel32.dll do paměti a potom zavolat funkci CreateFile(). Proto je u analýzy malwaru důležité věnovat pozornost voláním API, které nám umožní vytvořit si představu o funkcích malwaru. Závislosti souborů ve spustitelných souborech Windows jsou uloženy v importní tabulce struktury souboru PE. [4]

Z hlavičky PE lze získat následující důležité informace [3]:

- zda byl malware zabalen, [3]
- zda se jedná o dropper nebo downloader funkce URLDownloadToFile, [3]
- připojuje se k C&C, [3]
- jaké funkce má tento malware, [3]

- jaká je země původu útočníků, [3]
- zda neobsahuje digitální podpis ukradeným certifikátem, [3]
- případné příbuzné vzorky. [3]

3.2 Dynamická (behaviorální) analýza

Jedná se o proces spuštění podezřelého binárního souboru v izolovaném a monitorovaném prostředí. Výstupem je informace o chování malwaru při jeho běhu. Tato technika také nemusí odhalit veškeré funkce a vlastnosti malwaru. [7]

Je nutné dbát velké obezřetnosti, aby nedošlo k opuštění chráněného prostředí malwarem a nechtěným infikováním zařízení.

Jde o analýzu spuštěného souboru a sledování jeho chování v čase. Proto se tato analýza provádí se spuštěnými monitorovacími nástroji, které sledují činnost operačního systému jako je čtení a zápis na disk, změny registru, síťová aktivita (včetně zachytávání paketů) a další, při spuštěném vzorku malwaru. Monitorováním operačního systému vzniká velké množství logů, který vzniká provozem samotného operačního systému, nebo jiných spuštěných programů. Je tedy úkolem analytika, aby z těchto logů dokázal vyfiltrovat záznamy spjaté se zkoumaným vzorkem malwaru.

3.2.1 Síťové služby

Při dynamické analýze je důležité monitorování síťové aktivity operačního systému, ve kterém je spuštěn vzorek malwaru. Zachycený síťový provoz pomáhá identifikovat síťové vlastnosti malwarového vzorku. Například může vzorek malwaru generovat provoz na webový server a čekat na zpětnou interakci, ve které může čekat řídicí příkazy. Nebo po navázání spojení se vzdáleným serverem útočníka, může začít odesílat citlivé údaje z napadeného systému. Pro tyto účely je dobré hned ze začátku dynamické analýzy zachytávat síťový provoz, který odchází z napadeného systému, a kontrolovat, zda neobsahuje DNS dotazy, HTTP provoz, SMB provoz a podobně. Proto je nutné nastavit forenzní prostředí tak, aby na tyto požadavky reagovalo a odpovídalo. Systémy MS Windows nejsou nativně vybaveny nástroji pro zachytávání síťového provozu. Pro zachytávání síťového provozu je dobré zvolit nástroje jako je Wireshark, tcpdump a podobně. [6]

Každý malware může využívat různé druhy protokolů a různě skrývat svůj síťový provoz. Proto je velmi těžké stanovit jednotný postup. Pro účely zachytávání síťového provozu a

následné zprovoznění služeb, které budou simulovat služby C&C serveru, je dobré využít linuxovou distribuci Remnux⁹, která obsahuje vše potřebné nejen pro monitoring sítě, ale obsahuje i mnoho nástrojů pro interakci s malwarem.

3.3 Analýza kódu

Tato pokročilá technika se zaměřuje na analýzu kódu, za účelem pochopení vnitřního fungování spustitelného souboru. Analýzu kódu lze rozdělit na statickou a dynamickou. Statická zahrnuje disasemblování spustitelného souboru pro pochopení funkce malwaru a dynamická analýza kódu zahrnuje debugging podezřelého spustitelného souboru. [4]

3.4 Analýza paměti RAM

Jedná se o forenzní techniku analýzy paměti RAM, která se zaměřuje na nalezení artefaktů malwaru v paměti.

„Obvykle se jedná o forenzní techniku, ale její integrace do analýzy malwaru Vám pomůže porozumět chování malwaru po infekci systému. Analýza paměti je zvláště užitečná pro určení utajovaných a únikových schopností malwaru¹⁰.“ [4]

Pro účely analýzy vzorku malwaru je dobré po spuštění malwaru vytvořit výpis celé paměti (tzv. dump), který je uložený do souboru. Tím lze získat informace o chování malwaru, které bychom jinak těžko získávali. Autoři malwaru využívají různé antireverzní techniky jako například komprese obsahu souboru.

Výpis obsahu paměti RAM lze vytvořit několika způsoby, buď specializovaným softwarem, jako je např. FTK Imager¹¹, nebo například vyvolat chybu operačního systému MS Windows, kdy sám operační systém uloží výpis obsahu celé paměti do souboru.

⁹ Dostupná z webu <https://remnux.org>

¹⁰ It is typically a forensic technique, but integrating it into your malware analysis will assist in gaining an understanding of the malware's behavior after infection. Memory analysis is especially useful to determine the stealth and evasive capabilities of the malware.

¹¹ Dostupný z webu <https://accessdata.com/product-download/ftk-imager-version-4-5>

Pro účely analýzy nemusíme vytvářet výpis celého obsahu paměti, ale můžeme vytvořit výpis obsahu virtuální paměti procesu nástrojem jako je například Process Hacker¹².

Analýza obsahu výpisu paměti se provádí pomocí frameworku Volatility3¹³.

¹² Dostupný z webu <https://github.com/processhacker/processhacker>

¹³ Dostupný z webu <https://github.com/volatilityfoundation/volatility3>

4 MASKOVACÍ TECHNIKY MALWARU

Autoři malwaru jsou si vědomi toho, že jejich malware budou předmětem zájmu antivirových společností a zkoumání analytiků. Proto vytvářejí malware, který bude jednak nenápadný svojí činností a zároveň autoři malwaru chtějí, aby byla analytikům nebo antivirovým společnostem co nejvíce znemožněna analýza spustitelného souboru. Proto tvůrci malwaru využívají různé techniky ochrany malwaru jako jsou zabalení (komprimace) obsahu souboru, obfuskace řetězců nebo např. šifrování. [3]

4.1 Zabalení obsahu souboru (packing)

Zabalení obsahu souboru (packing) je metoda ochrany malwaru, která spočívá v tom, že je spustitelný soubor ve své vnitřní části zabalený (komprimovaný). To znamená, že po spuštění dojde k rozbalení vnitřní části a spuštění. Zabalený soubor lze snadno detekovat pomocí funkcí a také tím, že má malý počet importovaných funkcí. Dalším znakem souboru, který má zabalený obsah je vysoká entropie. [18]

Neznamená to, že každý soubor, který využívá zabalení obsahu souboru je škodlivý. Zabalení obsahu spustitelného souboru je legální funkce, která zmenší velikost souboru.

Pro identifikaci zabaleného souboru existuje několik způsobů. Každý nástroj, který umožňuje zabalit obsah souboru má své vlastní jedinečné vlastnosti, pomocí kterých jej můžeme identifikovat. Např. open source UPX packer přejmenuje všechny sekce na UPX1, UPX2 atd. poslední sekci pak na .aspack. [4]

Některé nástroje jako je PEiD nebo CFF Explorer umí na základě signatur identifikovat komprimační modul. [4]

Druhý způsob identifikace zabaleného obsahu souboru (packeru) je dle názvů sekcí. Rozbalený soubor obsahuje názvy jako .code, .data, .idata, .rsrc nebo .reloc. Komprimované soubory obsahují názvy jako UPX1, .aspack, .stub apod. Názvy sekcí lze jednoduše zjistit například pomocí nástroje PEiD.[4]

Další způsob identifikace zabaleného obsahu souboru je ten, že většina nástrojů, které se používají na zabalení obsahu souboru, zabaluje i části souboru PE, včetně tabulky importu a podobně. Poté přidá na konec novou část, která obsahuje kód pro rozbalení (stub). Většina nezabalených souborů provádí spuštění od první sekce, kdežto zabalené soubory se spouštějí od jedné z posledních sekcí, tzn. vstupní bod (entry point) neukazuje na první sekci a

oprávnění je spustitelné, tj. executable (v charakteristice sekce). První sekce bude mít oprávnění nastaveno na čtení/zápis tj.read/write. [3]

Pro získání zabaleného obsahu souboru je možné využít několik způsobů. Jedním z nich je emulace. Emulátory jsou programy, které simulují spouštěcí prostředí. Tyto nástroje jsou vhodné pro bezpečnou analýzu malwaru.

Další využívanou technikou pro získání zabaleného obsahu spustitelného souboru je využití výpisu paměti RAM (dump). Při spuštění zabaleného spustitelného souboru v bezpečném prostředí, dojde k automatickému rozbalení souboru v paměti a po vytvoření výpisu paměti RAM (dump), lze získat rozbalený obsah souboru, například pomocí frameworku Volatility3.

Poslední možností je manuální rozbalení obsahu pomocí jakéhokoli debuggeru. Je nutné nastavit bod zastavení (breakpoint) na místo kódu, kdy již došlo k rozbalení a zároveň nedošlo ke spuštění rozbalené části. Tím lze získat originální soubor. [3]

4.2 Obfuskace

Obfuskace je z anglického jazyka obfuscate tj. zatemnit.

„Obfuskací zdrojového kódu rozumíme snahu o znemožnění analýzy kódu. Při tomto procesu dochází obvykle k přejmenování názvů tříd, metod a proměnných. Volitelně se pak obfuskátor může pokusit odstranit z vašeho kódu prázdné znaky či nepoužívané metody a proměnné.“ [19]

Výsledkem obfuskace je pak kód, který po dekompilaci nedává na první pohled smysl a pro člověka je velmi nesnadné se orientovat v takovémto souboru. [4]

4.3 Šifrování a kódování

Tvůrci malwaru často používají jednoduché metody šifrování nebo kódování, a to z důvodu jakým je například utajení komunikace s řídicími servery, skrytí samotného malwaru před identifikací založené na signaturách nebo ke schování obsahu konfiguračního souboru. Dále mohou být použity k zašifrování informací, které mají být odeslané z napadeného systému, nebo na skrytí řetězců v binárním souboru. Malware používá různé typy šifrování a kódování. [3]

„Šifrování je v podstatě proces úpravy dat nebo informací tak, aby byly nečitelné nebo nepoužitelné bez tajného klíče, který je dán pouze lidem, od kterých se očekává, že si zprávu přečtou. Rozdíl mezi kódováním nebo balením a šifrováním spočívá v tom, že balení nepoužívá žádný klíč a jeho hlavní cíl nesouvisí s ochranou informací nebo omezením přístupu k nim ve srovnání se šifrováním. Existují dvě základní techniky šifrování informací: symetrické šifrování (také nazývané šifrování tajným klíčem) a asymetrické šifrování (také nazývané šifrování veřejného klíče)“ [3]

Tvůrci malwaru používají i silné šifrování. Pro identifikaci použití kryptografických funkcí v binárním kódu můžeme hledat kryptografické identifikátory (podpisy), jako jsou: [3]

- řetězce nebo importy, které odkazují na kryptografické funkce, [3]
- kryptografické konstanty, [3]
- jedinečné sekvence instrukcí používané kryptografickými rutinami. [3]

Pro nalezení kryptografických podpisů ve spustitelných souborech lze využít signsrch. Díky němu lze nalézt relativní virtuální adresy, kde byly detekovány signatury šifrování a následně použít disassembler (například IDA) pro ruční prohledání zdrojového kódu a případné nalezení šifrovacího klíče. Často je velmi obtížné získat klíč pro dešifrování spustitelného souboru. [3]

Autoři malwaru také využívají dynamického šifrování, kdy není zašifrován celý blok souboru najednou, jen všechny řetězce. Každý řetězec je dešifrován těsně předtím, než je použit. Díky tomu nejsou řetězce nikdy dostupné všechny dešifrované, a to může být velká překážka pro analýzu malwaru. Statickou analýzou tyto řetězce zjistit nelze. Tento způsob šifrování je nutné obejít jinými metodami například skriptováním v OllyDbg. [3]

Kódování je převod informace pomocí pevně daného kódu bez nutnosti použití tajného klíče. Mezi tvůrci malwaru je nejpoužívanější způsob kódování base64.

Base64 se původně používalo při komunikaci emailem. Jde o kódování, které převádí binární data do textového formátu na posloupnost tisknutelných znaků. Base64 kóduje tři oktety binárních dat pomocí čtyř znaků ASCII. Používají se znaky velké a malé anglické abecedy, číslice a znaky „+“ a „/“. Pokud počet původních dat není dělitelný třemi, doplní se na konec jeden nebo dva znaky „=“. [20]

Na následujícím obrázku je vidět kódování base64 pro slovo „password“ a „password1“.

```
remnux@remnux:~$ echo -n password | base64
cGFzc3dvcmQ=
remnux@remnux:~$ echo -n password1 | base64
cGFzc3dvcmQx
remnux@remnux:~$
```

Obrázek 4. Ukázka kódovni base64 [zdroj vlastní]

Kódování base64 většinou autoři malwaru používají společně se šifrováním. Příkladem může být síťová komunikace malwaru s řídicím serverem, která je zašifrovaná a zakódovaná pomocí base64. Následně je odeslaná na C&C server. Pro identifikaci kódování base64 lze využít např. YARA pravidla.

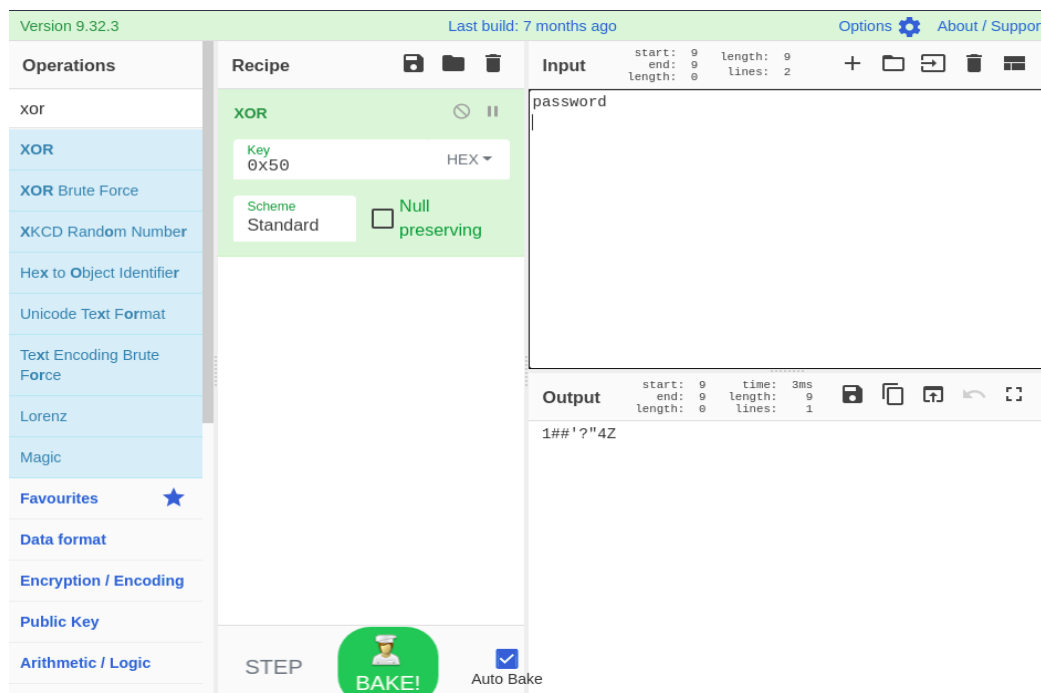
```
„rule contains_base64 : Base64
{
  meta:
    author = "Jaume Martin"
    description = "This rule finds for base64 strings"
    version = "0.2"
    notes = "https://github.com/Yara-Rules/rules/issues/153"
  strings:
    $a = /([A-Za-z0-9+\\]{4}){3,}([A-Za-z0-9+\\]{2})=|[A-Za-z0-9+\\]{3}=)/
  condition:
    $a
}“ [23]
```

Pro skrytí komunikace nebo skrytí řetězců, využívají autoři malwaru také často operaci exkluzivní disjunkce XOR. Pravdivostní tabulka této funkce je v následující tabulce.

Tabulka 1. Pravdivostní tabulka XOR

A	B	A XOR B
0	0	0
1	0	1
0	1	1
1	1	0

Pro příklad, když vezmeme řetězec „password“, na který aplikujeme exkluzivní disjunkci XOR s klíčem 0x50 hex, získáme řetězec „1##'?"4Z“, jak je vidět na Obrázek 5. Ukázka operace XOR textu password pomocí klíče 0x50hex v nástroji CyberChef [zdroj vlastní]v nástroji CyberChef.



Obrázek 5. Ukázka operace XOR textu password pomocí klíče 0x50hex v nástroji CyberChef [zdroj vlastní]

Pro identifikaci lze použít nástroje jako je xorstring. Tento nástroj budou podrobně popsány v kapitole Softwarové nástroje.

5 SOFTWAREVÉ NÁSTROJE

Každá analýza podezřelého souboru by měla začít zjištěním informací o souboru, včetně jeho metadat.

Softwarové nástroje pro analýzu malwaru lze rozdělit do několika skupin. Pro statickou analýzu, pro dynamickou analýzu, nástroje pro práci s výpisem paměti RAM a debugery/dissasembly. Nástroje pro statickou analýzu lze dále rozdělit na nástroje, které zjišťují informace o souboru na základě metadat, heuristik apod.

Z každé níže uvedené skupiny je dobré použít více nástrojů. Každý nástroj využívá jiné techniky přístupu k souborům a tím se mohou výsledky lišit.

5.1 Remnux

Remnux¹⁴ je linuxová distribuce se zaměřením na reverzní analýzu a analýzu malwaru. Jedná se o distribuci založenou na Ubuntu 20.04.4 LTS, do které jsou doinstalovány všechny potřebné nástroje pro účely statické i dynamické analýzy. Před samotnou analýzou je doporučeno zaktualizovat všechny nástroje následujícím příkazem:

```
$ remnux upgrade
```

Tím dojde k automatické aktualizaci veškerých nástrojů. Více informací lze dohledat v přehledné dokumentaci linuxové distribuce.

Remnux se při analýze malwaru pro tuto diplomovou práci velmi osvědčil. Obsahuje velké množství nástrojů, které jsou nakonfigurované pro okamžité použití. Díky tomu, že je distribuce založená na linuxové distribuci Ubuntu, je i uživatelské prostředí velmi příjemné na použití.

5.2 Nástroje pro statickou analýzu vzorku malwaru

Následující nástroje spadají do skupiny nástrojů pro statickou analýzu, to je získání informací ze souboru bez jeho spuštění.

¹⁴ Dostupná na <https://remnux.org>

5.2.1 Nástroj file

Jedná se o nástroj, pomocí kterého lze získat informace o typu souboru. Tento nástroj je součástí všech UNIXových operačních systémů, včetně GNU Linuxu, BSD i macOS. V manuálové stránce je uvedeno, že nástroj file využívá sadu testů, které jsou aplikovány na analyzovaný soubor. Díky tomu dokáže nástroj file spolehlivě určit o jaký typ souboru se jedná, viz Obrázek 6. Nástroj file [zdroj vlastní][21]

```
remnux@remnux:~$ file 7z2107-x64.exe
7z2107-x64.exe: PE32 executable (GUI) Intel 80386, for MS Windows
remnux@remnux:~$
```

Obrázek 6. Nástroj file [zdroj vlastní]

5.2.2 Nástroj TrID

Stejně jako nástroj file je i nástroj TrID¹⁵ určený k identifikaci typů souborů. TrID používá jiný způsob analýzy než nástroj file. TrID analyzuje soubor na základě jejich binárních podpisů a jeho databáze obsahuje kolem 14 500 vzorů podpisů. Další výhodou nástroje je, že ho lze snadno naučit nové typy souborů. [22]

Výstup z nástroje TrID je vidět na následujícím obrázku.

```
remnux@remnux:~$ trid 7z2107-x64.exe
TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
Definitions found: 14589
Analyzing...

Collecting data from file: 7z2107-x64.exe
 38.8% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)
 20.5% (.EXE) Microsoft Visual C++ compiled executable (generic) (16529/12/5)
 13.0% (.EXE) Win64 Executable (generic) (10523/12/4)
  8.1% (.DLL) Win32 Dynamic Link Library (generic) (6578/25/2)
  6.2% (.EXE) Win16 NE executable (generic) (5038/12/1)
remnux@remnux:~$
```

Obrázek 7. Nástroj TrID [zdroj vlastní]

¹⁵ Ke stažení na <https://mark0.net/soft-trid-e.html>

5.2.3 Nástroj Yara-Rules

Yara je nástroj, jehož cílem je klasifikovat vzorky malwaru a lze pomocí něho vytvářet signatury vzorků malwaru. Projekt Yara-Rules¹⁶ má za úkol shromažďovat a udržovat co neaktuálnější signatury Yara a byl od svého začátku vyvíjený jako Open Source (otevřené zdrojové kódy pod licencí GNU-GPLv2). Tím jsou tato pravidla volně dostupná. Pomocí tohoto nástroje lze jednoduše vytvořit statické přehledy a identifikovat škodlivé funkce, jak je vidět na dalším obrázku. [24]

```
remnux@remnux:~/malware$ yara-rules trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
SEH_Init trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
win_files_operation trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
Big_Numbers1 trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
Big_Numbers3 trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
IsPE32 trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
IsWindowsGUI trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
HasOverlay trickbot_55eff10a576b9cef045e4f98becf7a8aeb20a1e4d28b86231377ee6af15eb23
```

Obrázek 8. Nástroj Yara-Rules – malware trickbot [zdroj vlastní]

Detekuje následující kategorie malwaru: [24]

- Anti-debug/Anti-VM – obrana malwaru proti běhu v debugerech a virtuálních počítačích. [24]
- Capabilities – v této sekci jsou Yara pravidla, která nespádají do žádných jiných kategorií. [24]
- CVE – detekce známých zranitelností. [24]
- Crypto – detekce kryptografických algoritmů. [24]
- Exploit Kits – detekce exploitů. [24]
- Malicious Documents – detekce škodlivých dokumentů. [24]
- Malware – identifikace známého malwaru. [24]
- Packers – odhalení známých softwarových zabalovačů (komprimátorů). [24]

¹⁶ Ke stažení na <https://github.com/Yara-Rules/rules>

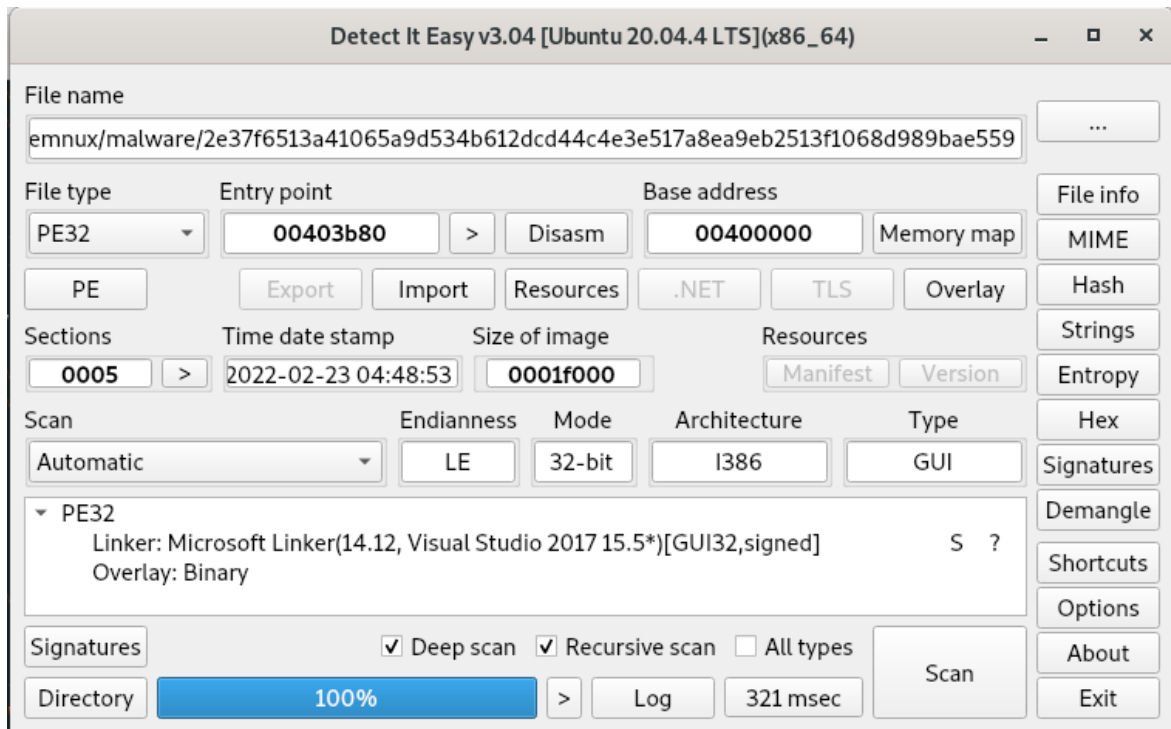
- WebShells – odhalení známých webshellů. [24]
- Email – identifikace škodlivých emailů. [24]
- Malware Mobile – identifikace mobilního malwaru. [24]
- Deprecated – zastaralá Yara pravidla. [24]

5.2.4 Nástroj Detect-It-Easy

Detect-It-Easy¹⁷, zkráceně DIE, je nástroj pro určování typu souboru. Jedná se o multiplatformní nástroj pro MS Windows, Linux a MacOS. DIE má otevřenou platformu, do které lze přidávat vlastní algoritmy detekcí pomocí skriptů. Existuje ve třech verzích, kdy základní plná verze se spouští pomocí příkazu „die“ a obsahuje grafické rozhraní. Druhá verze je verze Lite se spouští pomocí příkazu „diel“ má grafické rozhraní a obsahuje pouze základní informace. Poslední verze DIE je pro příkazový řádek a spouští se pomocí příkazu „diec“. [25]

Jak je vidět na Obrázek 9. Nástroj Detect It Easy – malware HermeticWipper [zdroj vlastní] nástroj obsahuje množství informací ze statické analýzy. Lze zjistit nejen typ souboru a architekturu, pro kterou je soubor vytvořený, ale i datum a čas kompilace souboru, vstupní bod (entry point) nebo například typ překladače. Mezi pokročilé funkce, které nástroj poskytuje je vyhledání řetězců (strings), entropii částí souboru, hexadecimální editor a další. [25]

¹⁷ Ke stažení na <https://github.com/horsicq/Detect-It-Easy>



Obrázek 9. Nástroj Detect It Easy – malware HermeticWipper [zdroj vlastní]

5.2.5 Nástroj ExifTool

Nástroj ExifTool¹⁸ vypisuje metadata z analyzovaného souboru. Nástroj je dostupný pro platformy MS Windows, Linux a MacOS. Tento nástroj podporuje velké množství typů souborů. Je velmi užitečný pro analýzu metadat z multimediálních souborů, ale lze ho využít i na spustitelné soubory. [26]

Na následujícím obrázku je vidět výstup z nástroje exiftool. Jako vzorek byl použitý malware HermeticWiper.

¹⁸ Ke stažení na <https://exiftool.org>

```
remnux@remnux:~/malware$ exiftool a58964d0bc832f9f30b4c956e097fb77615f42893c654b979075160731ee1e5a
ExifTool Version Number      : 12.16
File Name                    : a58964d0bc832f9f30b4c956e097fb77615f42893c654b979075160731ee1e5a
Directory                   : .
File Size                    : 2.5 MiB
File Modification Date/Time  : 1979:12:31 00:00:00-05:00
File Access Date/Time       : 2022:04:02 12:27:20-04:00
File Inode Change Date/Time  : 2022:04:02 12:25:27-04:00
File Permissions             : rw-rw-r--
File Type                    : Win64 EXE
File Type Extension         : exe
MIME Type                    : application/octet-stream
Machine Type                 : AMD AMD64
Time Stamp                   : 0000:00:00 00:00:00
Image File Characteristics   : No relocs, Executable, Large address aware, No debug
PE Type                      : PE32+
Linker Version               : 3.0
Code Size                    : 1999360
Initialized Data Size        : 101888
Uninitialized Data Size      : 0
Entry Point                  : 0x53600
OS Version                   : 4.0
Image Version                : 1.0
Subsystem Version            : 4.0
Subsystem                    : Windows command line
Warning                      : Error processing PE data dictionary
remnux@remnux:~/malware$
```

Obrázek 10. Nástroj exiftool – malware HermeticWiper [zdroj vlastní]

5.2.6 Nástroj signsrch

Nástroj signsrch¹⁹ dokáže ve spustitelném souboru nalézt šifrovací, kompresní a kódovací algoritmy. Tento nástroj je multiplatformní a je dostupný na platformy MS Windows, Linux a MacOS. [27]

¹⁹ Ke stažení na <https://aluigi.altervista.org/mytoolz.html>

```

- open file "062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025"
- 3723264 bytes allocated
- load signatures
- open file /usr/share/signsrch/signsrch.sig
- 3075 signatures in the database
- start 1 threads
- start signatures scanning:

offset  num  description [bits.endian.size]
-----
0000a63a 1283 Windows CryptAcquireContext [..21]
00010734 3050 compression algorithm seen in the game DreamKiller [32.le.12&]
000197f4 2545 anti-debug: IsDebuggerPresent [..17]
000388e2 3052 function where is handled the ZipCrypto password [32.le.12&]
0003a559 2417 MBC2 [32.le.248&]
0003a55c 2418 MBC2 [32.be.248&]
0003aaa0 894 AES Rijndael S / ARIA S1 [..256]
0003aba0 895 AES Rijndael Si / ARIA X1 [..256]
0003aca0 896 Rijndael Te0 (0xc66363a5U) [32.le.1024]
0003b0a0 898 Rijndael Te1 (0xa5c66363U) [32.le.1024]
0003b4a0 900 Rijndael Te2 (0x63a5c663U) [32.le.1024]
0003b8a0 902 Rijndael Te3 (0x6363a5c6U) [32.le.1024]
0003bca0 905 Rijndael Td0 (0x51f4a750U) [32.le.1024]
0003c0a0 907 Rijndael Td1 (0x5051f4a7U) [32.le.1024]
0003c4a0 909 Rijndael Td2 (0xa75051f4U) [32.le.1024]
0003c8a0 911 Rijndael Td3 (0xf4a75051U) [32.le.1024]
0003dca7 2412 Noekeon Nessie round [..17]
0003dd4c 3038 unlz table three [32.le.64]
0003ef10 2291 zlib_inflate_lengthStarts [32.le.116]
0003ef89 2295 zlib_inflate_lengthExtraBits [32.be.116]
0003ef8c 2294 zlib_inflate_lengthExtraBits [32.le.116]
0003f008 2298 zlib_inflate_distanceStarts [32.le.120]
0003f080 2303 zlib_inflate_distanceExtraBits [32.le.120]
0003f0f8 648 CRC-32-IEEE 802.3 [crc32.0xedb88320 lenorev 1.1024]
0003f0f8 641 CRC-32-IEEE 802.3 [crc32.0x04c11db7 le rev int_min.1024]
00041174 1289 Windows CryptDecrypt [..13]
000411a4 1285 Windows CryptImportKey [..15]

- 27 signatures found in the file in 21 seconds
- done
remux@remux:~/malware$

```

Obrázek 11. Nástroj signsrch – malware WannaCry [zdroj vlastní]

5.2.7 Nástroj ssdeep

Nástroj ssdeep²⁰ umí vypočítat Context Triggered Piecewise Hashes (CTPH), známé jako fuzzy hashes. Jedná se o identifikaci téměř identických souborů pomocí kontextu spouštěného po jednotlivých částech. [28]

Díky tomu lze vyhledávat podobné soubory, které se v některých částech mohou lišit, ale vycházejí ze společného základu. Na níže uvedeném obrázku je vidět výpočet CTPH na dvou různých souborech, které jsou oba z rodiny malwaru TrickBot, kde je vidět podobnost vypočteného CTPH.

²⁰ Ke stažení na <https://ssdeep-project.github.io/ssdeep/index.html>

```
remnux@remnux:~/malware/trickbot$ ssdeep trickbot_063dc766643efa2ff9524cda6d9327e2a4b4295fbc6a2b1b944b4e8d4ff32f1f.dll
ssdeep,1.1--blocksize:hash:hash,filename
768:DLR7ix70z0ozpW7YYCqgio1ph30yqwu3YME5g8:DLR7ix70zDzpW7YYiio1pzyis,"/home/remnux/malware/trickbot/trickbot_063dc766643efa2ff9524cda6d9327e2a4b4295fbc6a2b1b944b4e8d4ff32f1f.dll"
remnux@remnux:~/malware/trickbot$ ssdeep trickbot_55eff10a576b9cef045e4f98becf7a8aebe20a1e4d28b86231377ee6af15eb23
ssdeep,1.1--blocksize:hash:hash,filename
768:DLR7ix70z0ozpW7YYCqgio1ph30yqwu3YME5pfm:DLR7ix70zDzpW7YYiio1pzyij,"/home/remnux/malware/trickbot/trickbot_55eff10a576b9cef045e4f98becf7a8aebe20a1e4d28b86231377ee6af15eb23"
remnux@remnux:~/malware/trickbot$
```

Obrázek 12. Nástroj ssdeep – malware tricbot [zdroj vlastní]

5.2.8 Nástroj bulk_extractor

Nástroj `bulk_extractor`²¹ je forenzní nástroj, který je možné použít i pro účely statické analýzy. Nástroj prohledá jakýkoli druh vstupu (obrazy disků, soubory, adresáře souborů atd.) a extrahuje strukturované informace jako jsou e-mailové adresy, čísla kreditních karet, JPEG a útržky JSON. Není nutné připojovat a analyzovat samotný souborový systém. Výsledky jsou uloženy do textových souborů. `Bulk_extractor` zkoumá každý bajt vstupu a zjišťuje, zda se jedná o začátek sekvence, kterou lze dekomprimovat nebo jinak dekodovat. Pokud ano, dekodovaná data jsou rekurzivně znovu prozkoumána. Výsledkem je, že `bulk_extractor` dokáže najít např. JPEGy kódované v BASE64 nebo komprimované objekty JSON a podobně. [29]

Na následujícím obrázku je výstup z nástroje `bulk_extractor`, kdy jako vzorek malwaru byl použitý `HermeticWiper`.

```
remnux@remnux:~/malware/bulk_wiper$ ls
aes_keys.txt          httplogs.txt          unrar_carved.txt
alerts.txt            ip_histogram.txt     unzip_carved.txt
ccn_histogram.txt     ip.txt               url_facebook-address.txt
ccn_track2_histogram.txt jpeg_carved.txt      url_facebook-id.txt
ccn_track2.txt        json.txt             url_histogram.txt
ccn.txt              kml.txt             url_microsoft-live.txt
domain_histogram.txt ntfsindx_carved.txt url_searches.txt
domain.txt           ntfslogfile_carved.txt url_services.txt
elf.txt              ntfsmft_carved.txt  url.txt
email_domain_histogram.txt ntfsusn_carved.txt  utmp_carved.txt
email_histogram.txt  pii_teamviewer.txt  vcard.txt
email.txt            pii.txt              windirs.txt
ether_histogram.txt  rar.txt             winlnk.txt
ether.txt            report.xml           winpe_carved.txt
evtx_carved.txt      rfc822.txt          winpe.txt
exif.txt             sin.txt             winprefetch.txt
find_histogram.txt   sqlite_carved.txt   zip.txt
find.txt            telephone_histogram.txt
gps.txt             telephone.txt
remnux@remnux:~/malware/bulk_wiper$
```

Obrázek 13. Nástroj bulk_extractor – malware HermeticWiper

²¹ Ke stažení na https://github.com/simsong/bulk_extractor/

5.2.9 Nástroj manalyze

Nástroj manalyze²² je pokročilý nástroj pro platformu MS Windows a Linux, který je vydaný pod licencí GPLv3. Jedná se o robustní analyzátor souborů PE viz. Obrázek 14. Nástroj manalyze (část výstupu) – malware HermeticWiper [zdroj vlastní] Dokáže identifikovat kompilátor, detekuje zabalený obsah souboru, hledá podezřelé řetězce i podezřelé kombinace importu (např. WriteProcessMemory + CreateRemoteThread), detekuje kryptografické funkce a může odesílat hash souboru do služby VirusTotal. [30]

```
remnux@remnux:~/malware$ manalyze --dump=imports,sections --hashes 062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025
* Manalyze 0.9 *

-----
/home/remnux/malware/062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025
-----

Sections:
-----
.text:
  MD5:                570b25a22563c416fafb8a553e60afda
  SHA1:               4aad2b9141d5a3026effd139b2679308f7cd5a9
  SHA256:             c70b5004cbabd931b3976dbd2ea42d3f8b5110e5753fba29a45636dc209adf7c
  SHA3:               1b02ac078bd307f6154989427c9ab92a78e7113492c4d09b410b8fc23202a13d
  VirtualSize:        0x00008BCA
  VirtualAddress:     0x00001000
  SizeOfRawData:      0x00009000
  PointerToRawData:   0x00001000
  PointerToRelocations: 0x00000000
  PointerToLineNumbers: 0x00000000
  NumberOfLineNumbers: 0
  NumberOfRelocations: 0
  Characteristics:    IMAGE_SCN_CNT_CODE
                     IMAGE_SCN_MEM_EXECUTE
                     IMAGE_SCN_MEM_READ
  Entropy:            6.13445

.rdata:
  MD5:                d8037d744b539326c06e897625751cc9
  SHA1:               8c528f41cd4533228264ee639fad17e5be8bf817
  SHA256:             532e9419f23eaf5eb0e8828b211a7164cbf80ad54461bc748c1ec2349552e6a2
  SHA3:               5d39450e3d956e25cae7060d586eba36290927433c53df9b26f417ec5d91052d
```

Obrázek 14. Nástroj manalyze (část výstupu) – malware HermeticWiper [zdroj vlastní]

²² Ke stažení na <https://github.com/JusticeRage/Manalyze>

Nástroj strings

Nástroj strings je součástí většiny linuxových distribucí. Nástroj strings vyhledá a zobrazí veškeré řetězce uložené v ASCII znacích. Pomocí parametrů je možné nastavit minimální a maximální velikost řetězců. [31]

Na následujícím obrázku je vidět část výstup z nástroje strings.

```
InterlockedDecrement
CloseHandle
TerminateThread
WaitForSingleObject
InterlockedIncrement
GetCurrentThreadId
GetCurrentThread
ReadFile
GetFileSize
CreateFileA
MoveFileExA
SizeofResource
LockResource
LoadResource
FindResourceA
GetProcAddress
GetModuleHandleW
ExitProcess
GetModuleFileNameA
LocalFree
LocalAlloc
KERNEL32.dll
CryptAcquireContextA
CryptGenRandom
StartServiceA
CloseServiceHandle
CreateServiceA
OpenSCManagerA
SetServiceStatus
ChangeServiceConfig2A
RegisterServiceCtrlHandlerA
StartServiceCtrlDispatcherA
OpenServiceA
ADVAPI32.dll
WS2_32.dll
??1_Lockit@std@@QAE@XZ
??0_Lockit@std@@QAE@XZ
```

Obrázek 15. Nástroj strings (část výstupu) – malware HermeticWiper [zdroj vlastní]

5.2.10 Nástroj stringsifter

StringSifter²³ je pokročilý nástroj. Jako vstup bere seznam řetězců strings a vrací tytéž řetězce seřazené pomocí modelu strojového učení tak aby byly seřazeny podle jejich relevance pro analýzu malwaru, viz. Obrázek 16. Nástroj stringsifter (část výstupu) – malware HermeticWiper

[zdroj vlastní] Je vyvíjený společností FireEye Inc. pod licenci Apache Licence 2.0. [32]

Tento nástroj se spouští příkazem „flarestrings“. Pomocí parametrů lze nastavit minimální a maximální velikost řetězců.

```
seLoadDriverPrivilege
ServicesActive
SeBackupPrivilege
C:\Windows\SYSTEM
C:\System Volume Information
Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced
ShowCompColor
ShowInfoTip
SYSTEM\CurrentControlSet\Control\CrashControl
CrashDumpEnabled
$ATTRIBUTE_LIST
$EA_INFORMATION
$SECURITY_DESCRIPTOR
$DATA
$INDEX_ROOT
$INDEX_ALLOCATION
$BITMAP
$REPARSE_POINT
$LOGGED_UTILITY_STREAM
$I30
:;$INDEX_ALLOCATION
RCDATA
DRV_X64
DRV_X86
DRV_XP_X64
DRV_XP_X86
\Dev
<<<obsolete>>
RAny use of this
constitutes acceptance of the DigiCert CP/CPS and the Relying Party Agreement which limit liability and are incorporated herein by reference
remnux@remnux:~/malware/hermetic$
```

Obrázek 16. Nástroj stringsifter (část výstupu) – malware HermeticWiper

[zdroj vlastní]

5.2.11 Nástroj floss

Nástroj floss²⁴ (FLARE Obfuscated Strings Solver) umí v analyzovaném souboru vyhledat řetězce. Využívá pokročilé techniky statické analýzy k automatické deobfuskaci řetězců z malwarových binárních souborů. [33]

FLOSS extrahuje všechny následující typy řetězců:

²³ Ke stažení na <https://github.com/fireeye/stringsifter>

²⁴ Ke stažení na <https://github.com/mandiant/flare-floss>

- statické řetězce: "běžné" řetězce ASCII a UTF-16LE, [33]
- dekódované řetězce: řetězce dekódované ve funkci, [33]
- zásobníkové (stack) řetězce: řetězce vytvořené v zásobníku za běhu, [33]
- tight řetězce: speciální forma zásobníkových řetězců, dekódovaných na zásobníku. [33]

Na následujícím obrázku je vidět část výstupu z nástroje floss. Jako vzorek malwaru byl použitý malwaru HermeticWiper.

```
! "#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ[\]^_`ABCDEFGHIJKLMNPQRSTUVWXYZ{|}~
GetProcAddress
GetProcessWindowStation
GetObjectInformationW
GetLastActivePopup
GetActiveWindow
MessageBoxW
! "#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~
CloseHandle
WriteFile
CreateFileA
SizeofResource
LockResource
LoadResource
FindResourceA
CreateProcessA
KERNEL32.dll
GetCurrentThreadId
FlsSetValue
GetCommandLineA
DecodePointer
UnhandledExceptionFilter
SetUnhandledExceptionFilter
IsDebuggerPresent
RtlVirtualUnwind
RtlLookupFunctionEntry
RtlCaptureContext
EncodePointer
TerminateProcess
GetCurrentProcess
RtlUnwindEx
FlsGetValue
FlsFree
SetLastError
GetLastError
FlsAlloc
HeapFree
Sleep
```

Obrázek 17. Nástroj floss (část výstupu) – malware HermeticWiper [zdroj vlastní]

5.2.12 Nástroj PEframe

Nástroj PEframe²⁵ je open source nástroj určený pro statickou analýzu souborů PE a Microsoft Office. Dokáže detekovat v analyzovaném souboru komprimaci, XOR, digitální podpisy, funkce pro anti-debug a anti-vm, makra a další. [34]

Na následujícím obrázku je vidět část výstupu z nástroje PEframe. Jako vzorek malwaru byl použitý malwaru HermeticWiper.

```
-----
File Information (time: 0:00:11.652012)
-----
filename      062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025
filetype     PE32 executable (GUI) Intel 80386, for MS Windows
filesize     3723264
hash sha256  062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025
virustotal   /
imagebase    0x400000
entrypoint   0x9a16
imphash      9ecee117164e0b870a53dd187cdd7174
datetime     2010-11-20 09:03:08
dll          False
directories   import, tls, relocations, resources *
sections     .rdata, .rsrc, .text *, .data *
features     mutex, packer, crypto

-----
Yara Plugins
-----
IsPE32
IsWindowsGUI
HasRichSignature
CRC32 poly Constant
CRC32 table
Rijndael AES
Rijndael AES CHAR
Rijndael AES LONG

-----
Behavior
-----
anti dbg
Xor
win registry
win files operation
```

Obrázek 18. Nástroj PEframe (část výstupu) – malware HermeticWiper

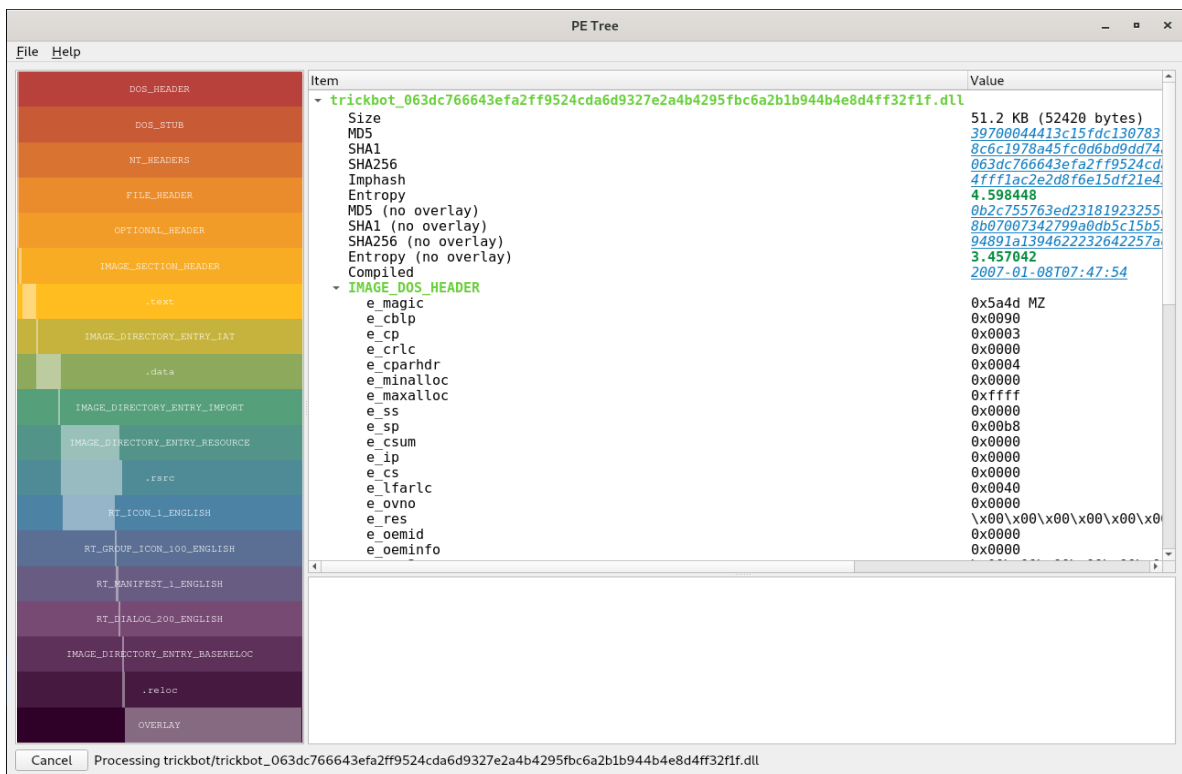
[zdroj vlastní]

²⁵ Ke stažení na <https://github.com/digitalsleuth/peframe>

5.2.13 Nástroj PE Tree

PE Tree²⁶ je modul Pythonu pro prohlížení souborů PE. Lze jej také použít s nástroji IDA Pro, Ghidra, Volatility k prohlížení výpisu souborů PE v paměti, stejně jako k provádění rekonstrukce importní tabulky. [35]

Na níže uvedeném obrázku je vidět část výstupu z nástroje PE Tree. Jako vzorek malwaru byl použitý malwaru Trickbot.



Obrázek 19. Nástroj PE Tree – malware trickbot [zdroj vlastní]

²⁶ Ke stažení na https://github.com/blackberry/pe_tree

5.2.14 Nástroj PEdump

PEdump²⁷ je nástroj pro statickou analýzu PE souborů, ze kterých umí extrahovat různé komponenty jako například MZ Header, DOS stub, Rich Header, PE Header, Data Directory, Resources, Strings, Imports, Export, VS_Versioninfo parsing, Packer/Compiler detection. Tyto komponenty umí vyextrahovat pro případnou další analýzu. [36]

```
remnux@remnux:~/malware$ pedit 062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025

=== MZ Header ===

    signature:                "MZ"
    bytes_in_last_block:      144      0x90
    blocks_in_file:           3         3
    num_relocs:                0         0
    header_paragraphs:        4         4
    min_extra_paragraphs:     0         0
    max_extra_paragraphs:     65535    0xffff
    ss:                        0         0
    sp:                        184      0xb8
    checksum:                  0         0
    ip:                        0         0
    cs:                        0         0
    reloc_table_offset:       64        0x40
    overlay_number:           0         0
    reserved0:                 0         0
    oem_id:                    0         0
    oem_info:                  0         0
    reserved2:                 0         0
    reserved3:                 0         0
    reserved4:                 0         0
    reserved5:                 0         0
    reserved6:                 0         0
    lfanew:                    248      0xf8

=== DOS STUB ===

00000000: 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 |.....!.L.!Th|
00000010: 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f |is program canno|
00000020: 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 |t be run in DOS |
00000030: 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 |mode....$.|

=== RICH Header ===

ID  VER      COUNT  DESCRIPTION
c  1c7b         1
e  1c83         4
a  1f6f        11
b  1f6f         1
```

Obrázek 20. Nástroj PEdump (část výstupu) – malware HermetiWiper

[zdroj vlastní]

²⁷ Ke stažení na <https://github.com/zed-0xff/pedit>

5.2.15 Nástroj pecheck

Pecheck²⁸ je nástroj pro analýzu souborů PE. Tento nástroj vypočítává hashe jednotlivých sekcí, detekuje podezřelé API a overlay. Dále umí extrahovat jednotlivé sekce a uložit je do souboru, viz. Obrázek 21. Nástroj pecheck (část výstupu) – malware HermeticWiper [zdroj vlastní]. Tento nástroj také umí detekovat i digitální podpisy. [37]

```
PE check for '062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025':
Entropy: 3.822108 (Min=0.0, Max=8.0)
MD5 hash: 172dd02cdcc1323389597a85919e8282
SHA-1 hash: 796623e814f5c4797c0cbf97383bd47a99b871a8
SHA-256 hash: 062b5b4be6a3dc885e87b1eae706e37c63e9e59e9a948b7db457988b6547c025
SHA-512 hash: 3d35f89161aaaa96a4085a61b7c2dd7d2eeb78b2d8215573e649c278dd4a73676a9b495e3f4f2e10c88126f9881909158e6
be19f340b9b1d76011869a04dd8
.text entropy: 6.134451 (Min=0.0, Max=8.0)
.rdata entropy: 3.503616 (Min=0.0, Max=8.0)
.data entropy: 6.098497 (Min=0.0, Max=8.0)
.rsrc entropy: 3.632421 (Min=0.0, Max=8.0)
Dump Info:
-----Parsing Warnings-----

Byte 0x00 makes up 63.9180% of the file's contents. This may indicate truncation / malformation.

Invalid VS_VERSION_INFO block:

-----DOS_HEADER-----

[IMAGE_DOS_HEADER]
0x0 0x0 e_magic: 0x5A4D
0x2 0x2 e_cblp: 0x90
0x4 0x4 e_cp: 0x3
0x6 0x6 e_crlc: 0x0
0x8 0x8 e_cparhdr: 0x4
0xA 0xA e_minalloc: 0x0
0xC 0xC e_maxalloc: 0xFFFF
0xE 0xE e_ss: 0x0
0x10 0x10 e_sp: 0xB8
0x12 0x12 e_csum: 0x0
0x14 0x14 e_ip: 0x0
0x16 0x16 e_cs: 0x0
0x18 0x18 e_lfarlc: 0x40
0x1A 0x1A e_ovno: 0x0
0x1C 0x1C e_res:
0x24 0x24 e_oemid: 0x0
0x26 0x26 e_oeminfo: 0x0
0x28 0x28 e_res2:
0x3C 0x3C e_lfanew: 0xF8

-----NT HEADERS-----
```

Obrázek 21. Nástroj pecheck (část výstupu) – malware HermeticWiper

[zdroj vlastní]

²⁸ Ke stažení na <https://blog.didierstevens.com/2020/03/15/pecheck-py-version-0-7-10/>

5.2.16 Nástroj base64dump

Base64dump²⁹ je nástroj, který umí vyhledat a dekodovat řetězce ve spustitelném souboru, které jsou kódované pomocí base64. Volitelně lze dekodovat i další způsoby kódování jako je Unicode, hex little-endian, hex big-endian a další. [38]

Na níže uvedeném obrázku je uveden výstup z nástroje base64dump na vzorku malwaru Trickbot.

```
remnux@remnux:~/malware$ base64dump.py trickbot/trickbot_063dc766643efa2ff9524cda6d9327e2a4b4295fbc6a2b1b944b4e8d4f32f1f.dll
ID  Size  Encoded  Decoded  md5 decoded
--  ----  -
1:  4  This  N..  501e2a2fa18a6fc751a2c926cfd87f8d
2:  4  mode  ..^  fb5d311465160621672451ab9138b859
3:  4  text  ..m  d235bf01945835d5f31a59b90c1a3a9f
4:  4  data  u.Z  359f5eaf26fbf0327b38595ddede06e2
5:  4  rsrc  ...  cbdcd1d6e9334099f97fedfbbecef6356
6:  4  QShd  A(]  c937a8ca87929bb2201bc1932d26b798
7:  4  OWPR  9c.  025dacf44fff34bf1a1418602f1a453d
8:  4  333/  .}  4f422f8349f7fa81287e50e956e152d3
9:  4  //3/  ..  20c57f643d1415ace59dd3e879ed7344
10: 4  SVWP  IU.  0399a39851d3567680fffc939a323e6e
11: 4  gucq  ..*  275180e13144e7de9662823d8aceb89d
12: 4  a2tE  kkD  11aacec0f962b1835b5992915888c2e9
13: 4  G01E  .MD  2c653b07d7738f71fa07bb3224126a6a
14: 4  w5uQ  ...  c27a248f6127b504743577585a374447
15: 8  SetFocus  I.E...  23eb2ec3f1367fa1fccba6c4f92a4f56
16: 8  EndPaint  ..wOj).  ec794af80fa0566fdd3f514984e21857
17: 12 SetLastError  I.Kj.D..+  d40f1b1fbb7e3a4561d16f80a8683d98
18: 8  ReadFile  E...)^  461893e3851e9158b68aec65aeafae9
19: 16 GetModuleHandleA  ..L...xv.vw.  1b7ad174aff72b50b2484077b9fe6e0c
20: 12 GetLastError  ..Kj.D..+  02712c754e21da2bcfdaf6f0bb8d207d
21: 8  KERNEL32  (DM...  4d4b2515f9fbc64660b482fb5f9ac80d
22: 8  TextOutA  M.m:.@  4628851bcf9ea1a0c527dd428ac13ddd
23: 8  COMCTL32  ...L..  240547bf3defb8667ecfb7ddef2fb11a
24: 8  version=  ....  e785bc51de0d390cd253667578713e63
25: 8  assembly  j....r  18f48e3349683c1c83213a713175e10a
26: 16 manifestVersion=  ...}-U...  c79f78f9cc6a44d32e93ef8f45bb11a
27: 16 assemblyIdentity  j....r!...+r  5247102280526c482016a0ab38bfca15
28: 8  version=  ....  e785bc51de0d390cd253667578713e63
29: 4  Name  5..  9a5b9d460674a776b6ad8aa8c0edb6ce
30: 12 /description  ...r...*'  83fc3d344d590a5290dd0edb2339abfa
31: 16 assemblyIdentity  j....r!...+r  5247102280526c482016a0ab38bfca15
32: 8  Controls  ....l  a6b863f39aeb2637c528e49a1574a9c7
33: 8  version=  ....  e785bc51de0d390cd253667578713e63
34: 16 6595b64144ccf1df  ..yo.5...W_  688a37f56ad70a60486f0dab039e467e
35: 4  888=  ..  a6df67456b70cb6616bc0964ee33596e
remnux@remnux:~/malware$
```

Obrázek 22. Nástroj base64dump – malware trickbot [zdroj vlastní]

²⁹ Ke stažení na <https://blog.didierstevens.com/2020/07/03/update-base64dump-py-version-0-0-12/>

5.2.17 Nástroj xorsearch

XORSearch je nástroj pro vyhledávání daného řetězce v binárním souboru kódovaném XOR, ROL, ROT a SHIFT. Soubor se kódováním ROL nebo ROR má znaky otočené o určitý počet bitů. Soubor s ROT znaky otočené o určitý počet pozic. Soubor s kódováním SHIFT má znaky posunuté doleva o určitý počet bitů. Tyto způsoby kódování používají autoři malwaru pro obfuskaci řetězců, jako např. adresy URL, viz. Obrázek 23. Nástroj xorsearch (část výstupu) – malware HermeticWiper [zdroj vlastní] [39]

```
remnux@remnux:~/malware/hermetic$ xorsearch -i 2e37f6513a41065a9d534b612dcd44c4e3e517a8ea9eb2513f1068d989bae559 http
Found XOR 00 position 12EDE: https://www..i.s.
Found XOR 00 position 15594: http.s://www...is.`.com./rpa (c).101.0,.a...%.fCl
Found XOR 00 position 1A5FF: http.s://www...is. .com./rpa (c).101.0,!...%&Cl
Found XOR 00 position 1BDD0: http://crl3.digicert.com/EVCodeSigningSHA2-g1.crl0
Found XOR 00 position 1BE09: http://crl4.digicert.com/EVCodeSigningSHA2-g1.crl0
Found XOR 00 position 1BE62: http://www.digicert.com/CPS0...g...0~..+.....r
Found XOR 00 position 1BEA4: http://ocsp.digicert.com0H...+...0.<http://cacer
Found XOR 00 position 1BECA: http://cacerts.digicert.com/DigiCertEVCodeSigningC
Found XOR 00 position 1C2D7: http://ocsp.digicert.com0I...+...0..=http://cacer
Found XOR 00 position 1C2FD: http://cacerts.digicert.com/DigiCertHighAssuranceE
Found XOR 00 position 1C350: http://crl3.digicert.com/DigiCertHighAssuranceEVRo
Found XOR 00 position 1C392: http://crl4.digicert.com/DigiCertHighAssuranceEVRo
Found XOR 00 position 1C3FE: http://www.digicert.com/ssl-cps-repository.htm0..d
Found XOR 20 position 12EDE: HTTPS...WWW.(
Found XOR 20 position 15594: HTTP.S...WWW...IS.@.COM..RPA
Found XOR 20 position 1A5FF: HTTP.S...WWW...IS.
Found XOR 20 position 1BDD0: HTTP...CRL..DIGICERT.COM.evCODeSIGNINGsha..G..CRL.
Found XOR 20 position 1BE09: HTTP...CRL..DIGICERT.COM.evCODeSIGNINGsha..G..CRL.
Found XOR 20 position 1BE62: HTTP...WWW.DIGICERT.COM.cps.'&%G.,!#.^&(!%!!$R
Found XOR 20 position 1BEA4: HTTP...OCSP.DIGICERT.COM.h&(!%!'."..HTTP...CACER
Found XOR 20 position 1BECA: HTTP...CACERTS.DIGICERT.COM.dIGICERTevCODeSIGNINGc
Found XOR 20 position 1C2D7: HTTP...OCSP.DIGICERT.COM.1&(!%!'."..HTTP...CACER
Found XOR 20 position 1C2FD: HTTP...CACERTS.DIGICERT.COM.dIGICERTIGHaSSURANCEe
Found XOR 20 position 1C350: HTTP...CRL..DIGICERT.COM.dIGICERTIGHaSSURANCEevr0
Found XOR 20 position 1C392: HTTP...CRL..DIGICERT.COM.dIGICERTIGHaSSURANCEevr0
Found XOR 20 position 1C3FE: HTTP...WWW.DIGICERT.COM.SSL.CPS.REPOSITORY.HTM..!D
Found ADD E0 position 12EDE: HTTPS...WWW..
Found ADD E0 position 15594: HTTP.S...WWW..|.IS.@.COM..RPA
Found ADD E0 position 1A5FF: HTTP.S...WWW...IS.
Found ADD E0 position 1BDD0: HTTP...CRL..DIGICERT.COM.%6#0DE3IGNING3(!..G..CRL.
Found ADD E0 position 1BE09: HTTP...CRL..DIGICERT.COM.%6#0DE3IGNING3(!..G..CRL.
Found ADD E0 position 1BE62: HTTP...WWW.DIGICERT.COM.#03...Ga...^.....R
Found ADD E0 position 1BEA4: HTTP...OCSP.DIGICERT.COM.(.....f.HTTP...CACER
Found ADD E0 position 1BECA: HTTP...CACERTS.DIGICERT.COM.$IGI#ERT%6#0DE3IGNING#
Found ADD E0 position 1C2D7: HTTP...OCSP.DIGICERT.COM.).....f.HTTP...CACER
Found ADD E0 position 1C2FD: HTTP...CACERTS.DIGICERT.COM.$IGI#ERT(IGH!SSURANCE%
Found ADD E0 position 1C350: HTTP...CRL..DIGICERT.COM.$IGI#ERT(IGH!SSURANCE%620
Found ADD E0 position 1C392: HTTP...CRL..DIGICERT.COM.$IGI#ERT(IGH!SSURANCE%620
Found ADD E0 position 1C3FE: HTTP...WWW.DIGICERT.COM.SSL.CPS.REPOSITORY.HTM..b.D
```

Obrázek 23. Nástroj xorsearch (část výstupu) – malware HermeticWiper

[zdroj vlastní]

5.2.18 Nástroj capa

Capa je open source nástroj pro analýzu malwaru. Tento nástroj poskytuje framework pro zjištění kódování, rozpoznání a sdílení chování malwaru. Detekuje známé škodlivé funkce ve spustitelných souborech typu PE, ELF a shellcode. Umí detekovat např. backdoor, komunikaci HTTP a další. [40]

Na níže uvedeném obrázku je uveden výstup z nástroje capa na vzorku malwaru HermeticWiper.

```
remnux@remnux:~/malware/hermetic$ ./capa.exe
Loading : 100% | 487/487 [00:00<00:00, 1579.18 rules/s]
matching: 0 functions [00:00, ? functions/s]

-----
md5          | 78535025d6cd3c0b529cf621eafe9482
sha1         | 95e43734cb2cb1a44f9a04cf686e6086690eabcd
sha256       | a58964d0bc832f9f30b4c956e097fb77615f42893c654b979075160731ee1e5a
path         | a58964d0bc832f9f30b4c956e097fb77615f42893c654b979075160731ee1e5a
-----

ATT&CK Tactic | ATT&CK Technique
DEFENSE EVASION | Obfuscated Files or Information [T1027]
-----

MBC Objective | MBC Behavior
DATA          | Check String [C0019]
              | Encode Data::Base64 [C0026.001]
-----

CAPABILITY    | NAMESPACE
compiled with Go | compiler/go
reference Base64 string | data-manipulation/encoding/base64
-----

remnux@remnux:~/malware/hermetic$
```

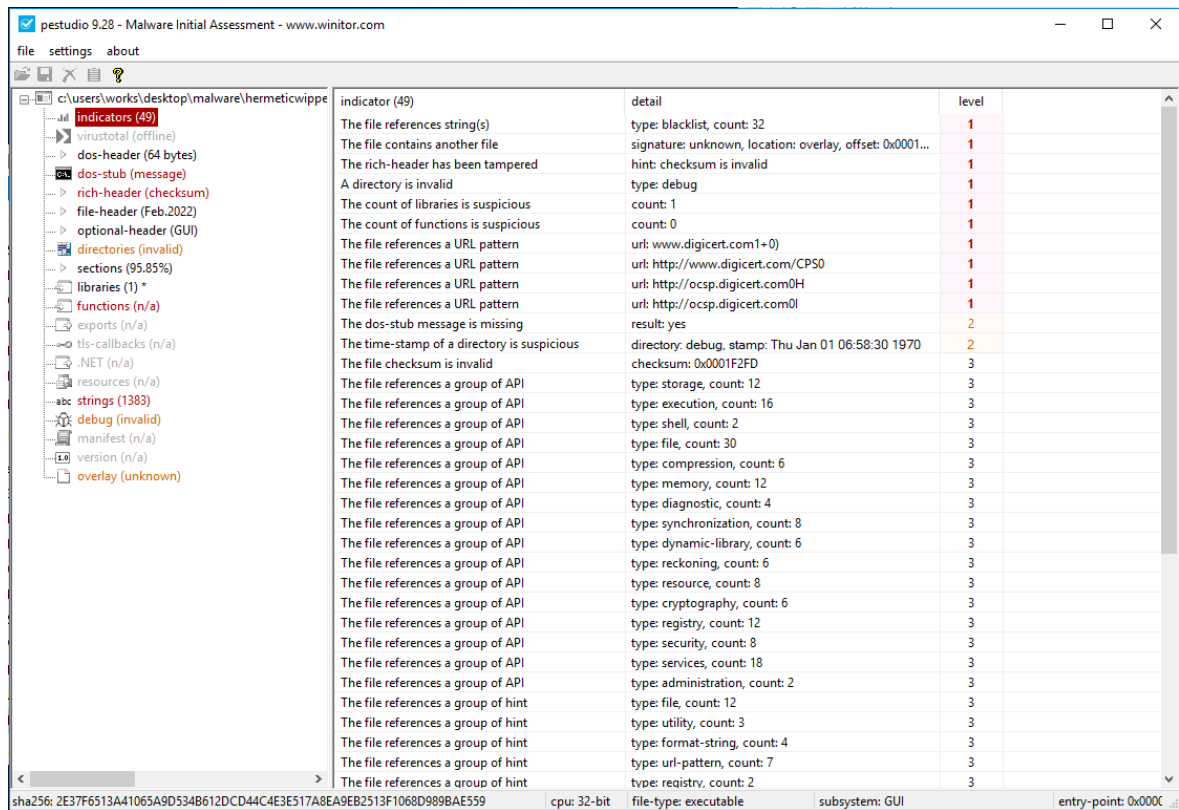
Obrázek 24. Nástroj capa – malware HermeticWiper [zdroj vlastní]

5.2.19 PEStudio

PEStudio je nástroj pro analýzu spustitelných souborů PE. Je dostupný pro MS Windows a je vydán ve dvou verzích. První verze Free je volně ke stažení a poskytuje základní moduly jako je např. zjištění informací o souboru, včetně digitálních podpisů, nalezení pevně zakódovaných adres IP adres a URL, dokáže vyhledat a zobrazit metadata, importy, exporty, řetězce a další. [41]

Druhá verze PEStudia je označená PRO jedná se o placenou verzi, která nabízí navíc možnosti skriptování, automatické počítání hash, zobrazení jmenných prostorů .NET a další. [41]

Na Obrázek 25. Nástroj PEStudio – malware HermeticWiper [zdroj vlastní] je uveden výstup z nástroje PEStudio na vzorku malwaru HermeticWiper.



Obrázek 25. Nástroj PEStudio – malware HermeticWiper [zdroj vlastní]

5.2.20 YARA

YARA³⁰ je open source nástroj, který má pomoci analytikům identifikovat a klasifikovat vzorky malwaru. Umožňuje vytvářet popisy (nebo pravidla – rules) pro rodiny malwaru na základě textových nebo binárních vzorů. YARA je multiplatformní, tj. běží na Linuxu, Windows a Mac OS X. Nástroj YARA lze používat prostřednictvím příkazového řádku nebo ze skriptů Python s rozšířením YARA-Python. [42]

³⁰ Ke stažení na <https://github.com/Yara-Rules/rules>

Vzorek malwaru může obsahovat mnoho jedinečných řetězců nebo binárních identifikátorů. Při klasifikaci malwaru může pomoci identifikovat data, která jsou jedinečná pro daný vzorek malwaru nebo celou rodinu malwaru.

Pravidla YARA vypadají následovně:

```
rules suspicious_strings
{
  strings:
    $a = „PortScanner“
    $b = „Keylogger“
    $c = „malware“

  condition:
    ($a or $b or $c)
}[4]
```

Yara pravidla se skládají z následujících částí:

- Rules Identifier: jedná se o název, který popisuje toto pravidlo. Název může obsahovat libovolný alfanumerický znak a znak podtržítka, ale nesmí obsahovat číslice. Rozlišují se velká a malá písmena a maximální délka je 128 znaků. [4]
- String Definition: jedná se o definici řetězců, které mohou být textové, hexadecimální nebo definované pomocí regulárních výrazů. Pokud pravidlo nespolehá na žádné řetězce, lze tuto sekci vynechat. Každý řetězec má identifikátor, který začíná znakem \$ a následuje posloupnost alfanumerických znaků a podtržitek. Zjednodušeně řečeno, jedná se o proměnné, které jsou použity v následující sekci. [4]
- Condition Section: V této části se definuje logika pravidla. Tato sekce obsahuje booleovský výraz, který určuje podmínku, zda bude pravidlo odpovídat zadanému výrazu. [4]

Když je pravidlo YARA sestavené, můžeme pomocí nástroje YARA skenovat soubory na přítomnost daných pravidel.

```
$ yara -r suspicious.yara samples/
```

Tím dojde k vyhledání daného pravidla v adresáři samples. [4]

Pomocí pravidel YARA lze zadat i složitá pravidla, včetně detekce podpisu certifikátem nebo např. detekci UPX packeru. [4]

5.3 Hexadecimální editory

Hexadecimální editor nebo také binární editor je nástroj pro prohlížení nebo editaci binárních dat. Lze jím prohlížet nebo editovat jakýkoliv binární soubor. Jeho použití vyžaduje vysoké znalosti analýzy.

5.3.1 Hexadecimální editor xxd

Hexadecimální editor xxd je nástroj, který je obsažený ve většině linuxových distribucí. Dokáže vytvořit hexadecimální výpis z daného souboru nebo i ze standardního vstupu. Umožňuje také převést hexadecimální výpis zpět do jeho původní podoby. Pomocí xxd lze také přenášet binární data v reprezentaci ASCII, například pro emailovou komunikaci. [43]

Na Obrázek 26. Hexadecimální editor xxd – malware HermeticWiper [zdroj vlastní] je vidět část výstupu z nástroje xxd na vzorku malwaru HermeticWiper.

```

00000000: 4d5a 9000 0300 0000 0400 0000 ffff 0000  MZ.....
00000010: b800 0000 0000 0000 4000 0000 0000 0000  .....@.....
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000030: 0000 0000 0000 0000 0000 0000 e000 0000  .....e000 0000
00000040: 0e1f ba0e 00b4 09cd 21b8 014c cd21 5468  .....!..L.!Th
00000050: 6973 2069 7320 7468 6520 636f 6f6c 6573  is is the cool
00000060: 7420 6d61 6c77 6172 6520 6576 6572 7920  t malware every
00000070: 6275 7420 6e6f 7420 2400 0000 0000 0000  but not $.
00000080: 0e2b a746 4a4a c915 4a4a c915 4a4a c915  .+.FJJ..JJ..JJ..
00000090: 4332 5c15 4e4a c915 4332 4a15 4f4a c915  C2\..NJ..C2J.OJ..
000000a0: 4332 5a15 454a c915 4a4a c815 294a c915  C2Z.EJ..JJ..)J..
000000b0: a12e c014 454a c915 a12e 3615 4b4a c915  ....EJ....6.KJ..
000000c0: 4a4a 5e15 4b4a c915 a12e cb14 4b4a c915  JJ^.KJ.....KJ..
000000d0: 5269 6368 4a4a c915 0000 0000 0000 0000  RichJJ.....
000000e0: 5045 0000 4c01 0500 0503 1662 0000 0000  PE..L.....b....
000000f0: 0000 0000 e000 0201 0b01 0e0c 0040 0000  .....@...
00000100: 0078 0100 0000 0000 803b 0000 0010 0000  .x.....;.....
00000110: 0050 0000 0000 4000 0010 0000 0002 0000  .P....@.....
00000120: 0500 0100 0000 0000 0500 0100 0000 0000  .....
00000130: 00f0 0100 0004 0000 fdf2 0100 0200 4081  .....@...
00000140: 0000 1000 0010 0000 0000 1000 0010 0000  .....
00000150: 0000 0000 1000 0000 0000 0000 0000 0000  .....
00000160: 5c5a 0000 a000 0000 0080 0000 805a 0100  \Z.....Z..
00000170: 0000 0000 0000 0000 00ba 0100 080f 0000  .....
00000180: 00e0 0100 9803 0000 c058 0000 3800 0000  .....X..8...
00000190: 0000 0000 0000 0000 0000 0000 0000 0000  .....
000001a0: 0000 0000 0000 0000 f858 0000 4000 0000  .....X..@...
000001b0: 0000 0000 0000 0000 0050 0000 9001 0000  .....P.....
000001c0: 0000 0000 0000 0000 0000 0000 0000 0000  .....
000001d0: 0000 0000 0000 0000 2e74 6578 7400 0000  .....text...
000001e0: eb3f 0000 0010 0000 0040 0000 0004 0000  .?.....@.....
000001f0: 0000 0000 0000 0000 0000 0000 2000 0060  .....
00000200: 2e72 6461 7461 0000 fa12 0000 0050 0000  .rdata.....P..
00000210: 0014 0000 0044 0000 0000 0000 0000 0000  .....D.....
00000220: 0000 0000 4000 0040 2e64 6174 6100 0000  ....@..@.data...
00000230: 8403 0000 0070 0000 0002 0000 0058 0000  .....p.....X..
00000240: 0000 0000 0000 0000 0000 0000 4000 00c0  .....@...

```

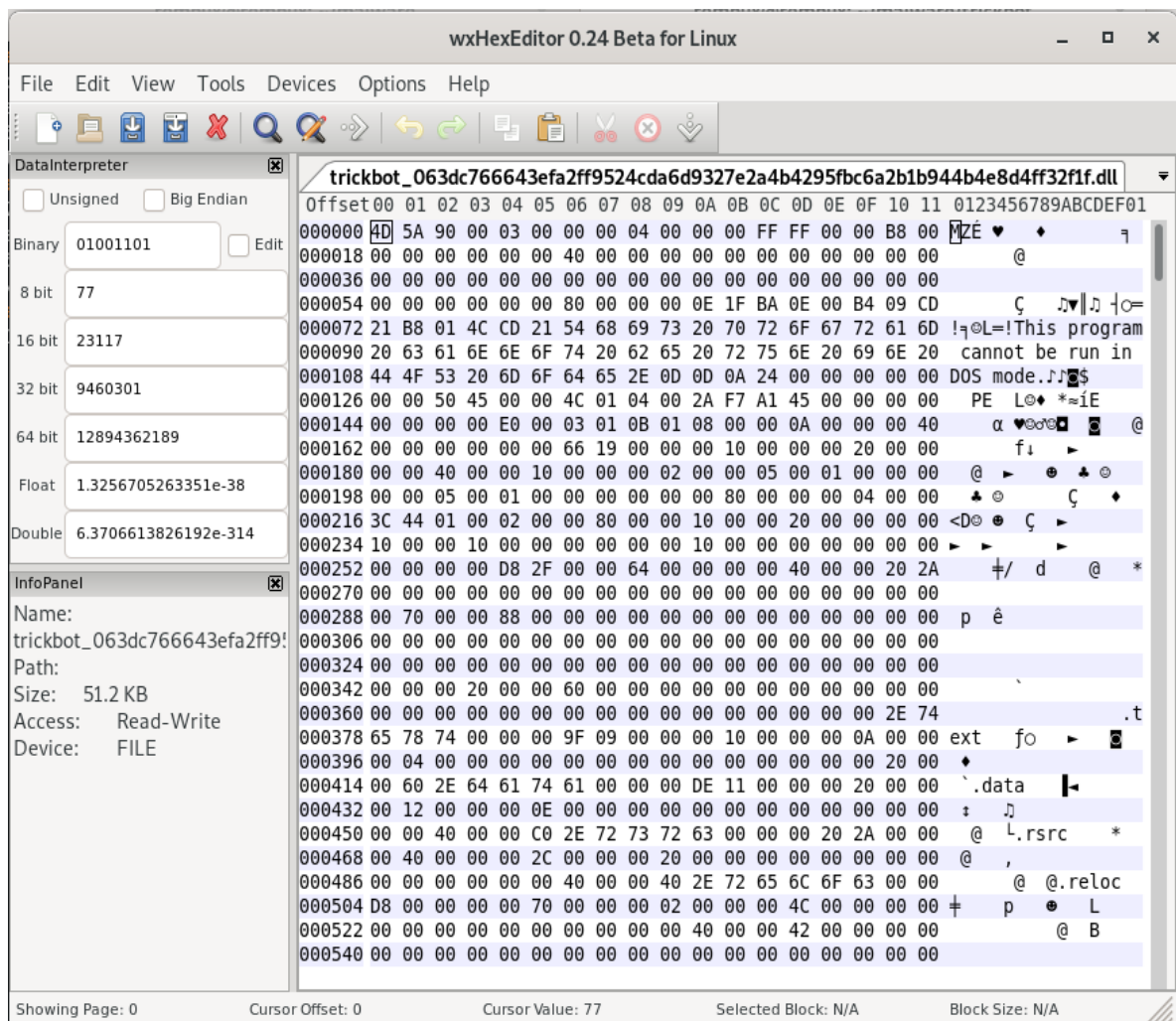
Obrázek 26. Hexadecimální editor xxd – malware HermeticWiper [zdroj vlastní]

5.3.2 Hexadecimální editor wxHexEditor

Hexadecimální editor wxHexEditor³¹ je původně určený pro platformu Linux. V dnešní době je dostupný i pro macOS a Windows. Mezi jeho výhody patří malé vytížení paměti RAM, nevytváří žádné dočasné soubory a podporuje soubory až do velikosti 2 EB (ExaByte). [44]

Na níže uvedeném obrázku je vidět část výstupu z nástroje wxHexEditor na vzorku malwaru Trickbot.

³¹ Ke stažení na <https://sourceforge.net/projects/wxhexeditor/>



Obrázek 27. Hexadecimální editor wxHexEditor – malware trickbot [zdroj vlastní]

5.4 Nástroje pro dynamickou analýzu vzorku malwaru

Nástroje pro dynamickou analýzu jsou určeny pro monitorování chování spuštěného vzorku malwaru. Při dynamické analýze je nutné dbát velké opatrnosti a dynamickou analýzu provádět v bezpečném prostředí.

5.4.1 Síťové nástroje

Tyto nástroje monitorují síťový provoz, který odchází z operačního systému, ve kterém byl spuštěn vzorek malwaru anebo odpovídají na síťové dotazy infikovaného operačního systému.

5.4.1.1 *Nástroj tcpdump*

Nástroj tcpdump³² je dostupný pod BSD licenci a je určený pro GNU Linux a další Unixové systémy. Pro platformu MS Windows je tcpdump dostupný pod jménem WinDump³³ a je plně kompatibilní s tcpdump. Tento nástroj umožňuje zachytávat a analyzovat síťový provoz v příkazové řádce. Nástroj tcpdump vypisuje popis síťového paketu včetně časového razítka. Umí ukládat síťový provoz do souboru, automaticky překládat IP adresy na doménová jména, analyzovat data z uloženého souboru, aplikovat různé filtry na zdrojové nebo cílové adresy nebo porty. Jedná se o mocný nástroj pro zachytávání síťového provozu. Tím, že je určený pro příkazovou řádku, lze jej pro usnadnění práce provozovat pomocí různých skriptů. [45]

³² Ke stažení na <https://www.tcpdump.org>

³³ Ke stažení na <https://www.winpcap.org/windump>

Obrázek 28. Nástroj tcpdump [zdroj vlastní] ukazuje výstup zachytávání síťového provozu nástrojem tcpdump, při běžném procházení webových stránek.

```
12:29:21.554323 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 1, win 64240, length 0
12:29:21.557343 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [P.], seq 1:518, ack 1, win 64240, length 517
12:29:21.557702 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], ack 518, win 65535, length 0
12:29:21.587669 IP dns-eu-fra2.nordvpn.com.domain > remnux.51023: 55045 1/0/1 PTR ip108.ip-146-59-30.eu. (90)
12:29:21.617061 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], seq 1:2921, ack 518, win 65535, length 2920
12:29:21.617082 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 2921, win 62780, length 0
12:29:21.617381 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 2921:3380, ack 518, win 65535, length 459
12:29:21.617390 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 3380, win 62321, length 0
12:29:21.622348 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [P.], seq 518:644, ack 3380, win 62780, length 126
12:29:21.622681 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], ack 644, win 65535, length 0
12:29:21.753659 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 3380:3622, ack 644, win 65535, length 242
12:29:21.753680 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 3622, win 62780, length 0
12:29:21.774170 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [P.], seq 644:821, ack 3622, win 62780, length 177
12:29:21.774667 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], ack 821, win 65535, length 0
12:29:21.774836 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [P.], seq 821:1166, ack 3622, win 62780, length 345
12:29:21.775191 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], ack 1166, win 65535, length 0
12:29:21.845746 IP remnux.57394 > ns31479445.ip-141-95-47.eu.https: Flags [P.], seq 1501:1594, ack 33022, win 62780, length 93
12:29:21.846187 IP ns31479445.ip-141-95-47.eu.https > remnux.57394: Flags [.], ack 1594, win 65535, length 0
12:29:21.900800 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 3622:3666, ack 1166, win 65535, length 44
12:29:21.900865 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 3666, win 62780, length 0
12:29:21.901082 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [P.], seq 1166:1204, ack 3666, win 62780, length 38
12:29:21.901368 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [.], ack 1204, win 65535, length 0
12:29:21.903389 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 3666:5114, ack 1204, win 65535, length 1448
12:29:21.903584 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 5114:6562, ack 1204, win 65535, length 1448
12:29:21.903594 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 6562, win 62780, length 0
12:29:21.903715 IP ip108.ip-146-59-30.eu.https > remnux.39696: Flags [P.], seq 6562:6774, ack 1204, win 65535, length 212
12:29:21.952679 IP remnux.39696 > ip108.ip-146-59-30.eu.https: Flags [.], ack 6774, win 62780, length 0
12:29:21.988171 IP ns31479445.ip-141-95-47.eu.https > remnux.57394: Flags [P.], seq 33022:35974, ack 1594, win 65535, length 2952
12:29:21.988200 IP remnux.57394 > ns31479445.ip-141-95-47.eu.https: Flags [.], ack 35974, win 62780, length 0
12:29:22.006221 IP ns31479445.ip-141-95-47.eu.https > remnux.57394: Flags [P.], seq 35974:41068, ack 1594, win 65535, length 5094
12:29:22.006244 IP remnux.57394 > ns31479445.ip-141-95-47.eu.https: Flags [.], ack 41068, win 61320, length 0
12:29:22.224666 IP remnux.39100 > lb.sdn.cz.https: Flags [P.], seq 9029:9131, ack 1439441, win 65535, length 102
12:29:22.225092 IP lb.sdn.cz.https > remnux.39100: Flags [.], ack 9131, win 65535, length 0
12:29:22.229055 IP remnux.39100 > lb.sdn.cz.https: Flags [P.], seq 9131:9238, ack 1439441, win 65535, length 107
12:29:22.229447 IP lb.sdn.cz.https > remnux.39100: Flags [.], ack 9238, win 65535, length 0
12:29:22.323739 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1439441:1441559, ack 9238, win 65535, length 2118
12:29:22.323739 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1441559:1442818, ack 9238, win 65535, length 1259
12:29:22.323770 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1441559, win 65535, length 0
12:29:22.323803 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1442818, win 65535, length 0
12:29:22.329586 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1442818:1445274, ack 9238, win 65535, length 2456
12:29:22.329657 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1445274, win 65535, length 0
12:29:22.329752 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1445274:1446502, ack 9238, win 65535, length 1228
12:29:22.329758 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1446502, win 65535, length 0
12:29:22.330146 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1446502:1448958, ack 9238, win 65535, length 2456
12:29:22.330154 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1448958, win 65535, length 0
12:29:22.330264 IP lb.sdn.cz.https > remnux.39100: Flags [P.], seq 1448958:1450186, ack 9238, win 65535, length 1228
12:29:22.330270 IP remnux.39100 > lb.sdn.cz.https: Flags [.], ack 1450186, win 65535, length 0
```

Obrázek 28. Nástroj tcpdump [zdroj vlastní]

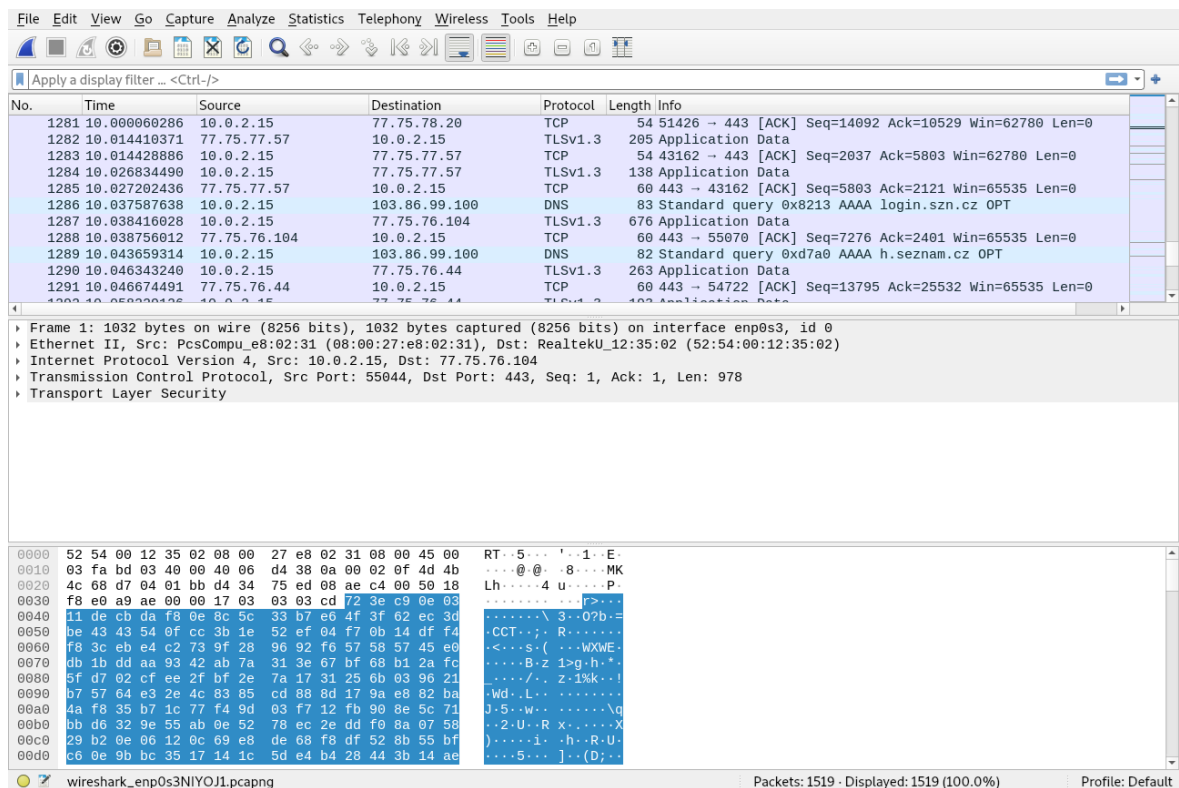
5.4.1.2 Nástroj Wireshark

Wireshark³⁴, dříve Ethertcap, je multiplatformní nástroj pro MS Windows, Linux, macOS, rodinu BSD a další. Wireshark umí zachytávat síťový provoz a ten analyzovat. Wireshark umožňuje hloubkovou inspekci různých protokolů. Umí pracovat s online provozem i analyzovat již dříve zachycený síťový tok pomocí i jiných nástrojů jako např. tcpdump a podobně. Wireshark má vestavěny pokročilé filtry a podporuje dešifrování mnoha protokolů, jako IPsec, Kerberos, ISAKMP, SSL/TLS, WPA a dalších. [46]

Jeho možnosti jsou velmi obsáhlé a tento nástroj lze považovat za jeden z nejlepších analyzátorů síťového provozu.

³⁴ Ke stažení na <https://www.wireshark.org>

Obrázek 29. Nástroj Wireshark [zdroj vlastní] ukazuje prostředí nástroje Wireshark při zachytávání běžného webového provozu.



Obrázek 29. Nástroj Wireshark [zdroj vlastní]

5.4.1.3 Nástroj INetSim

INetSim³⁵ je softwarový nástroj pro simulaci běžných internetových služeb ve forenzním prostředí a pro analýzu síťového chování neznámých vzorků malwaru.

V současné době jsou součástí distribuce INetSim moduly pro simulaci služeb HTTP/HTTPS, kdy podporuje metody GET, HEAD, POST a OPTIONS s HTTP/1.0 a HTTP/1.1. Dále poskytuje dva režimy. První, reálný režim, poskytuje existující soubory z adresáře webroot, druhý, falešný (fake) režim, poskytuje falešné soubory v požadavku HTTP (např. .html nebo .exe). Dále vyřizuje požadavky vzorku malwaru na checkip.dyndns.org, které jsou vyřizeny IP adresou INetSim. [47]

³⁵ Ke stažení na <https://www.inetsim.org/downloads.html>

Další modul poskytuje podporu SMTP/SMTSPS, kdy dokáže vyřizovat e-maily, které jsou uloženy ve formátu mbox. Také podporuje ESMTP, POP3 / POP3S u kterých umí dynamicky vytvářet obsahu poštovní schránky z dodaných souborů mbox. [47]

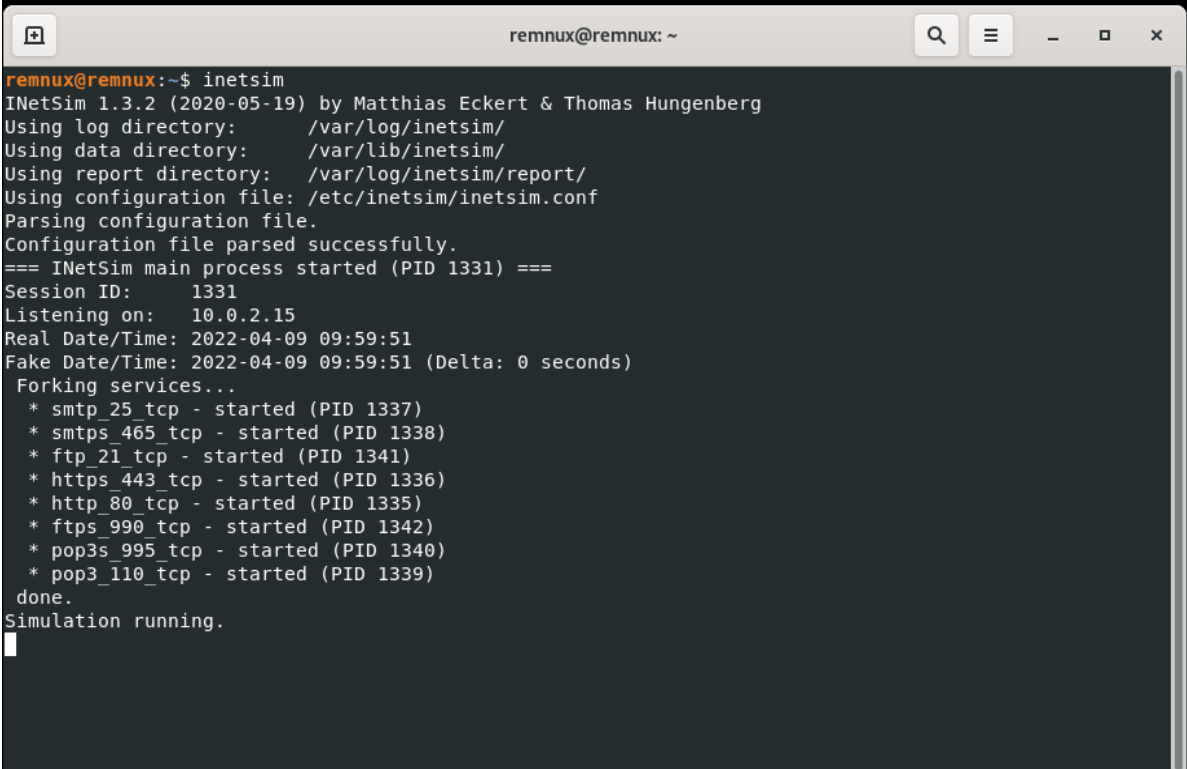
Jako další modul je modul DNS, které umožňuje dopředné i zpětné vyhledávání. [47]

Modul FTP/FTPS/TFTP poskytuje stažení i nahrávání souborů, kdy vytváří virtuální souborový systém založený na existující struktuře adresářů, které umožňuje vytváření a mazání libovolných souborů. [47]

Mezi další moduly, které lze využít patří IRC, NTP, Syslog a další. [47]

Jedná se opravdu o univerzální a všestranný nástroj pro síťovou dynamickou analýzu.

Na následujícím obrázku je vidět spuštění služeb nástroje INetSim.



```
remnux@remnux:~$ inetsim
INetSim 1.3.2 (2020-05-19) by Matthias Eckert & Thomas Hungenberg
Using log directory: /var/log/inetsim/
Using data directory: /var/lib/inetsim/
Using report directory: /var/log/inetsim/report/
Using configuration file: /etc/inetsim/inetsim.conf
Parsing configuration file.
Configuration file parsed successfully.
=== INetSim main process started (PID 1331) ===
Session ID: 1331
Listening on: 10.0.2.15
Real Date/Time: 2022-04-09 09:59:51
Fake Date/Time: 2022-04-09 09:59:51 (Delta: 0 seconds)
Forking services...
 * smtp_25_tcp - started (PID 1337)
 * smtps_465_tcp - started (PID 1338)
 * ftp_21_tcp - started (PID 1341)
 * https_443_tcp - started (PID 1336)
 * http_80_tcp - started (PID 1335)
 * ftps_990_tcp - started (PID 1342)
 * pop3s_995_tcp - started (PID 1340)
 * pop3_110_tcp - started (PID 1339)
done.
Simulation running.
```

Obrázek 30. Nástroj INetSim [zdroj vlastní]

5.4.1.4 *Nástroj accept-all-ips*

Accept-all-ips³⁶ je open source nástroj, který po spuštění přijme jakékoliv připojení ke všem adresám IPv4 a IPv6 a přesměruje je na nastavený odpovídající místní port. Tento nástroj je vhodný pro zachycení a přesměrování jakéhokoliv provozu. Využívá se v kombinaci s nástrojem INetSim, při analýze síťového provozu. Pro správnou funkci nástroje accept-all-ips je nutné na monitorovaném operačním systému nastavit výchozí síťovou bránu na IP adresu stroje, kde běží accept-all-ips. [48]

5.4.2 **Nástroje pro monitoring spuštěného malwaru**

Tyto nástroje jsou určeny pro monitoring operačního systému, ve kterém běží vzorek malwaru.

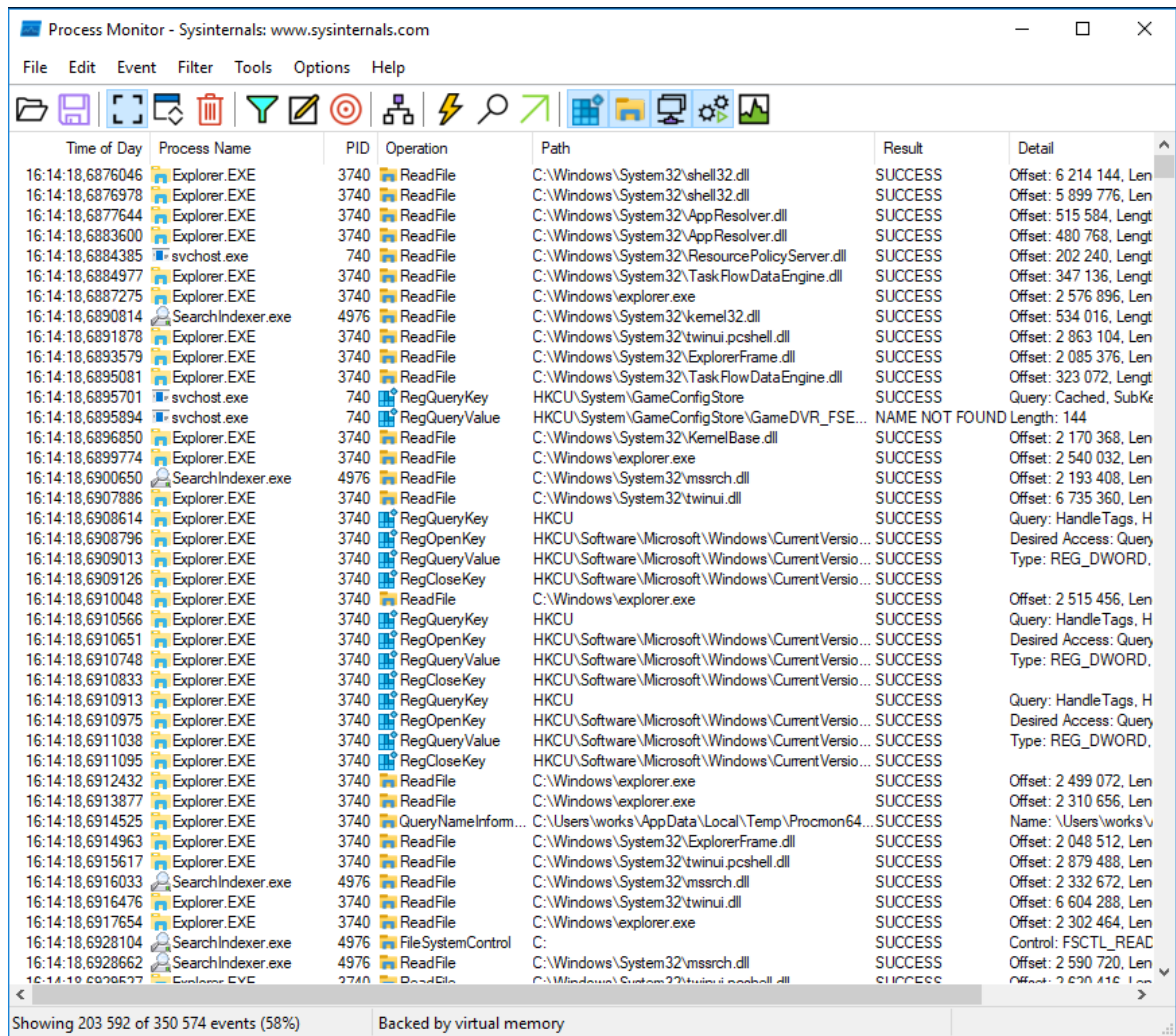
5.4.2.1 *Nástroj Process Monitor*

Process Monitor³⁷ (zkráceně procmon) je pokročilý monitorovací nástroj pro operační systém MS Windows, který v reálném čase monitoruje souborový systém, registry Windows a aktivitu procesu. Umožňuje filtrování podle ID relace, uživatelského jména apod. Získané informace lze protokolovat do různých typů souborů. Jedná se o důležitý základní nástroj pro dynamickou analýzu malwaru a je vyvíjen společností Microsoft. [49]

³⁶ Ke stažení na <https://github.com/REMnux/distro/blob/master/files/accept-all-ips>

³⁷ Ke stažení na <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>

Na Obrázek 31. Nástroj Process Monitor [zdroj vlastní] je vidět prostředí nástroje Process Monitor, při monitorování běhu operačního systému.



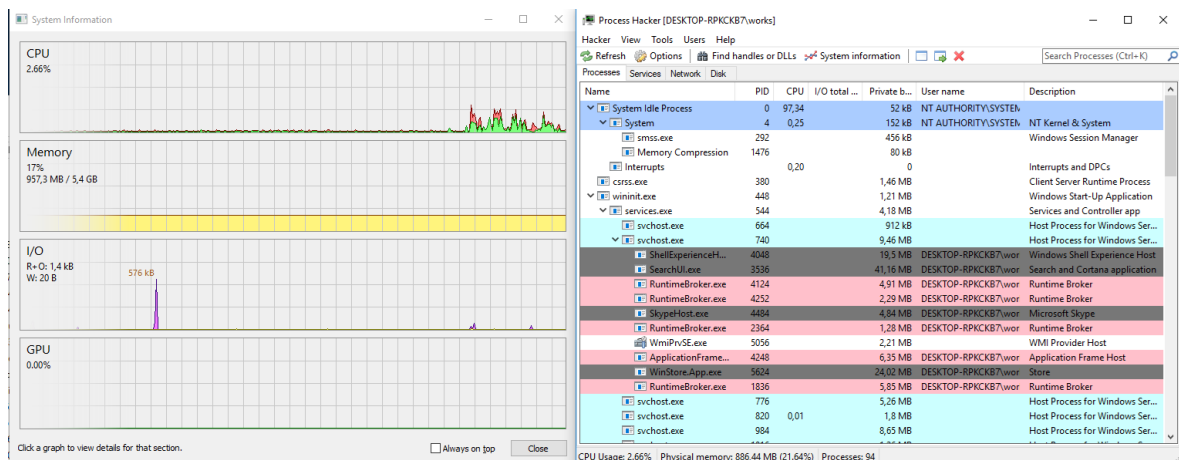
Obrázek 31. Nástroj Process Monitor [zdroj vlastní]

5.4.2.2 Nástroj Process Hacker

Process Hacker³⁸ je víceúčelový nástroj, pomocí kterého lze sledovat systémové prostředky jako jsou spuštěné procesy, přehled služeb, síťových spojení a přehled přístupů k pevnému

³⁸ Ke stažení na <https://github.com/processhacker/processhacker/>

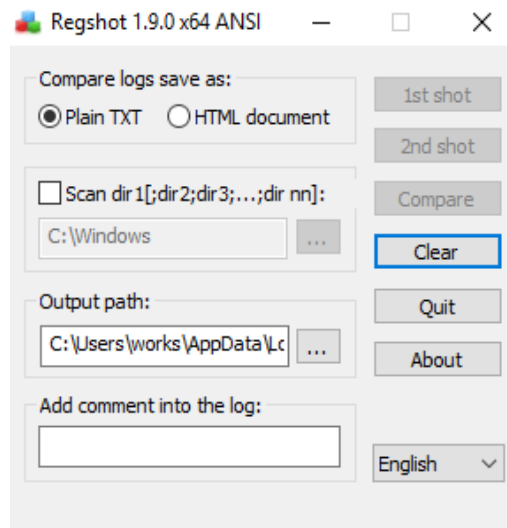
disku. Process Hacker je vydaný pod licencí GLP v3 a nevyžaduje instalaci. Jedná se o důležitý základní nástroj pro dynamickou analýzu vzorku malwaru. [50]



Obrázek 32. Nástroj Process Hacker [zdroj vlastní]

5.4.2.3 Nástroj RegShot

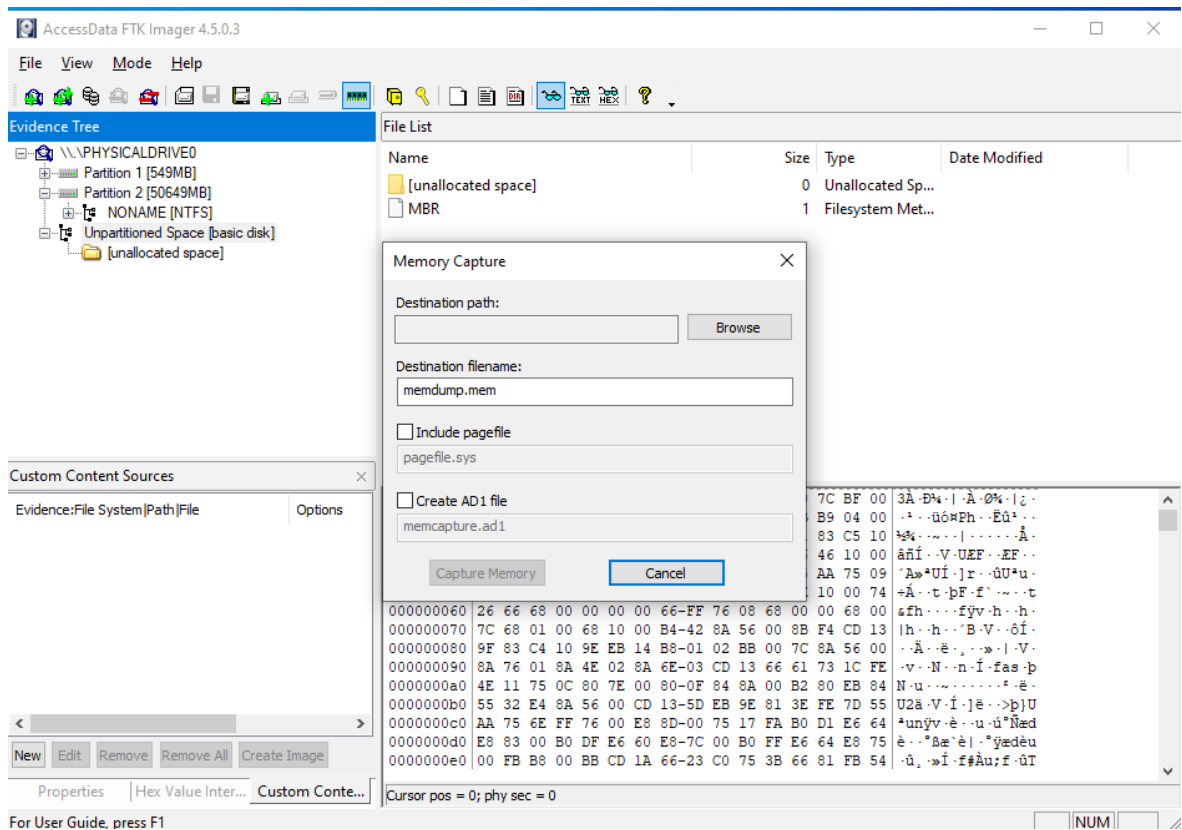
RegShot³⁹ je nástroj vydaný pod licencí LGPL, který umožňuje vytvořit snímek registrů MS Windows před spuštěním vzorku malwaru a po jeho spuštění. Následně automaticky porovná tyto snímky a zobrazí změny registru. Tím lze zjistit všechny zápisy malwaru do registru. [51]



Obrázek 33. Nástroj RegShot [zdroj vlastní]

³⁹ Ke stažení na <https://github.com/Seabreg/Regshot>

patří možnost vytvořit výpis obsahu paměti RAM (dump), viz. Obrázek 35. Nástroj FTK Imager [zdroj vlastní] [53]



Obrázek 35. Nástroj FTK Imager [zdroj vlastní]

5.6 Nástroj pro analýzu výpisu paměti RAM volatily3

Framework volatility3 je nástroj pro forenzní analýzu výpisu paměti RAM. Podporuje výpisy paměti z MS Windows, Linux a macOS v různých formátech. Podporuje mnoho modulů, pomocí nich lze z výpisu obsahu paměti získat např. seznam běžících procesů, seznam otevřených síťových spojení a další. Pomocí modulu malfind dokáže framework volatility3 z výpisu obsahu paměti získat seznam procesů, které mohou být potencionálně infikované injektovaným kódem. Jeho možnosti jsou velmi obsáhlé. [54]

Na níže uvedeném obrázku je vidět výpis procesů ze zachyceného výpisu obsahu paměti v nástroji Volatility3.

```
remnux@remnux: /media/sf.VirtualBoxShare02$ vol3 -f memdump_sample01-longdata.mem windows.pslist
Volatility 3 Framework 2.0.3
Progress: 100.00
PDB scanning finished
PID      PPID      ImageFileName      Offset(V)      Threads Handles SessionId      Wow64      CreateTime      ExitTime      File output
4        0         System             0xe18568c4f340 97             -           N/A        False 2022-04-25 14:18:04.000000      N/A        Disabled
272     4         smss.exe           0xe1856a306040 4              -           N/A        False 2022-04-25 14:18:04.000000      N/A        Disabled
360     352      csrss.exe          0xe1856a2dc5c0 11             -           0          False 2022-04-25 14:18:10.000000      N/A        Disabled
432     272      smss.exe           0xe1856c798080 0              -           1          False 2022-04-25 14:18:11.000000      2022-04-25 14:18:11.000000      Disabled
440     352      wininit.exe       0xe1856c5fd080 5              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
452     432      csrss.exe          0xe1856a2c85c0 11             -           1          False 2022-04-25 14:18:11.000000      N/A        Disabled
512     432      winlogon.exe      0xe18569dad080 6              -           1          False 2022-04-25 14:18:11.000000      N/A        Disabled
552     440      services.exe      0xe1856c45e080 22             -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
560     440      lsass.exe         0xe1856a2cc480 9              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
660     512      fontdrvhost.exe  0xe1856c7f85c0 6              -           1          False 2022-04-25 14:18:11.000000      N/A        Disabled
668     440      fontdrvhost.exe  0xe1856ca755c0 6              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
676     552      svchost.exe       0xe1856ca785c0 2              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
740     552      svchost.exe       0xe1856ca955c0 33             -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
772     552      svchost.exe       0xe1856cafe5c0 18             -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
820     552      svchost.exe       0xe1856cb19400 5              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
892     512      dmw.exe           0xe1856cb5f5c0 12             -           1          False 2022-04-25 14:18:11.000000      N/A        Disabled
960     552      svchost.exe       0xe1856cb92340 18             -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
980     552      svchost.exe       0xe1856cb975c0 5              -           0          False 2022-04-25 14:18:11.000000      N/A        Disabled
1008    552      svchost.exe       0xe1856cbb05c0 4              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1016    552      svchost.exe       0xe1856cbb45c0 4              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
456     552      svchost.exe       0xe1856cb01080 11             -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
328     552      svchost.exe       0xe1856c8fb080 4              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
752     552      svchost.exe       0xe18568d47880 9              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
844     552      svchost.exe       0xe18568d3e080 6              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1052    552      svchost.exe       0xe18568cc0080 11             -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1116    552      VBoxService.exe  0xe1856ca285c0 12             -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1128    552      svchost.exe       0xe1856ca375c0 14             -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1176    552      svchost.exe       0xe1856cc155c0 6              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1192    552      svchost.exe       0xe1856cc235c0 9              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1200    552      svchost.exe       0xe1856cc275c0 8              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1208    552      svchost.exe       0xe1856cc2b5c0 4              -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
1280    552      svchost.exe       0xe1856cc455c0 17             -           0          False 2022-04-25 14:18:12.000000      N/A        Disabled
```

Obrázek 36. Framework Volatility3 modul pslist [zdroj vlastní]

5.7 Disassemblery/Debuggery

Autoři malware programují malware většinou v nějakém vysokoúrovňovém programovacím jazyce jako je C nebo C++, který je poté zkompilován do spustitelného souboru pomocí kompilátoru. Proto je potřeba převést zkompilovaný binární soubor do nízkoúrovňového Jazyka Symbolických Adres (zkratka JSA), pro lepší čitelnost. [4]

„Debugger je program, který umožní zastavit běh programu po každém vykonání jednoho řádku programu nebo na předem definované záračce (breakpoint). Pokud je laděný program pozastaven, lze si prohlížet či měnit obsah paměti či konkrétních proměnných, stav zásobníku, registrů a podobně. Tímto způsobem kontrolujeme, zda program skutečně dělá to, co jsme zamýšleli.“ [55]

5.7.1 Nástroj GDB

GDB⁴² je GNU Project debugger/disassembler, který je součástí většiny distribucí Linuxu a pracuje pouze v textovém režimu. Umožňuje mimo jiné zastavení běhu programu nastavením bodu přerušení (breakpoint), umožňuje zobrazení a změnu aktuálních hodnot registrů a podobně. Aktuálně podporuje programovací jazyky jako Ada, Assembly, C, C++, D, Fortran, Go, Objective-C, OpenCL, Modula-2, Pascal a Rust. [56]

Obrázek 37. Disassembler GDB [zdroj vlastní]ukazuje prostředí nástroje gdb.

```
(gdb) disas main
Dump of assembler code for function main:
   0x0000000000000113c <+0>:      endbr64
   0x00000000000001140 <+4>:      push   %rbp
   0x00000000000001141 <+5>:      mov    %rsp,%rbp
   0x00000000000001144 <+8>:      sub    $0x20,%rsp
   0x00000000000001148 <+12>:     mov    %edi,-0x14(%rbp)
   0x0000000000000114b <+15>:     mov    %rsi,-0x20(%rbp)
   0x0000000000000114f <+19>:     movl   $0x1,-0x8(%rbp)
   0x00000000000001156 <+26>:     jmp    0x1170 <main+52>
   0x00000000000001158 <+28>:     movl   $0x1,-0x4(%rbp)
   0x0000000000000115f <+35>:     mov    -0x4(%rbp),%eax
   0x00000000000001162 <+38>:     mov    %eax,%edi
   0x00000000000001164 <+40>:     callq 0x1129 <inc>
   0x00000000000001169 <+45>:     mov    %eax,-0x4(%rbp)
   0x0000000000000116c <+48>:     addl   $0x1,-0x8(%rbp)
   0x00000000000001170 <+52>:     cmpl   $0x4,-0x8(%rbp)
   0x00000000000001174 <+56>:     jle    0x1158 <main+28>
   0x00000000000001176 <+58>:     mov    -0x14(%rbp),%eax
   0x00000000000001179 <+61>:     leaveq
   0x0000000000000117a <+62>:     retq
End of assembler dump.
(gdb) █
```

Obrázek 37. Disassembler GDB [zdroj vlastní]

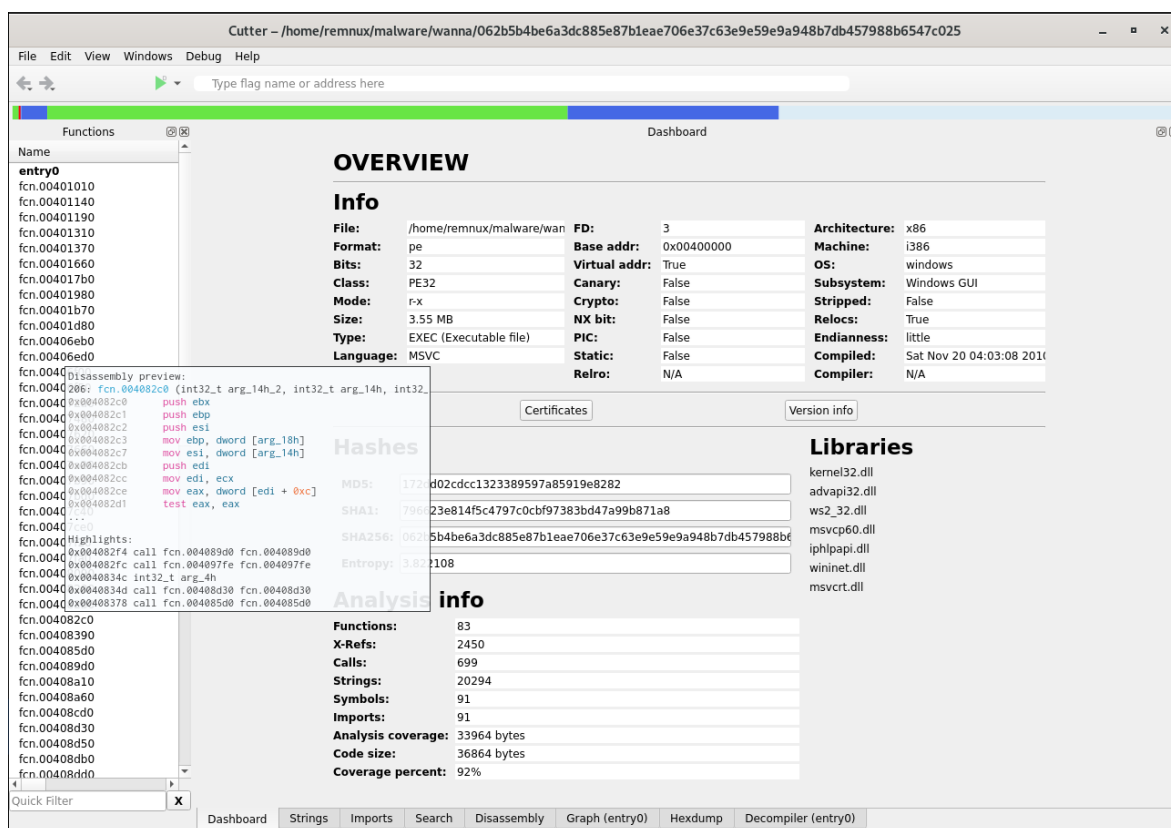
⁴² Ke stažení na <https://www.sourceware.org/gdb/>

5.7.2 Nástroj cutter

Cutter⁴³ je open source nástroj pro reverzní inženýrství a poskytuje velké množství různých widgetů a funkcí. Je dostupný pro platformy MS Windows, Linux a macOS. Cutter poskytuje pluginy pro několik dekompilátorů jako je Ghidra, RetDec, JSDec. [57]

Obsahuje několik záložek, které ukazují různé důležité informace. Dashboard ukazuje základní informace o souboru jako je typ souboru, importované knihovny, certifikáty a další informace. V záložce strings zobrazuje nástroj cutter řetězce v různých typech kódování. V dalších záložkách je možnost vyhledávání, disasemblovaný kód, včetně grafického zobrazení funkcí. Dále je možné nastavit breakpointy a spustit debugování kódu.

Na dalším obrázku je vidět prostředí nástroje cutter. Jako vzorek malwaru byl použitý malware WannaCry.



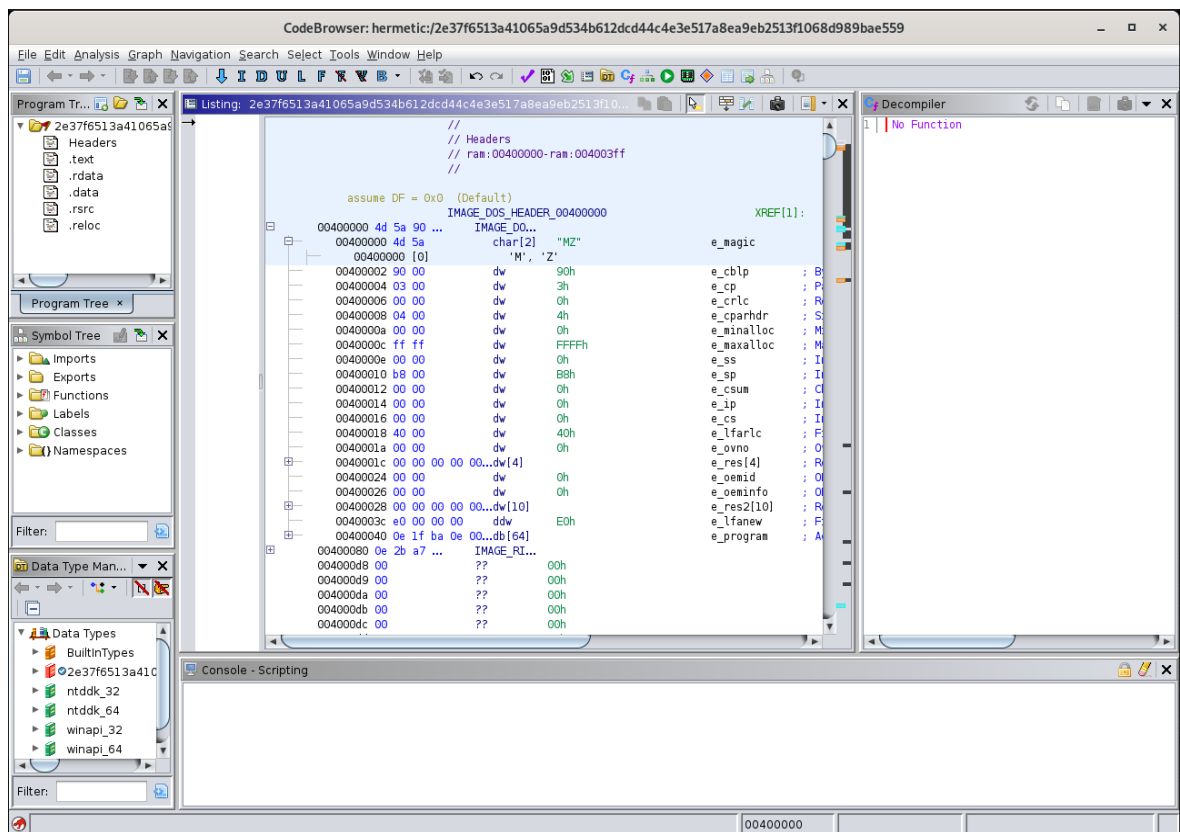
Obrázek 38. Disassembler cutter [zdroj vlastní]

⁴³ Ke stažení na <https://cutter.re/>

5.7.3 Nástroj Ghidra

Ghidra⁴⁴ je jedním z mnoha projektů open source softwaru vyvinutých a spravovaných v rámci Národní bezpečnostní agentury NSA. Obsahuje analytické nástroje, které umožňují analyzovat zkompileovaný kód na různé platformy včetně Windows, macOS a Linux. Schopnosti zahrnují disassembler, assembler, lze vytvářet grafy a podporuje skriptování. Ghidra podporuje širokou škálu instrukčních sad procesorů a spustitelných formátů a lze je spustit ve dvou režimech. Interaktivní a automatizovaný režim. [58]

Na dalším obrázku je vidět prostředí nástroje Ghidra. Jako vzorek malwaru byl použitý malware HermeticWiper.



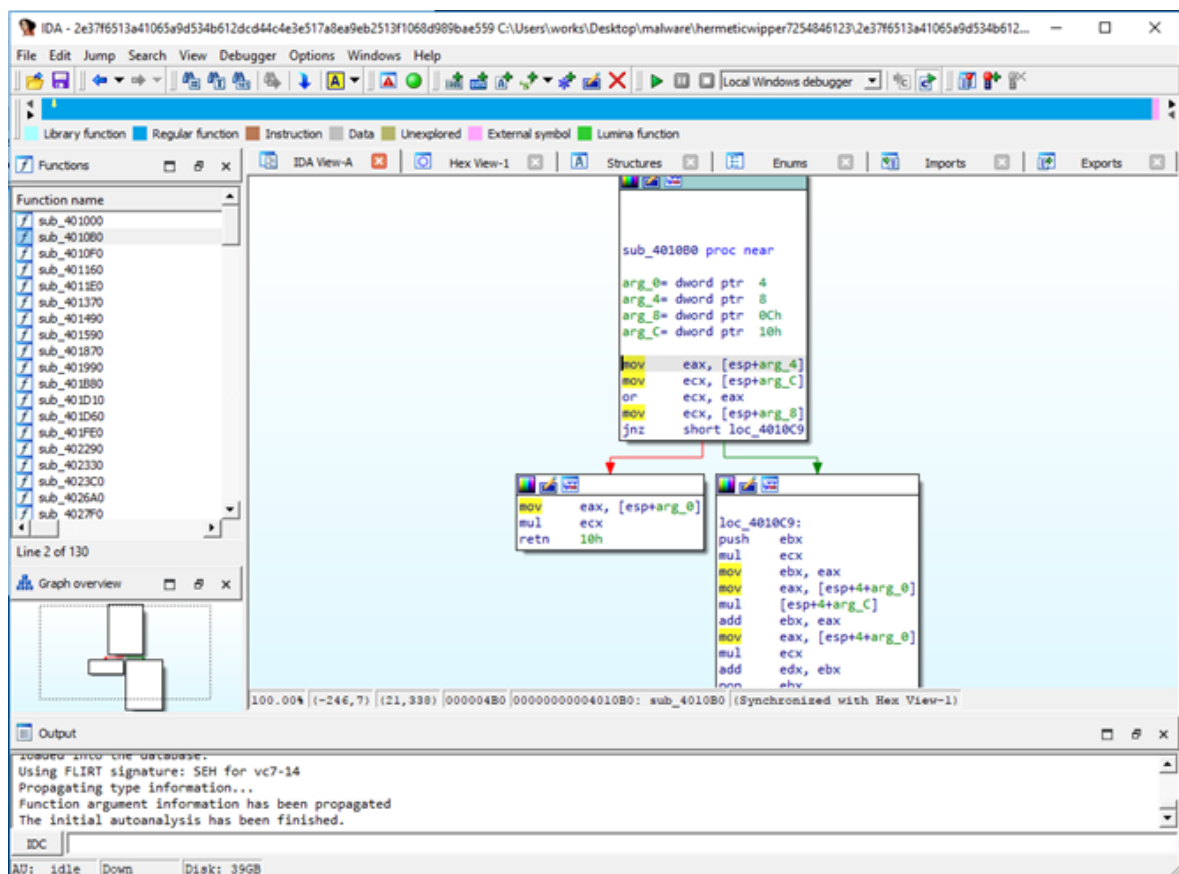
Obrázek 39. Disassembler Ghidra [zdroj vlastní]

⁴⁴ Dostupný ke stažení na <https://github.com/NationalSecurityAgency/ghidra>

5.7.4 Nástroj IDA

IDA⁴⁵ patří mezi nejvýkonnější a nejoblíbenější komerční disassembler/debugger. IDA může běžet na různých platformách (Windows, Linux a macOS) a podporuje analýzu různých formátů souborů, včetně formátů PE/ELF/Macho-O. Kromě komerční verze IDA Pro a IDA home, je IDA distribuován ve verzi Freeware. Tato verze má řadu omezení a je pouze pro nekomerční použití. Umožňuje disasemblovat binární 32bitový i 64bitový systém Windows. Dalším omezením je, že nelze uložit rozpracovaný projekt a postrádá podporu IDA-Pythonu. [59]

Obrázek 40. Disassembler IDA Free [zdroj vlastní]ukazuje prostředí nástroje IDA Free.



Obrázek 40. Disassembler IDA Free [zdroj vlastní]

⁴⁵ Ke stažení na <https://hex-rays.com/ida-free/>

5.7.5 Nástroj ILSpy

Tento multiplatformní nástroj je dekompiler pro C#. Jedná se o konzolovou aplikaci. Nabízí celou řadu možností dekompilace. Tím, že se jedná o open source software, lze rozšiřovat pomocí vlastních pluginů. Jeho použití je velmi jednoduché, spustí se příkazem `ilspycmd` a jako parametr se uvede soubor určený pro dekompilaci. Dekompilovaný soubor se poté zobrazí na standardní výstup. [60]

5.8 Zhodnocení nástrojů pro analýzu malwaru

Nástroje uvedené v této diplomové práci byly všechny vyzkoušeny na různých vzorcích malwaru.

Pro rychlé zjištění informací o typu souboru postačuje využít nástroj `file`, jeho výsledky jsou dostatečné, pro určení spustitelných souborů.

Pro vyhledání vložených řetězců je nejúčelnější nástroj `floss`. Jeho silnou stránkou je, že dokáže vyhledat řetězce vložené nejen v ASCII, ale také v Unicode a umí je automaticky dekódovat. Díky tomu dokáže najít nejvíce vložených řetězců ze všech ostatních nástrojů. Nástroje `strings` a `stringsifter` poskytují shodné výsledky. Výhoda nástroje `stringsifter` je v tom, že řadí vyhledané řetězce tak, aby analytik nemusel prohledávat celý soubor nalezených řetězců, což může být výhodné. Slabou stránkou nástrojů `strings` a `stringsifter` je to, že neumění zobrazit vložené Unicode řetězce.

Dalším nástrojem, který stojí za zmínku je nástroj `Yara-Rules`, který je postaven na signaturách YARA. Díky tomu dokáže určit škodlivé funkce, které obsahují spustitelné soubory. [24]

Mezi nástroje, které jsou nezbytné pro účely analýzy malwaru určitě patří nástroj `ssdeep`, který na základě výpočtu fuzy hash dokáže lehce určit podobnosti částí souborů. To pomáhá rozhodnout, zda analyzovaný soubor je součástí dané rodiny malwaru. [28]

Další nástroj, který je dobré použít při analýze malwaru, je nástroj `PEframe`. Dokáže zobrazí informace z hlavičky PE, včetně rozložení sekcí a podobně. Jeho silnou stránkou je, že umí vypočítat importní hash, tzv. `imphash`, který je jedinečný pro každou rodinu malwaru. Díky tomu lze detekovat rodiny malwaru. [34]

Další z nástrojů, který je nepostradatelný pro analýzu malwaru je nástroj YARA. Jedná se o open source nástroj, který pomáhá analytikům identifikovat a klasifikovat vzorky malwaru. Umožňuje vytvářet popisy (nebo pravidla – rules) pro rodiny malwaru na základě textových nebo binárních vzorů, které jsou jedinečné pro daný vzorek malwaru nebo celou rodinu malwaru. Tím lze jednoduše vzorek malwaru detekovat a identifikovat. Jeho slabinou je to, že musí mít dobře napsaná pravidla, která lze stáhnout i z internetu. Pravidla YARA musí být napsaná na základě dat ze statické analýzy. [42]

Nástroj, který si zaslouží pozornost, je open source nástroj pro analýzu malwaru s názvem capa. Umí detekovat škodlivé funkce, ale hlavně umí rozpoznat chování malwaru. Jeho slabou stránkou je, že nepodporuje všechny programovací jazyky jako např. framework .NET. Z nástrojů pro dynamickou analýzu jsou nezbytné nástroje INetSim a accept-all-ips. INetSim simuluje síťové služby jako např. DNS, webové servery, poštovní servery a podobně. Nástroj accept-all-ips přijme jakékoliv síťové připojení a přesměruje je na odpovídající místní port. Díky těmto dvou nástrojům lze jednoduše připravit a nastavit laboratorní prostředí pro účely dynamické analýzy. [47][48]

Nástroje jako Wireshark, tcpdump/windump, Process Monitor a RegShot jsou další nepostradatelné nástroje pro dynamickou analýzu. Pomocí nich lze monitorovat činnost malwaru a tím realizovat dynamickou analýzu. Při dynamické analýze operační systém i další spuštěné aplikace generují velké množství balastu v logu nástrojů. Pro účely filtrování logů je velkým pomocníkem nástroj ProcDot. Ten dokáže sdružit a vyfiltrovat logy získané z ostatních nástrojů použitých při dynamické analýze a výstup graficky zobrazit, včetně časové osy. Tím je chování malwaru hezky přehledné a lze díky tomu snadno pochopit jeho činnost. Slabou stránkou nástroje ProcDot je to, že nepracuje s žádnou optimalizovanou databází, ale přímo s logy. Díky tomu je práce s objemnějšími logy velmi pomalá.

Z debuggerů/disassemblerů je pomyslný standard nástroj IDA Pro. Jedná se o placený software. Lze využít i verzi Free, která je zdarma, ale možnosti této verze jsou omezené. Z ostatních disassemblerů/debuggerů je velmi dobrý nástroj Ghidra, který poloautomaticky pomáhá analyzovat soubor a umí vizualizovat zdrojový kód zobrazením v grafech. Při práci s nástrojem Ghidra byl nalezen limit 500 objektů vykreslených v grafech, což může při větších projektech působit problémy.

II. PRAKTICKÁ ČÁST

6 ANALÝZA PRVNÍ RODINY MALWARU

Pro účely diplomové práce bylo získáno osm vzorků jedné rodiny malwaru. I když byla známá identita těchto vzorků malwaru, bylo k nim přistupováno jako k neznámým vzorkům malwaru, a proto byly označeny jako „první rodina malwaru“. Tyto soubory byly označeny názvy sample01.exe až sample08.exe. Velikost těchto souborů byla shodná a to 14 KB. Aby bylo vyloučeno, že se jedná o stejný vzorek, byl spočítán hash SHA256 jednotlivých souborů, kdy bylo zjištěno, že se jedná o rozdílné soubory a nic nenasvědčuje tomu, že by se mohlo jednat o stejnou rodinu malwaru.

SHA256 sample01.exe:

00cf9ea8d96ded16db829687c41d0d619868a58ea0ce456f0014168cb553e521

SHA256 sample02.exe:

c9dc59066fe4cd939a7c33572a4db6918e18e1abc76c897abd39b127e7575e18

SHA256 sample03.exe:

6e6daa421b992ea72284f7f955511d063b52f8118f37f7f41877789e1ce7d195

SHA256 sample04.exe:

1cb1bfc6922b47360cc3a42ae778e998e2667391bb202cf703ae6e8d01520dd

SHA256 sample05.exe:

fe6d6d15e0ffa8717c2a5ac80b7f117e853c05cd642c746bb2eab0f70416150d

SHA256 sample06.exe:

0e56c159b8c4fe60ee4a9d9bac1118c9467965086d1de239e1b27ecbbe540182

SHA256 sample07.exe:

b9d6bf45d5a7fefc79dd567d836474167d97988fc77179a2c7a57f29944550ba

SHA256 sample08.exe:

e551275aa089805c48ec1734d3d4ecd03997663e58892323bf174f0b7eb52504

6.1 Statická analýza první rodiny malwaru

Statická analýza proběhla pomocí níže uvedených nástrojů, v bezpečném prostředí ve virtuální počítači s nainstalovaným operačním systémem GNU Linux distribuce Remnux.

6.1.1 Informace o souborech první rodiny malwaru

Na začátku analýzy vzorků malwaru, bylo zjištěno co nejvíce informací o jednotlivých souborech, zda se jedná o spustitelné soubory a pro jaký operační systém.

Pomocí nástroje file bylo zjištěno, že všechny vzorky první rodiny malwaru jsou spustitelné soubory pro 32bitovou platformu MS Windows, viz. Obrázek 41. Nástroj file – první rodina

```
remnux@remnux:/media/sf_MALWARE/doina$ file sample0*.exe
sample01.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample02.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample03.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample04.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample05.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample06.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample07.exe: PE32 executable (GUI) Intel 80386, for MS Windows
sample08.exe: PE32 executable (GUI) Intel 80386, for MS Windows
remnux@remnux:/media/sf_MALWARE/doina$
```

malwaru [zdroj vlastní]

Obrázek 41. Nástroj file – první rodina malwaru [zdroj vlastní]

Následně byla tato informace ověřena pomocí nástroje TrID, který poskytl více informací. Dle výstupu nástroje TrID se jedná o spustitelný soubor MS Windows, kdy byl použitý kompilátor MS Visual C++. Výstup z nástroje TrID je vidět na dalším obrázku.

```
remnux@remnux:/media/sf_MALWARE/doina$ trid sample0*.exe

TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
Definitions found: 14589
Analyzing...

File: sample01.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample02.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample03.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample04.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample05.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample06.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample07.exe
 54.4% (.EXE) Win32 Executable MS Visual C++ (generic) (31206/45/13)

File: sample08.exe
```

Obrázek 42. Nástroj TrID – první rodina malwaru [zdroj vlastní]

Dalším nástrojem signsrch bylo zjištěno, že v žádném uvedeném vzorku malwaru není použito šifrování nebo komprimace. Výsledky byly shodné pro všechny analyzované vzorky první rodiny malwaru.

```
remnux@remnux:/media/sf_MALWARE/doina$ signsrch sample01.exe

Signsrch 0.2.4
by Luigi Auriemma
e-mail: aluigi@autistici.org
web:    aluigi.org
  optimized search function by Andrew http://www.team5150.com/~andrew/
  disassembler engine by Oleh Yuschuk

- open file "sample01.exe"
- 13824 bytes allocated
- load signatures
- open file /usr/share/signsrch/signsrch.sig
- 3075 signatures in the database
- start 1 threads
- start signatures scanning:

offset  num  description [bits.endian.size]
-----
- 0 signatures found in the file in 0 seconds
- done
```

Obrázek 43. Nástroj signsrch – první rodina malwaru [zdroj vlastní]

Pomocí nástroje analyze nebyl detekován žádný zabalený obsah souboru, nebyly nalezeny podezřelé řetězce a ani škodlivé kombinace importu knihoven, v žádném vzorku první rodiny malwaru. Výstupy analýzy všech vzorků první skupiny malwaru byly shodné a bylo zjištěno, že vzorky obsahují grafické rozhraní GUI a byly zkompileovány se stejnou časovou značkou 18.3.2021 7:16:23, bez uvedení časového pásma, jak je vidět na následujícím obrázku.

```
remnux@remnux:/media/sf_MALWARE/doina$ analyze sample01.exe
* Analyze 0.9 *

-----
/media/sf_MALWARE/doina/sample01.exe
-----

Summary:
-----
Architecture:  IMAGE_FILE_MACHINE_I386
Subsystem:      IMAGE_SUBSYSTEM_WINDOWS_GUI
Compilation Date: 2021-Mar-18 07:16:23
```

Obrázek 44. Nástroj analyze – vzorek01.exe [zdroj vlastní]

Nástrojem ssdeep bylo zjištěno, že svou strukturou jsou si jednotlivé sekce ve vzorcích malwaru podobné. Na základě těchto dat lze říci, že se jedná o stejnou rodinu malwaru.

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUN6:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUN,"sample01.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9buOkrUNqY:
C2jTbZ0pj/vcqP+ctCYSw3GV9bMrUNq,"sample02.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUNla:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUNl,"sample03.exe"

192: C2WjQTbZ1eBopvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUNla:
C2jTbZ0Ej/vcqP+ctCYSw3GV9bhrUNl,"sample04.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUNC:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUN,"sample05.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUN+f:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUN+,"sample06.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUNW:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUN,"sample07.exe"

192: C2WjQTbZ1eBppvfj/j2+cPM3P+Q/tCvwSw3uM76V9bhHOkrUNla:
C2jTbZ0pj/vcqP+ctCYSw3GV9bhrUNl,"sample08.exe"

6.1.2 Analýza vložených řetězců

Pro analýzu vložených řetězců byly použité nástroje strings a floss. Výstup řetězců byl omezen na řetězce delší než 5 znaků.

Přehled počtu nalezených řetězců nástrojem strings a floss je vidět v následující tabulce.

Tabulka 2. Počet nalezených řetězců v první rodině malwaru

Název souboru	Počet řetězců nalezených pomocí strings	Počet řetězců nalezených pomocí floss
sample01.exe	128	136
sample02.exe	130	138
sample03.exe	127	135
sample04.exe	127	135
sample05.exe	127	135
sample06.exe	127	135
sample07.exe	127	135
sample08.exe	127	135

Jelikož nástroj floss poskytoval více nalezených řetězců, byla následující analýza provedena z výsledků nástroje floss. Ze všech nalezených řetězců byly vyhledány shodné řetězce, které se nacházely ve všech vzorcích. Tím bylo nalezeno 126 řetězců, které mohou následně sloužit pro identifikaci této rodiny malwaru.

V těchto řetězcích, které se vyskytují ve všech vzorcích malwaru, byly nalezeny názvy různých systémových knihoven, volání funkcí a podobně.

Mezi nejzajímavější nalezené podezřelé řetězce, které se vyskytují ve všech vzorcích první skupiny malwaru patří:

powershell

-WindowStyle Hidden -ep bypass -file "

connect

CreateThread

DeleteFileA

EnumWindows

ExitProcess

GET %s HTTP/1.0

shutdown

Sleep

start

UpdateWindow

User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:66.0) Gecko/20100101

Firefox/66.0

Z výše uvedených řetězců lze vyvodit chování vzorků malwaru. Například z vložených řetězců „powershell“ a „-WindowStyle Hidden -ep bypass -file " lze vyvodit spuštění powershellového skriptu bez zobrazení viditelného okna. [61]

Z vloženého řetězce „User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0“ lze vyvodit, že se malware bude připojovat na webové stránky s výše uvedenou identifikací.

Mimo výše uvedené společné řetězce byly v každém vzorku malwaru nalezeny řetězce, označené HOST1, HOST2 a PORT, které mohou napovědět, na jaké servery se bude malware připojovat, jak je vidět v následující tabulce.

Tabulka 3. Přehled zájmových řetězců nalezených v první rodině malwaru

Jméno souboru	Řetězec	Řetězec	Řetězec
sample01.exe	HOST1:babama.wtf	HOST2:shops.gt	PORT1:4224
sample02.exe	HOST1:5.101.78.2	HOST2:192.53.123.202	PORT1:4127
sample03.exe	HOST1:93.115.29.50	HOST2:192.53.123.202	PORT1:443
sample04.exe	HOST1:93.115.29.50	HOST2:192.53.123.202	PORT1:443
sample05.exe	HOST1:88.80.188.245	HOST2:88.80.188.245	PORT1:4170
sample06.exe	HOST1:85.25.207.68	HOST2:moscow11.icu	PORT1:4208
sample07.exe	HOST1:5.183.95.197	HOST2:192.169.6.197	PORT1:4210
sample08.exe	HOST1:93.115.29.50	HOST2:192.53.123.202	PORT1:443

Pomocí nástroje base64dump byly vyhledány veškeré řetězce uložené pomocí kódování base64. Tyto výsledky odpovídají nalezeným řetězcům pomocí nástroje floss. Část výstupu z nástroje base64dump na vzorku sample01.exe je vidět na následujícím obrázku.

```
ole32.dll
GetUserNameExA
GetUserNameExW
secur32.dll
GetModuleFileNameExA
psapi.dll
BEGINDATA
HOST1:babama.wtf
HOST2:shops.gt
3.202
PORT1:4224
Fwow64
start
ALLUSERSPROFILE
win32app
Microsoft
kernel32.dll
IsWow64Process
RtlGetVersion
powershell
-WindowStyle Hidden -ep bypass -file "
ntdll.dll
LdrLoadDll
GET %s HTTP/1.0
Host: %s
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:66.0) Gecko/20100101 Firefox/66.0
Connection: close
0"00T0b0y0
5.8?8d8u8
9F9|:
:,<n=
=8>o>d?
051H1_3
7'818V8`8
?N?T?Z?`?f?l?r?x?~?
0 0&0,02080>0D0J0P0V0\0b0h0n0t0z0
```

Obrázek 45. Nalezené řetězce pomocí nástroje base64dump – sample01.exe

[zdroj vlastní]

Nástrojem yara-rules byly analyzovány všechny vzorky první rodiny malwaru a bylo zjištěno, že jsou detekovány stejné škodlivé funkce.

network_tcp_socket

network_dns

win_mutex

win_token

win_files_operation

Str_Win32_Winsock2_Library

powershell
maldoc_find_kernel32_base_method_1
IsPE32
IsWindowsGUI
HasRichSignature
MASMTASM
TASM_MASM
TASM_MASM_additional
PE_Diminisher_v01_additional

```
remnux@remnux:/media/sf_MALWARE/doina$ cat yara-rules_sample01.exe.txt
network_tcp_socket sample01.exe
network_dns sample01.exe
win_mutex sample01.exe
win_token sample01.exe
win_files_operation sample01.exe
Str_Win32_Winsock2_Library sample01.exe
powershell sample01.exe
maldoc_find_kernel32_base_method_1 sample01.exe
IsPE32 sample01.exe
IsWindowsGUI sample01.exe
HasRichSignature sample01.exe
MASMTASM sample01.exe
TASM_MASM sample01.exe
TASM_MASM_additional sample01.exe
PE_Diminisher_v01_additional sample01.exe
```

Obrázek 46. Nalezené informace pomocí nástroje Yara-rule – sample01.exe

[zdroj vlastní]

Pomocí nástroje capa byly analyzovány všechny vzorky první rodiny malwaru. Bylo zjištěno, že obsahují obdobné údaje.

- Spouštění skrytého okna.
- Obfuskace souboru nebo informací.
- Zjišťování uživatelského účtu.
- Zjišťování aplikací Windows.
- Zjišťování souborů a adresářů.
- Zjišťování informací o systému.
- Zjišťování informací o uživatelích.

- Využití příkazového a skriptovacího interpreteru.
- Modul sdílení.
- Persistence pomocí „Plánovače úloh“.
- DNS komunikace (resolve).
- Šifrování dat pomocí RC4.
- Využití operace XOR.
- Obfuskace souboru nebo informací pomocí „Encodyng-Standard Algorithm“.
- Možnost mazání souborů.
- Možnost zápisu do souborů.
- Možnost vytvoření mutexu.
- Zjištění mutexu.
- Možnost vytvoření vlákna.
- Možnost ukončení procesu.

Část výstupu z nástroje capa na vzorku malwaru sample01.exe je vidět na následujícím obrázku.

ATT&CK Tactic	ATT&CK Technique
DEFENSE EVASION	Hide Artifacts::Hidden Window [T1564.003] Obfuscated Files or Information [T1027]
DISCOVERY	Account Discovery [T1087] Application Window Discovery [T1010] File and Directory Discovery [T1083] System Information Discovery [T1082] System Owner/User Discovery [T1033]
EXECUTION	Command and Scripting Interpreter [T1059] Shared Modules [T1129]
PERSISTENCE	Scheduled Task/Job::Scheduled Task [T1053.005]
MBC Objective	MBC Behavior
COMMUNICATION	DNS Communication::Resolve [C0011.001]
CRYPTOGRAPHY	Encrypt Data::RC4 [C0027.009] Encryption Key::RC4 KSA [C0028.002]
DATA	Encode Data::XOR [C0026.002]
DEFENSE EVASION	Obfuscated Files or Information::Encoding-Standard Algorithm [E1027.m02]
FILE SYSTEM	Delete File [C0047] Writes File [C0052]
PROCESS	Check Mutex [C0043] Create Mutex [C0042] Create Thread [C0038] Terminate Process [C0018]
CAPABILITY	NAMESPACE
encode data using XOR	data-manipulation/encoding/xor
encrypt data using RC4 KSA	data-manipulation/encryption/rc4
accept command line arguments (2 matches)	host-interaction/cli
get common file path	host-interaction/file-system
delete file	host-interaction/file-system/delete
write file	host-interaction/file-system/write
enumerate gui resources (2 matches)	host-interaction/gui

Obrázek 47. Nástroj capa – informace o souboru sample01.exe [zdroj vlastní]

6.1.3 Analýza hlaviček vzorků malwaru

Analýza hlaviček PE proběhla pomocí nástrojů PEframe a PEdump, které byly popsány v části věnované nástrojům.

Pomocí nástroje PEframe bylo zjištěno, že všechny vzorky první rodiny malwaru obsahují shodné informace. Významnou charakteristikou této rodiny malwaru je shodný importní hash tzv. imphash 801793b2be29822524e8824fc3c47535. Dále bylo nalezeno datetime 2021-03-18 07:16:23, features mutex, antidebug, packer a další. Následně nástroj PEframe našel IP adresy, které již byly nalezeny pomocí vyhledávání řetězců. Oproti tomu nástroj PEframe nenalezl podezřelé domény, které byly nalezeny pomocí vyhledávání vložených řetězců, viz. Obrázek 48. Informace nalezené nástrojem PEframe – sample01.exe [zdroj vlastní]

```
remnux@remnux:/media/sf_MALWARE/doina$ peframe sample01.exe

-----
File Information (time: 0:00:00.658556)
-----
filename           sample01.exe
filetype           PE32 executable (GUI) Intel 80386, for MS Windows
filesize           13824
hash sha256        00cf9ea8d96ded16db829687c41d0d619868a58ea0ce456f0014168cb553e521
virustotal         /
imagebase          0x400000
entrypoint         0x1000
imphash            801793b2be29822524e8824fc3c47535
datetime           2021-03-18 07:16:23
dll                False
directories         import, tls, relocations
sections           .text, .rdata, .data, .reloc
features           mutex, antidebug, packer

-----
Yara Plugins
-----
IsPE32
IsWindowsGUI
HasRichSignature

-----
Behavior
-----
Xor
network tcp socket
network dns
win mutex
win token
win files operation

-----
Packer
-----
MASMTASM
```

Obrázek 48. Informace nalezené nástrojem PEframe – sample01.exe [zdroj vlastní]

Za pomoci nástroje PEDump byly vypsaný informace o hlavičkách v souborech první rodiny malwaru. Následnou analýzou bylo zjištěno, že výstupy PEDump ze všech vzorků první rodiny malwaru jsou shodné, včetně rozložení sekcí v souborech. Na Obrázek 49. Informace nalezené nástrojem pedump – sample01.exe [zdroj vlastní] je vidět část výstupu PEDump na vzorku malwaru sample01.exe

```
remnux@remnux:/media/sf_MALWARE/doina$ pedump sample01.exe

=== MZ Header ===

      signature:                "MZ"
  bytes_in_last_block:         144      0x90
    blocks_in_file:             3         3
      num_relocs:                0         0
  header_paragraphs:           4         4
min_extra_paragraphs:         0         0
max_extra_paragraphs:        65535     0xffff
      ss:                         0         0
      sp:                        184      0xb8
      checksum:                   0         0
      ip:                          0         0
      cs:                          0         0
  reloc_table_offset:          64        0x40
    overlay_number:              0         0
    reserved0:                    0         0
    oem_id:                        0         0
    oem_info:                       0         0
    reserved2:                       0         0
    reserved3:                       0         0
    reserved4:                       0         0
    reserved5:                       0         0
    reserved6:                       0         0
      lfaneu:                      184      0xb8

=== DOS STUB ===

00000000: 0e 1f ba 0e 00 b4 09 cd 21 b8 01 4c cd 21 54 68 |.....!..L.!Th|
00000010: 69 73 20 70 72 6f 67 72 61 6d 20 63 61 6e 6e 6f |is program canno|
00000020: 74 20 62 65 20 72 75 6e 20 69 6e 20 44 4f 53 20 |t be run in DOS |
00000030: 6d 6f 64 65 2e 0d 0d 0a 24 00 00 00 00 00 00 00 |mode...$......|

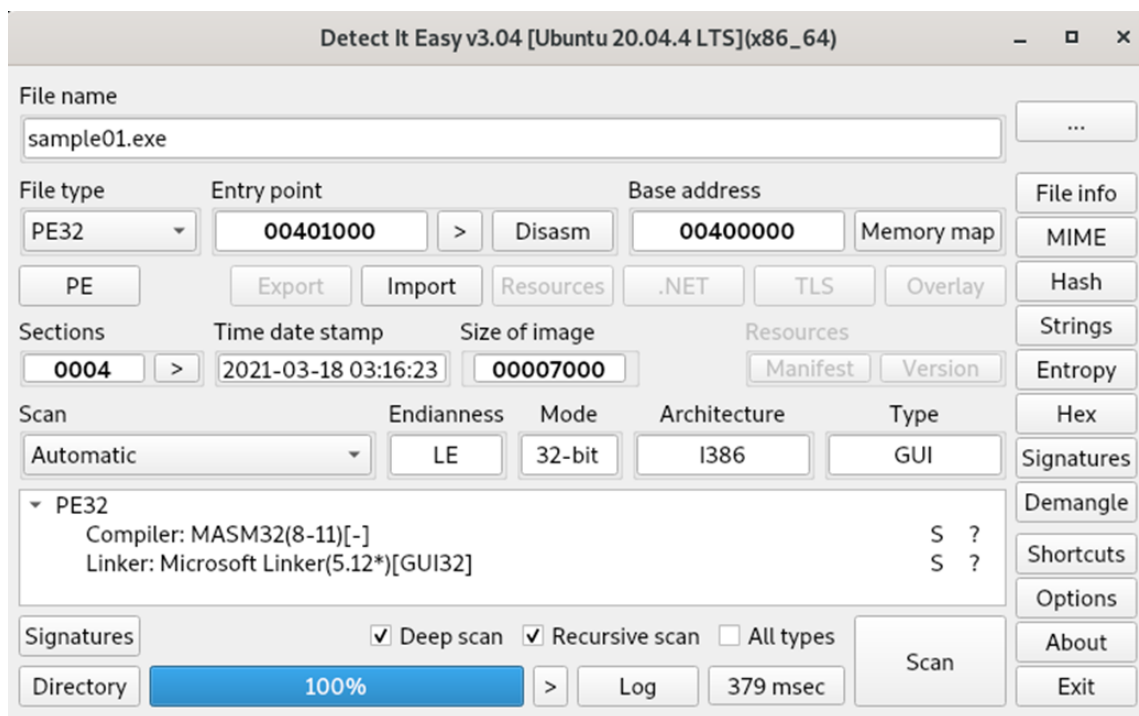
=== RICH Header ===

  ID  VER      COUNT  DESCRIPTION
  13  1f8e      126
  12  20fc         1
```

Obrázek 49. Informace nalezené nástrojem pedump – sample01.exe
[zdroj vlastní]

Všechny vzorky první rodiny malwaru byly analyzovány pomocí nástroje Detect-It-Easy a bylo zjištěno, že veškeré tyto vzorky byly zkompileovány ve stejnou dobu, tj. 18.3.2021 v 03:16:23 bez udání časového pásma, pomocí kompilátoru MASM32(8-11), což je Microsoft Macro Assembler, tj. kompilátor, který byl součástí Visual C++ 2005 .NET. [62]

Dále dle nástroje Detect-It-Easy byl použit linker Microsoft Linker ve verzi 5.12*, jak je vidět na dalším obrázku.



Obrázek 50. Nástroj Detect It Easy vzorek malwaru sample01.exe [zdroj vlastní]

Pomocí tohoto nástroje byly následně vypsány informace o souboru a u všech analyzovaných vzorků bylo zjištěno, že obsahují shodné informace o rozložení sekcí souboru. Tyto nálezy odpovídají výsledkům analýzy pomocí nástroje PEdump.

6.1.4 Antivirová kontrola ClamAV

Pomocí antivirového systému ClamAV byly vzorky podrobeny antivirové kontrole a bylo zjištěno, že všechny tyto vzorky malwaru ClamAV detekuje jako Win.Malware.Doina-9878360-0.

Na dalším obrázku je vidět výstup antivirové kontroly první rodiny malwaru pomocí antiviru ClamAV s aktuální virovou databází.

```
remnux@remnux:/media/sf_MALWARE/doina$ clamscan -i sample0*.exe
/media/sf_MALWARE/doina/sample01.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample02.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample03.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample04.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample05.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample06.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample07.exe: Win.Malware.Doina-9878360-0 FOUND
/media/sf_MALWARE/doina/sample08.exe: Win.Malware.Doina-9878360-0 FOUND

----- SCAN SUMMARY -----
Known viruses: 8612220
Engine version: 0.103.5
Scanned directories: 0
Scanned files: 8
Infected files: 8
Data scanned: 0.09 MB
Data read: 0.09 MB (ratio 1.00:1)
Time: 21.013 sec (0 m 21 s)
Start Date: 2022:04:25 06:39:37
End Date: 2022:04:25 06:39:58
```

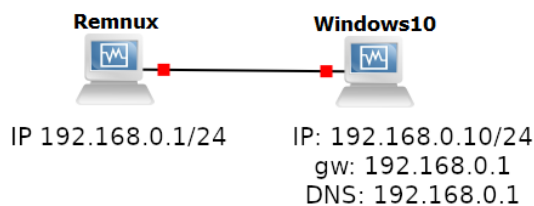
Obrázek 51. Antivirová kontrola ClamAV [zdroj vlastní]

6.2 Disassembling/Debugging

Vzorky malwaru s názvy sample01.exe až sample08.exe byly postupně načteny do nástroje Ghidra. Bylo nalezeno velké množství importů a funkcí a přes veškeré úsilí se nepodařilo z obfuskovaného kódu získat smysluplné informace.

6.3 Dynamická analýza první rodiny malwaru

Před samotnou dynamickou analýzou bylo připraveno laboratorní prostředí. Pomocí Oracle VM VirtualBox⁴⁶ byl vytvořen virtuální počítač s operačním systémem Remnux. Dále byl vytvořený virtuální počítač MS Windows verze 10.0.16299 Build 16299. Byla zvolena starší verze MS Windows 10, protože v nových vydáních tohoto operačního systému již nelze plně deaktivovat antivirovou ochranu MS Defender. Pomocí Group Policies byla zakázána antivirová kontrola a firewall. V operačním systému Remnux byla nastavena síťová karta na IP adresu 192.168.0.1/24 a v operačním systému MS Windows byla nastavená síťová karta na IP adresu 192.168.0.10/24, výchozí síťová brána 192.168.0.1 a DNS na 192.168.0.1, tak jak je vidět na Obrázek 52. Nastavení laboratorního prostředí [zdroj vlastní] Síťová komunikace byla vyzkoušena pomocí nástroje ping. Následně byl proveden snímek virtuálních strojů pomocí VirtualBoxu tak, aby bylo možné se vždy vrátit do tohoto výchozího stavu.



Obrázek 52. Nastavení laboratorního prostředí [zdroj vlastní]

V operačním systému Remnux byl dále spuštěn nástroj InetSim, Accept-all-ips a wireshark. V operačním systému MS Windows byl spuštěný nástroj RegShot, Process Hacker, Process Monitor a nástrojem WinDump s parametry „windump -i 1 -q -w C:\tmp\sample01.pcap -n -W 10 -U -s 0“ byl zachytáván síťový provoz do souboru c:\tmp\sample01.pcap. Pak byl vytvořen první snímek registrů a spuštěný monitoring změn v Process Monitoru. Poté již bylo možné spouštět jednotlivé vzorky malwaru. Malware byl spuštěn několik minut, dokud

⁴⁶ Ke stažení na <https://virtualbox.org>

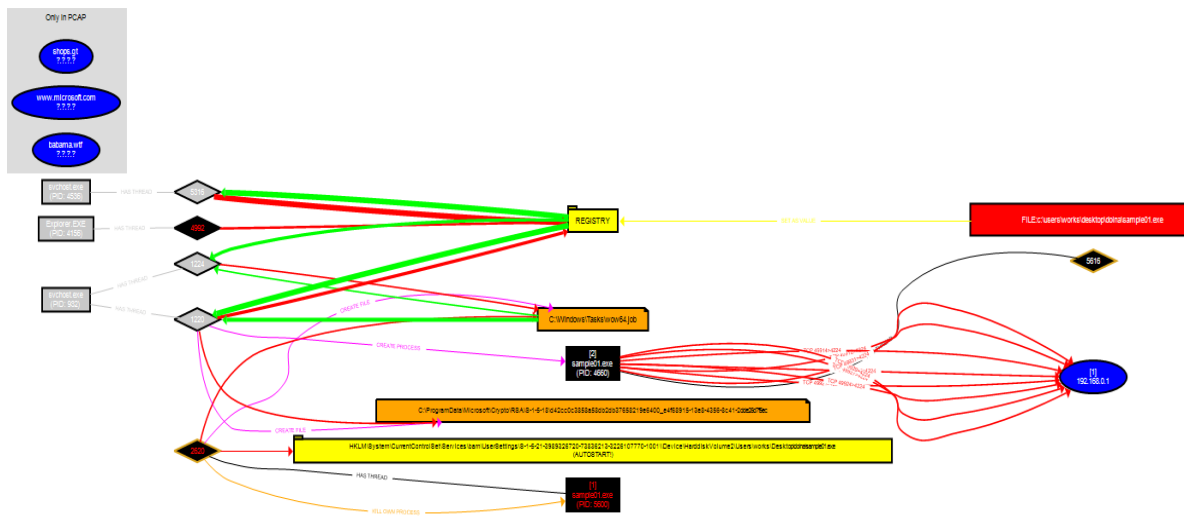
nedošlo k samovolnému ukončení procesu vzorku malwaru. Následně byl vytvořen nový snímek registrů pomocí nástroje RegDump a v tomto nástroji byl vytvořen textový soubor změn registru pro další analýzu. Následně byly získané logy načteny v čistém operačním systému MS Windows do nástroje ProcDot, pomocí kterého byla graficky zobrazena činnost malwaru. Z načtených logů byl v nástroji ProcDot použitý filtr na zájmové procesy.

6.3.1 Vzorek malwaru sample01.exe

Logy získané pomocí dynamické analýzy byly analyzovány v nástroji ProcDot. Po spuštění vzorku malwaru sample01.exe došlo k vytvoření stejnojmenného procesu, který vytvořil soubor C:\Windows\Tasks\wow64.job. Poté byly vloženy klíče a hodnoty do registru MS Windows tak, aby byla vytvořena „Plánovaná úloha“ MS Windows s názvem wow64.job. Tato plánovaná úloha automaticky spouští každé dvě minuty soubor sample01.exe z aktuálního umístění, a to bez ohledu na přihlášení uživatele. Tím je zaručeno automatické spuštění malwaru po restartu operačního systému a dále je zaručeno automatické spuštění malwaru při náhodném ukončení uživatelem nebo antivirovým systémem. Proces sample01.exe nastavil klíč registru

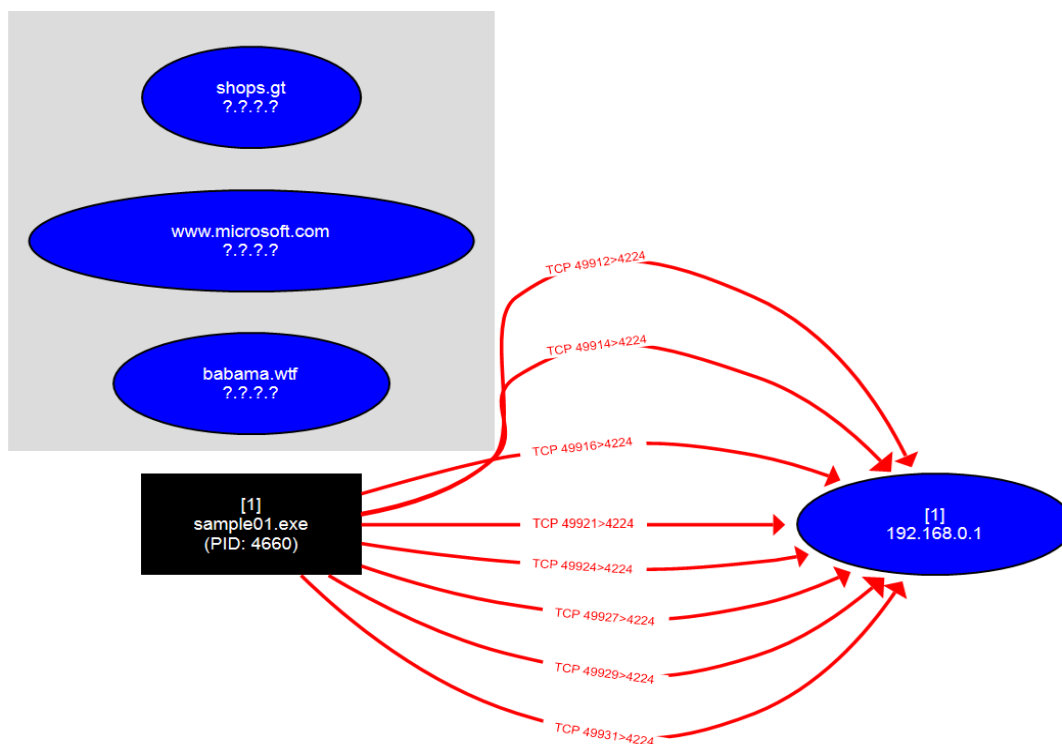
```
HKLM\SYSTEM\ControlSet001\Services\bam\UserSettings\S-1-5-21-3989326720-73836213-3226107770-1001\Device\HarddiskVolume2\Users\works\Desktop\doina\sample01.exe.
```

Jedná se o nastavení služby operačního systému MS Windows s názvem Background Activity Moderator Driver. Tato služba je zodpovědná za řízení aktivity aplikací na pozadí. [63] Poté se proces malwaru ukončil. Na následujícím obrázku je grafický výstup z nástroje ProcDot, do kterého byly načteny logy z dynamické analýzy vzorku malwaru sample01.exe.



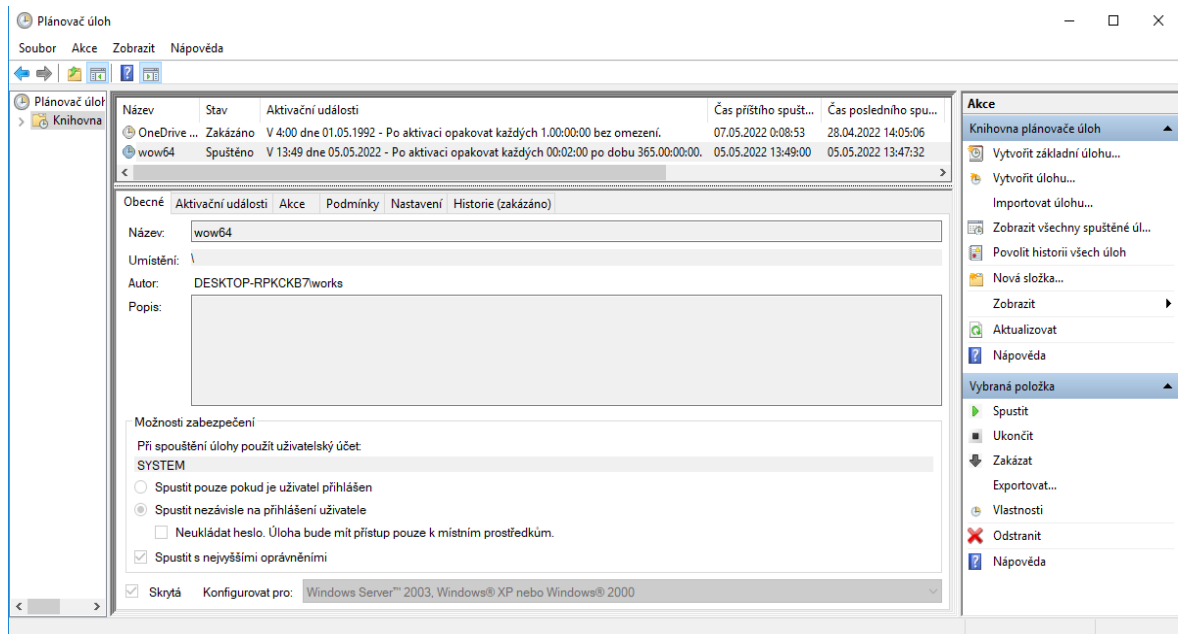
Obrázek 53. Přehled chování malwaru sample01.exe PID 5600 v nástroji ProcDot [zdroj vlastní]

„Plánovaná úloha“ MS Windows, kterou vytvořil malware se automaticky spustila. Tím došlo k vytvoření nového procesu s názvem sample01.exe. Tento nový proces navázal TCP spojení na doménu babama.wtf na port 4224 na port TCP a na doménu shops.gt na port 4224 TCP. Navázání TCP spojení proběhlo na virtuálním počítači Remnux se spuštěnými službami InetSim a Accept-All-Access. Tím došlo k navázání TCP spojení na požadované IP adresy, jak je zobrazeno na dalším obrázku V této TCP komunikaci již malware pravděpodobně očekával řídicí instrukce. Žádná další činnost malwaru nebyla zaznamenána.



Obrázek 54. Přehled chování malwaru `sample01.exe` PID 4600 v nástroji ProcDot [zdroj vlastní]

Na následujícím obrázku je vidět vytvořená „Plánovaná úloha“, kterou vytvořil vzorek malwaru sample01.exe.



Obrázek 55. Plánovač úloh infikovaného počítače malwarem sample01.exe

[zdroj vlastní]

6.3.2 Vzorky malwaru sample02.exe až sample08.exe

Logy získané pomocí dynamické analýzy byly analyzovány v nástroji ProcDot. Chování vzorků malwaru sample02.exe až sample08.exe je stejné jako u vzorku sample01.exe. Byl vytvořen proces, který vytvořil soubor C:\Windows\Tasks\wow64.job a vložil klíče a hodnoty do registru MS Windows tak, aby byla vytvořena „Plánovaná úloha“ MS Windows s názvem wow64. Tato plánovaná úloha automaticky spouští každé dvě minuty soubor malwaru z aktuálního umístění, a to bez ohledu na přihlášení uživatele. Tím je zaručeno automatické spuštění malwaru po restartu operačního systému a dále je zaručeno automatické spuštění malwaru při náhodném ukončení uživatelem nebo antivirovým systémem. Nakonec je nastavena služba MS Windows s názvem Background Activity Moderator Driver, která je zodpovědná za řízení aktivity aplikací na pozadí. Poté se proces malwaru ukončí.

U vzorku malwaru sample02.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 5.101.78.2 na port 4127 a na IP adresu 192.53.123.202 na port 4127.

U vzorku malwaru sample03.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 93.115.29.50 na port 443 a na IP adresu 192.53.123.202 na port 443.

U vzorku malwaru sample04.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 93.115.29.50 na port 443 a na IP adresu 192.53.123.202 na port 443.

U vzorku malwaru sample05.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 88.80.188.245 na port 4170.

U vzorku malwaru sample06.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 85.25.207.68 na port 4208 a na doménu moscow11.icu na port 4208.

U vzorku malwaru sample07.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 5.183.95.197 na port 4210 a na IP adresu 192.169.6.197 na port 4210.

U vzorku malwaru sample08.exe „Plánovaná úloha“ navázala TCP spojení na IP adresu 93.115.29.50 na port 443 a na IP adresu 192.53.123.202 na port 443.

6.3.3 Detekce první rodiny malwaru

Z provedených analýz první rodiny malwaru je možné stanovit jednoznačný indikátor kompromitace operačního systému touto rodinou malwaru.

- Naplánovaná úloha wow64 v „Plánovač úloh“, spuštěná každé 2 minuty nezávisle na přihlášení uživatele, s nejvyššími oprávněními.
- Soubor C:\Windows\Tasks\wow64.job
- Přítomnost klíče registru MS Windows
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Schedule\TaskCache\Tree\wow64

Pro detekci této rodiny malwaru lze využít jednotného importního hashe:

```
imphash 801793b2be29822524e8824fc3c47535
```

Tento importní hash je jedinečný pro tuto rodinu malwaru.

Pro účely detekce bylo napsáno následující YARA pravidlo, které vychází z nalezených řetězců v souborech první rodiny malwaru.

```
rule Trojan_Doina {  
  
    meta:  
  
        author = "Vaclav Hess"  
  
        date = "2022-05-05"  
  
        description = "Detekce Trojanu Doina"  
  
    strings:  
  
        $mz = { 4D 5A }  
  
        $s1 = "powershell" ascii  
  
        $s2 = "-WindowStyle Hidden -ep bypass -file" ascii  
  
        $s3 = "GET %s HTTP/1.0" ascii  
  
        $s4 = "PORT1:" ascii  
  
        $s5 = "HOST1:" ascii  
  
        $s6 = "HOST2:" ascii  
  
        $s7 = "User-Agent:" ascii  
  
        $s8 = "Sleep" ascii  
  
        $s9 = "start" ascii  
  
        $s10 = "EnumWindows" ascii  
  
    condition:  
  
        ( $mz and filesize < 20KB and all of ($s*) )  
  
} [zdroj vlastni]
```

Následně bylo toto pravidlo ověřeno pomocí nástroje YARA, kdy v každém souboru sample01.exe až sample08.exe byl detekován Trojan Doina. Při testu jakéhokoli jiného vzorku malwaru byl výsledek negativní.

7 ANALÝZA DRUHÉ RODINY MALWARU

Pro účely diplomové práce bylo získáno pět vzorků jedné rodiny malwaru. I když byla známá identita těchto vzorků malwaru, bylo k nim přistupováno jako k neznámým vzorkům malwaru, a proto byly označeny jako „druhá rodina malwaru“. Tyto soubory byly označeny názvy 2sample01.exe až 2sample05.exe. Velikost těchto souborů je od 483 KB do 1,4 MB. Na začátku byl spočítán hash SHA256 jednotlivých souborů, aby bylo vyloučeno, že se jedná o stejný soubor.

SHA256 2sample01.exe:

20c8a88fe7f680a0cdaf61fafbc9e6dcdf626f326c5392b4d2660fec9adc2bf1

SHA256 2sample02.exe:

48e3b8c67534ca86179306e98a15a0a71273f9fd845fd026c4dd584a1dfa7928

SHA256 2sample03.exe:

45fe7e8293ea52c150e0caa578ef2dcf2da7f92cf2da1ede6f9f40a78e110f33

SHA256 2sample04.exe:

c3c23cf38a88f1722b981b7a60bfd67f63b8612c15f748293bf4e7892e064030

SHA256 2sample05.exe:

48b0d7d0fbfb271a9454c888ec3fac0d59d73636a77f358601f2b19715183a54

7.1 Statická analýza druhé rodiny malwaru

Statická analýza proběhla pomocí níže uvedených nástrojů, v bezpečném prostředí ve virtuální počítači s nainstalovaným operačním systémem GNU Linux distribuce Remnux.

7.1.1 Informace o souborech druhé rodiny malwaru

Nejprve bylo zjištěno, o jaký typ souborů se jedná pomocí nástroje file. Bylo zjištěno, že se jedná o spustitelné soubory pro 32bitovou platformu MS Windows. Z výpisu je zřejmé, že

```
remnux@remnux:/media/sf_MALWARE/RAT$ file 2sample0*
2sample01.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
2sample02.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
2sample03.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
2sample04.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
2sample05.exe: PE32 executable (GUI) Intel 80386 Mono/.Net assembly, for MS Windows
remnux@remnux:/media/sf_MALWARE/RAT$
```

Obrázek 56. Nástroj file – druhá rodina malwaru [zdroj vlastní]

pro tuto rodinu malwaru byl využitý framework Microsoft .NET, jak je zobrazeno na obrázku 56.

Nástrojem TrID bylo zjištěno, že všechny vzorky druhé rodiny malwaru mají shodnou identifikaci .NET. Na níže uvedeném obrázku je proto uvedený pouze vzorek malwaru 2sample01.exe.

```
TrID/32 - File Identifier v2.24 - (C) 2003-16 By M.Pontello
Definitions found: 14589
Analyzing...

Collecting data from file: 2sample01.exe
60.4% (.EXE) Generic CIL Executable (.NET, Mono, etc.) (73123/4/13)
10.8% (.SCR) Windows screen saver (13101/52/3)
 8.6% (.EXE) Win64 Executable (generic) (10523/12/4)
 5.4% (.DLL) Win32 Dynamic Link Library (generic) (6578/25/2)
 4.1% (.EXE) Win16 NE executable (generic) (5038/12/1)
```

Obrázek 58. Nástroj TrID – druhá rodina malwaru [zdroj vlastní]

Nástrojem sigsrch byla provedena analýza jednotlivých vzorků. Bylo zjištěno, že ve vzorku 2sample01.exe detekoval nástroj sigsrch 17 signatur, jak je vidět na dalším obrázku.

```
offset  num  description [bits.endian.size]
-----
0000041d 1090 Zlib base_length [32.be.116]
00000420 1089 Zlib base_length [32.le.116]
00000498 1086 Zlib dist_code [..512]
00000698 1091 Zlib base_dist [32.le.120]
000006cd 2417 MBC2 [32.le.248&]
000006d0 2418 MBC2 [32.be.248&]
00000710 3052 function where is handled the ZipCrypto password [32.le.12&]
0000071d 2295 zlib_inflate_lengthExtraBits [32.be.116]
00000720 2294 zlib_inflate_lengthExtraBits [32.le.116]
00000798 1087 Zlib length_code [..256]
00000898 2303 zlib_inflate_distanceExtraBits [32.le.120]
00000a10 1113 Bzip2 BZ2_rNums [32.le.2048]
00001210 2291 zlib_inflate_lengthStarts [32.le.116]
00002ab0 2298 zlib_inflate_distanceStarts [32.le.120]
00003028 3038 unlz_x_table_three [32.le.64]
00003070 2415 Misty_md5const [32.le.256]
000f9bc4 3032 PADDINGXXPADDING [..16]

- 17 signatures found in the file in 1 seconds
```

Obrázek 57. Nástroj sigsrch - 2sample01.exe [zdroj vlastní]

Dále bylo nalezeno ve vzorcích 2sample02.exe až 2sample05.exe celkem 16 signatur, jak je vidět na dalším obrázku.

```

offset  num  description [bits.endian.size]
-----
0000041d 1090 Zlib base_length [32.be.116]
00000420 1089 Zlib base_length [32.le.116]
00000498 1086 Zlib dist_code [..512]
00000698 1091 Zlib base_dist [32.le.120]
000006cd 2417 MBC2 [32.le.248&]
000006d0 2418 MBC2 [32.be.248&]
00000710 3052 function where is handled the ZipCrypto password [32.le.12&]
0000071d 2295 zinflate_lengthExtraBits [32.be.116]
00000720 2294 zinflate_lengthExtraBits [32.le.116]
00000798 1087 Zlib length_code [..256]
00000898 2303 zinflate_distanceExtraBits [32.le.120]
00000a10 1113 Bzip2 BZ2_rNums [32.le.2048]
00001210 2291 zinflate_lengthStarts [32.le.116]
00002ab0 2298 zinflate_distanceStarts [32.le.120]
00003028 3038 unlzx table three [32.le.64]
00078930 3032 PADDINGXXPADDING [..16]

- 16 signatures found in the file in 1 seconds

```

Obrázek 59. Nástroj signsrch – 2sample02.exe až 2sample05.exe [zdroj vlastní]

Pomocí nástroje analýzy bylo u vzorku 2sample01.exe zjištěno, že byl zkompileován dne 30.3.2022 17:07:30 bez uvedení časového pásma. Jedná se o 32bitový spustitelný soubor pro platformu MS Windows s využitím grafického rozhraní. Jako použitá jazyková mutace byla detekována americká angličtina. Další položky byly pravděpodobně vygenerovány

```

* Manalyze 0.9 *
-----
/media/sf_MALWARE/RAT/2sample01.exe
-----

Summary:
-----
Architecture:      IMAGE_FILE_MACHINE_I386
Subsystem:         IMAGE_SUBSYSTEM_WINDOWS_GUI
Compilation Date:  2022-Mar-30 17:07:30
Detected languages: English - United States
ProductName:       qn
CompanyName:       jeSvDsIhjDbsclmYjqSeAuLZ0t
InternalName:      nkyBP6HbWUkCav5HaM0.exe
LegalCopyright:    j4kMsHV
Comments:          KQ2op9T8KBihf4rD9YljFYsN9R
OriginalFilename:  39SZHaPfFwIese77ogpocwfNMZg.exe
ProductVersion:    411.749.97.890
File Version:      622.001.050.050

```

Obrázek 60. Část výstupu nástroje analýzy na vzorku malwaru sample01.exe

[zdroj vlastní]

náhodně, viz. Obrázek 60. Část výstupu nástroje analýzy na vzorku malwaru sample01.exe [zdroj vlastní]

U ostatních vzorků druhé rodiny malwaru bylo zjištěno, že datum kompilace je 15.3.2022 23:42:27, nástroj analýzy u těchto vzorků detekoval, že se jedná o spustitelný soubor pro 32bitovou platformu MS Windows s grafickým rozhraním. Ostatní údaje jsou pravděpodobně vygenerovány náhodně.

Nástrojem ssdeep bylo zjištěno, že vzorek malwaru 2sample01.exe je svou strukturou rozdílný oproti ostatním vzorkům. Vzorky 2sample02.exe, 2sample03.exe, 2sample04.exe a 2sample05.exe jsou si velmi podobné.

```
ssdeep,1.1--blocksize:hash:hash,filename
12288:/9pLLk45WSSY1BX6f4bIS7rMNetPfc9Vs6IFGs0jxAqXj9xPSI0dzNgCoD7WX+Iu:/9pP5WS3lrMnyc9TJPCXBi,"/media/sf_MALWARE/RAT/2sample01.exe"
12288:R0jQEMCYFzhC79Lb3TS+E+gPa/10k1iV7:RpHCcNk3ptKs10,"/media/sf_MALWARE/RAT/2sample02.exe"
12288:h0jQEMCYFzhC79Lb3TS+E+gPa/10k1iV7:hpHCcNk3ptKs10,"/media/sf_MALWARE/RAT/2sample03.exe"
12288:x0jQEMCYFzhC79Lb3TS+E+gPa/10k1iV7:xpHCcNk3ptKs10,"/media/sf_MALWARE/RAT/2sample04.exe"
12288:B0jQEMCYFzhC79Lb3TS+E+gPa/10k1iV7:BpHCcNk3ptKs10,"/media/sf_MALWARE/RAT/2sample05.exe"
```

Obrázek 62. Nástroj ssdeep – druhá rodina malwaru [zdroj vlastní]

Za pomoci nástroje xorseach byl v jednotlivých souborech vyhledaný řetězec „http“, kdy bylo zjištěno, že žádný z uvedených vzorků druhé rodiny malwaru neobsahuje algoritmus XOR, ROT, ADD řetězce „http“.

7.1.2 Analýza vložených řetězců

Pro analýzu vložených řetězců byly použity nástroje strings a floss. Výstup řetězců byl omezen na řetězce delší než 5 znaků. Přehled nalezených řetězců nástrojem strings a floss je vidět v následující tabulce.

Tabulka 4. Počet nalezených řetězců u druhé rodiny malwaru

Název souboru	Počet řetězců – nástroj strings	Počet řetězců – nástroj floss
2sample01.exe	2356	3186
2sample02.exe	851	1723
2sample03.exe	851	1723
2sample04.exe	851	1722
2sample05.exe	851	1722

Jelikož nástroj floss poskytuje větší seznam nalezených řetězců, byl pro následující analýzu použitý výstup z nástroje floss.

Z druhé rodiny malwaru byly vyfiltrovány ty řetězce, které se vyskytují ve všech vzorcích. Celkem bylo nalezeno celkem 1461 řetězců. Z těchto vložených řetězců si lze udělat přehled o možné činnosti rodiny malwaru.

Mezi podezřelé řetězce vložené v analyzovaných souborech druhé rodiny malwaru byly nalezeny řetězce, které mohou napovídat o činnosti malwaru. Mezi ně patří řetězce BIOS-Name, BIOSVersion, CPUName, MotherboardID, MotherboardManufacturer, MotherboardName, Antivirus, Firewall, ipinfo, geoplugin_city. Z nich je zřejmé, že tento malware zjišťuje informaci o zařízení, na kterém je spuštěný, včetně informací o zabezpečení. Dále byly nalezeny řetězce Screens, Webcams, Microphones, [Clipboard] Saving information, Clipboard [Files].txt, Clipboard [Text].txt, keyboar, mouse, mouse_event, SELECT * FROM Win32_PnPEntity WHERE (PNPClass = 'Image' OR PNPClass = 'Camera'), set CDAudio door closed, set CDAudio door open, které napovídají že malware může zjišťovat citlivých údajů z mikrofону, webkamery, schránky operačního systému Windows a podobně a obsluhovat zařízení jako je CD mechanika. Další řetězce, které mohou napovídat o činnosti malwaru byly http://, HttpWebRequest, HttpWebResponse, POST / HTTP/1.1, různé druhy user-agenta webového prohlížeče a další napovídají, že malware bude mít možnost se připojovat na webové servery. Následující nalezené řetězce cipherText, CreateEncryptor, FromBase64String, PKWare AES128, PKWare AES192, PKWare AES256, mohou napovědět o možnostech šifrování malwaru. Další řetězce, které byly nalezeny a mohou napovídat o perzistenci malwaru jsou sctasks.exe /create /tn, Software\Microsoft\Windows\CurrentVersion\Run, SOFTWARE\Microsoft\Windows NT\CurrentVersion, Software\Microsoft\Windows NT\CurrentVersion\Winlogon, cmd.exe, cscript.exe, explorer.exe, powershell.exe, \System32\cmd.exe.

Byly také nalezeny řetězce s názvy DiscordPath, SteamApps, SteamPath, SteamUser, Telegram, Telegram.exe, TelegramPath a další. Ty mohou znamenat na jaké aplikace se bude malware zaměřovat.

V řetězcích druhé rodiny malwaru bylo nalezeno textové logo Dark Crystal RAT, které je na následujícím obrázku.

```
Processing other information...
sysinfo
[SystemInfromation] Saving information...
screens
[Dark Crystal RAT]
PC Name:
User Name:
Windows:
CPU Name:
CPUName
CPUDescription
CPU Cores:
CPUCores
CPULogicalProcessors
```

Obrázek 63. Logo Dark Crystal RAT ve vložených řetězcích [zdroj vlastní]

A dále byly nalezeny řetězce DCRat.Code, DCRat-Log#.

Z toho lze vyvodit, že se jedná o rodinu malwaru s názvem Dark Crystal RAT, který lze zakoupit za 55\$. [64]

7.1.3 Analýza hlaviček vzorků druhé rodiny malwaru

Nástrojem PEframe bylo zjištěno, že všechny vzorky druhé rodiny malwaru jsou spustitelné soubory pro 32bitovou platformu MS Windows, využívající frameworku .NET. Dále bylo zjištěno, že všechny soubory mají shodnou importní hash tzv. imphash, která je f34d5f2d4577ed6d9ceec516c1f5a744. Datum kompilace vzorku 2sample01.exe je 30.3.2022 17:07:30, u ostatních vzorků to je 15.3.2022 23:42:47. Všechny tyto časové značky jsou bez udání časového pásma. Yara Plugins v nástroji PEframe zobrazil u druhé rodiny malwaru shodné následující informace.

IsPE32

IsNET EXE

IsWindowsGUI

NETexecutableMicrosoft

CRC32 poly Constant

V sekci nazvané „Behavior“ (chování) detekoval nástroj PEframe u všech vzorků druhé rodiny malwaru xor. a screenshot, tj. pořizování snímků obrazovky. U vzorku malwaru 2sample02.exe, 2sample03.exe, 2sample04.exe a 2sample05.exe detekoval nástroj PEframe navíc ještě WMI VM Detect, což znamená detekci běhu ve virtuálním prostředí.

Nástrojem PEDump byly analyzované veškeré vzorky druhé rodiny malwaru a bylo zjištěno, že mají všechny vzorky obdobné rozložení sekcí. Položky jako je např. „ProductName“ jsou u všech vzorků pravděpodobně náhodně vygenerované.

```

=== SECTIONS ===
NAME             RVA             VSZ             RAW_SZ          RAW_PTR         nREL            REL_PTR         nLINE           LINE_PTR        FLAGS
.text            2000            782a4           78400           200             0               0               0               0               60000020 R-X CODE
.rsrc            7c000           354             400             78600           0               0               0               0               40000040 R-- IDATA
.reloc           7e000           c               200             78a00           0               0               0               0               42000040 R-- IDATA DISCARDABLE

=== RESOURCES ===
FILE_OFFSET      CP  LANG          SIZE  TYPE          NAME
0x78658          1252 0x409          764  VERSION      #1

=== IMPORTS ===
MODULE_NAME      HINT  ORD  FUNCTION_NAME
mscorlib.dll      0     0    _CorExeMain

=== VERSION INFO ===
# VS_FIXEDFILEINFO:
FileVersion      : 921.574.176.583
ProductVersion   : 414.404.593.389
StrucVersion     : 0x10000
FileFlagsMask    : 0x3f
FileFlags        : 0
FileOS           : 4
FileType         : 0
FileSubtype      : 0

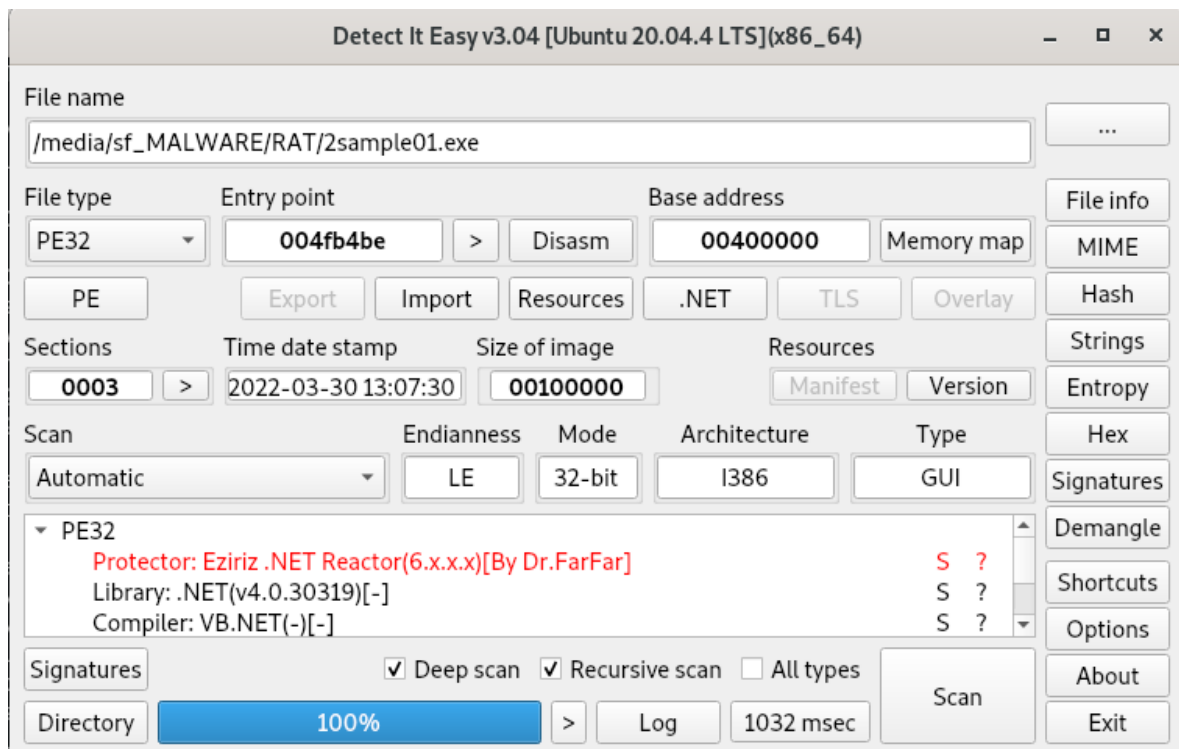
# StringTable 040904b0:
ProductName      : "4S2dvBGbP39Gf0VwE"
CompanyName      : "DVzLFJvX5aLZo3o9iWAWLQD"
InternalName     : "jZzuSUmIVdiCmLMLRYG.exe"
LegalCopyright   : "smVGifk9wKEbRHCj"
Comments         : "nFaxCE5WuKVJR"
OriginalFilename : "TKoA.exe"
ProductVersion   : "414.404.593.389"
FileVersion      : "921.574.176.583"

VarFileInfo      : [ 0x409, 0x514 ]

```

Obrázek 64. Nástroj pedump – 2sample01.exe [zdroj vlastní]

Nástrojem Detect-It-Easy byly analyzovány veškeré vzorky druhé rodiny malwaru a bylo mimo jiné zjištěno, že vzorek malwaru 2sample01.exe využívá obfuskátor Eziriz .NET Reactor (6.x.x.x). U ostatních vzorků nebyl nástrojem Detect-It-Easy žádný obfuskátor detekován, viz. obrázku 65.



Obrázek 65. Nástroj Detect It Easy – 2sample01.exe

Pomocí tohoto nástroje byly následně vypsány informace o souboru a u všech analyzovaných vzorků bylo zjištěno, že obsahují shodné informace o rozložení sekcí souboru. Tyto výsledky odpovídají výsledkům analýzy pomocí nástroje PEdump.

Nástroj capa neumí zpracovat soubory .NET, proto byly zjištěny pouze obecné informace o souboru, které byly nalezeny i jinými nástroji.

7.1.4 Antivirová kontrola ClamAV

Pomocí antivirového systému ClamAV s aktualizovanou virovou databází byl proveden antivirový sken druhé rodiny malwaru. Bylo zjištěno, že všechny vzorky jsou klasifikovány jako Win.Packed.Passwordstealera-9917697-0, viz. Obrázek 66. Antivirová kontrola Cla-

```
remnux@remnux:/media/sf_MALWARE/RAT$ clamscan -i 2sample0*
/media/sf_MALWARE/RAT/2sample01.exe: Win.Packed.Passwordstealera-9917697-0 FOUND
/media/sf_MALWARE/RAT/2sample02.exe: Win.Packed.Passwordstealera-9917697-0 FOUND
/media/sf_MALWARE/RAT/2sample03.exe: Win.Packed.Passwordstealera-9917697-0 FOUND
/media/sf_MALWARE/RAT/2sample04.exe: Win.Packed.Passwordstealera-9917697-0 FOUND
/media/sf_MALWARE/RAT/2sample05.exe: Win.Packed.Passwordstealera-9917697-0 FOUND
/media/sf_MALWARE/RAT/2sample06.exe: Win.Packed.Passwordstealera-9917697-0 FOUND

----- SCAN SUMMARY -----
Known viruses: 8613827
Engine version: 0.103.5
Scanned directories: 0
Scanned files: 6
Infected files: 6
Data scanned: 3.50 MB
Data read: 3.32 MB (ratio 1.05:1)
Time: 22.676 sec (0 m 22 s)
Start Date: 2022:04:28 07:51:54
End Date: 2022:04:28 07:52:17
```

Obrázek 66. Antivirová kontrola ClamAV [zdroj vlastní]

mAV [zdroj vlastní]

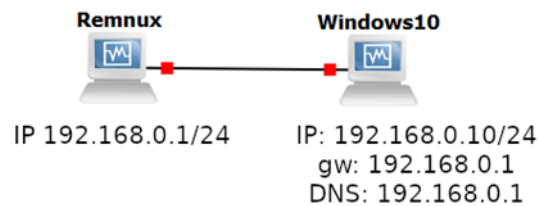
7.2 Disassembling/Debugging

Pomocí nástroje ILSpy byly soubory 2sample01.exe až 2sample05.exe převedeny do zdrojového kódu. Každý z výše uvedených souborů obsahuje kolem 35000 řádků kódu. Tento zdrojový kód byl následně načten do MS Visual Studia. Ani přes veškerou snahu se nepodařilo pomocí debugingu zjistit funkci této rodiny malwaru.

7.3 Dynamická analýza druhé rodiny malwaru

Před samotnou dynamickou analýzou bylo připraveno laboratorní prostředí, shodné jako při analýze první rodiny malwaru. Pomocí Oracle VM VirtualBox byl vytvořen virtuální počítač s operačním systémem Remnux. Dále byl vytvořen virtuální počítač MS Windows verze 10.0.16299 Build 16299. Byla zvolena starší verze MS Windows 10, protože v nových vydáních tohoto operačního systému již nelze plně deaktivovat antivirovou ochranu MS Defender. Pomocí Group Policies byla zakázána antivirová kontrola a firewall. V operačním systému Remnux byla nastavena síťová karta na IP adresu 192.168.0.1/24 a v operačním systému MS Windows byla nastavena síťová karta na IP adresu 192.168.0.10/24, výchozí

síťová brána 192.168.0.1 a také DNS na 192.168.0.1, jak je vidět na Obrázek 67. Nastavení laboratorního prostředí. [zdroj vlastní] Síťová komunikace byla vyzkoušena pomocí nástroje ping. Následně byl proveden snímek virtuálních strojů pomocí VirtualBoxu, tak, aby bylo možné se vždy vrátit do tohoto výchozího stavu.



Obrázek 67. Nastavení laboratorního prostředí. [zdroj vlastní]

V operačním systému Remnux byl dále spuštěn nástroj InetSim, Accept-all-ips a wireshark. V operačním systému MS Windows byl spuštěn nástroj RegShot, Process Hacker, Process Monitor a nástrojem WinDump s parametry „windump -i 1 -q -w C:\tmp\sample01.pcap -n -W 10 -U -s 0“ byl zachytáván síťový provoz do souboru c:\tmp\sample01.pcap. Pak byl vytvořen první snímek registrů a spuštěný monitoring změn v Process Monitoru. Poté již bylo možné spouštět jednotlivé vzorky malwaru. Malware byl spuštěn několik minut. Poté byl ukončen pomocí nástroje Process Hacker. Následně byl vytvořen nový snímek registrů pomocí nástroje RegDump a v tomto nástroji byl vytvořen textový soubor změn registru pro další analýzu. Následně byly získané logy načteny v čistém operačním systému MS Windows do nástroje Procdot, který dokáže graficky zobrazit činnost malwaru. Z načteného logu byl v nástroji Procdot použitý filtr na zájmové procesy.

7.3.1.1 Vzorek malwaru 2sample01.exe

Logy získané pomocí dynamické analýzy byly analyzovány v nástroji ProcDot. Protože po prvním spuštění dynamické analýzy souboru 2sample01.exe neposkytlo dostatek informací o jeho chování, bylo přistoupeno k opakování celé dynamické analýzy na tomto vzorku.

Jelikož byly nalezeny podezřelé úlohy v „Plánovači úloh“ MS Windows, byly tyto úlohy ručně spuštěny v průběhu dynamické analýzy.

Následně byly logy analyzovány nástrojem ProcDot a výsledky byly porovnávány.

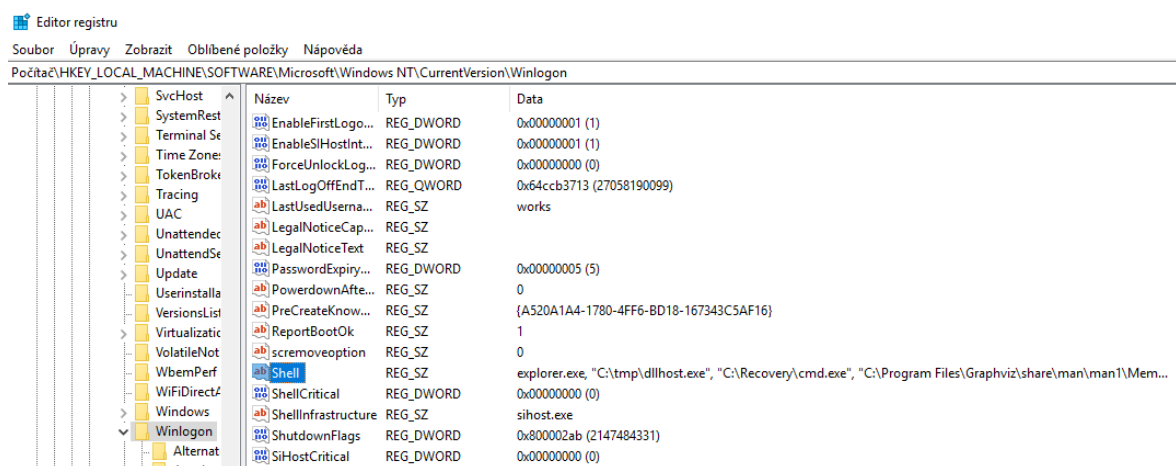
Bylo zjištěno, že po spuštění tohoto vzorku malwaru dojde k vytvoření několika nových spustitelných souborů malwaru. Jejich počet je náhodný. Jejich názvy jsou shodné s názvy jiných legitimních souborů, které jsou již spuštěny v operačním systému. Jejich umístění je také náhodné, nelze tedy predikovat, kde budou tyto nové spustitelné soubory malwaru umístěny, vždy je použita náhodná cesta v souborovém systému. Nové adresáře nebyly malwarem vytvořeny. Poté je vytvořen následující klíč a hodnoty registru:

```
HKLM\SOFTWARE\Microsoft\Windows\
CurrentVersion\Run\<název_vytvořeného_souboru>
```

Tím je zajištěno, že se malware automaticky spustí po restartu systému. Dále je vložen klíč registru:

```
HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell
```

Tento klíč registru obsahuje výchozí shell, který je ve výchozím stavu nastavený na explorer.exe. Do tohoto klíče se vloží názvy, včetně celé cesty, všech nových spustitelných souborů, které malware vytvořil, jak je vidět na dalším obrázku.



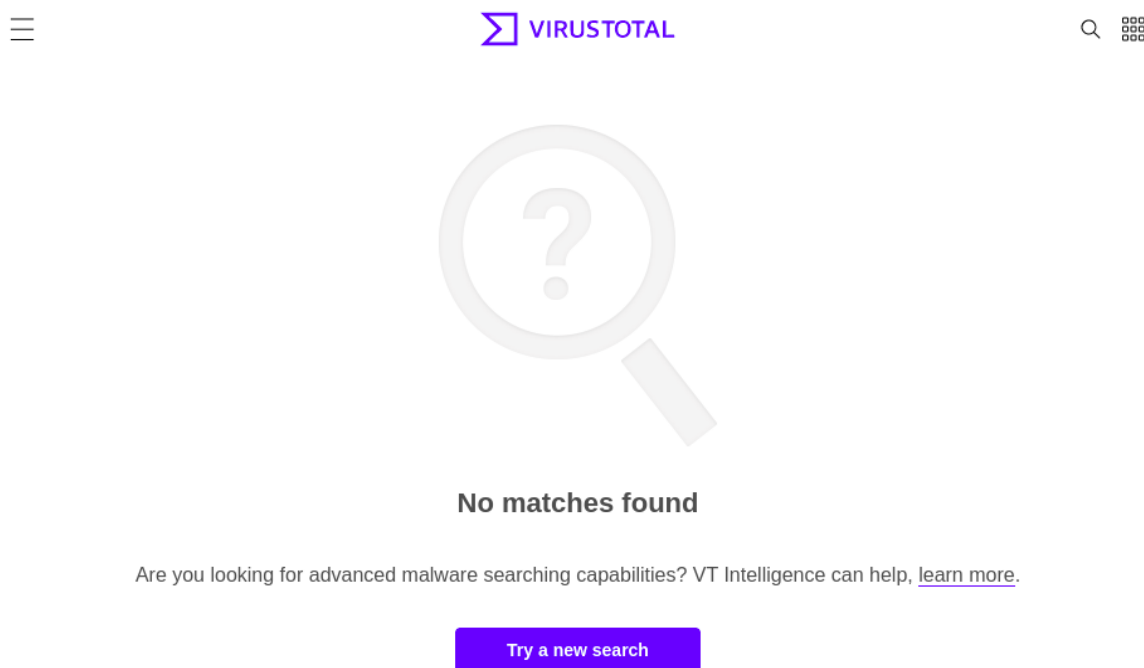
Obrázek 68. Upravený klíč registru, po spuštění malwaru 2sample01.exe

[zdroj vlastní]

Mimo spustitelné soubory vytvořil malware ještě textové soubory, které mají náhodný název obsahující alfanumerické znaky o délce 14 znaků. V těchto souborech jsou alfanumerické

znaky o různé délce. Pravděpodobně se jedná o zašifrované informace malwaru, které se nepodařilo rozšifrovat.

Následně proces malwaru 2sample01.exe vytvořil soubor s náhodným názvem a příponou .tmp, vždy u již vytvořeného nového spustitelného souboru malwaru. Pak proces 2sample01.exe střídavě čte ze spustitelného souboru malwaru a zapisuje do souboru .tmp. Po skončení zápisu dojde k přejmenování .tmp souboru na původní název souboru malwaru. Pravděpodobně dochází k šifrování části obsahu spustitelného souboru, protože dojde ke změně hodnoty hash. Tato hash je vždy unikátní a následným vyhledáním hodnoty hash ve službě Virustotal a bylo zjištěno, že takovýto nový soubor malwaru nebyl nikdy testován, viz. Obrázek 69.



Obrázek 69. Výsledek vyhledávání hashe SHA256 nově vytvořeného spustitelného souboru malwaru na webu virustotal.com

Dále malware 2sample01.exe nastaví hodnotu registru do následujícího klíče:

HKCU\Software\3160cbb73a3519433ec068ebb164ae746b853459\

REG_SZ s názvem 39468c434e86e3836df6812dded2b6d569460b15

Kde vložená hodnota REG_SZ je kódovaná pomocí base64:

```
"WyJcXFxcdmJveHN2clxcTUFMV0FSRVxcUkFUXFwyc2FtcGxlMDEu-  
ZXhlliwiQzpcXFByb2dyYW0gRmlsZXMgKHg4NilcXFdpbmRvd3MgUG9yd-  
GFibGUgRGV2aWNlc1xcVkJveFRyYXkuZXhlliwiQzpcXFByb2dyYW1EY-  
XRhXFxQYWNRyWdlIENhY2hlXFx7NjYyQTAwODgtNk-  
ZDRC00NURELTIQTctNjg2NzQwNThBRUQ1fXYxNC4zMC4zMDcwN-  
FxccGFja2FnZXNcXHZjUnVudGltZU1pbmltdW1fYW1kNjRcX-  
FZCb3hTZXJ2aWNlLmV4ZSIsIkM6XFxQcm9ncmFtIEZpbGVzICh4ODYpX-  
FxAaW5kb3dzIE1haWxcXFN5c3RlS5leGUiLCJDOlxcUHJvZ3JhbSBGa-  
Wxlc1xcV2luZG93cyBQb3J0YWJsZSBEZXZpY2VzXFxTaGVsbEV4cGVya-  
WVuY2VIb3N0LmV4ZSIsIkM6XFxQcm9ncmFtIEZpbGVzICh4ODYpX-  
FxAaW5kb3dzIFNpZGVhYXJcXFNoYXJlZCBHYWRnZXRzX-  
FxAaW5TdG9yZS5BcHAuZXhlliwiQzpcXFdpbmRvd3NcXFB-  
MQVxcTWVtb3J5IENvbXBvZXRzZXNzaW9uLmV4ZSI-  
sIkM6XFxQcm9ncmFtRGF0YVxcRG9rdW1lbnR5XFxXaW5EdW1wLmV4ZSI-  
sIkM6XFxXaW5kb3dzXFx-  
TeXN0ZW0zMlxcSXBtaVxcUnVudGltZUJyb2tldi5leGUiLCJDOlxcZG1wXFxTZ-  
WFYy2hQcm90b2NvbEhvc3QuZXhlli0="
```

Po dekódování dostaneme seznam všech spustitelných souborů malwaru.

```
. [\\"\\\\vboxsvr\\MALWARE\\RAT\\2sample01.exe", "C:\\Program Files (x86)\\Win-  
dows Portable Devices\\VBoxTray.exe", "C:\\ProgramData\\Package Ca-  
che\\{662A0088-6FCD-45DD-9EA7-  
68674058AED5}v14.30.30704\\packages\\vcRuntimeMinimum_amd64\\VBoxSer-  
vice.exe", "C:\\Program Files (x86)\\Windows Mail\\System.exe", "C:\\Program Fi-  
les\\Windows Portable Devices\\ShellExperienceHost.exe", "C:\\Program Files  
(x86)\\Windows Sidebar\\Shared Gadgets\\WinStore.App.exe", "C:\\Win-  
dows\\PLA\\Memory Compression.exe", "C:\\ProgramData\\Dokumenty\\Win-  
Dump.exe", "C:\\Windows\\System32\\Ipmi\\RuntimeBroker.exe", "C:\\tmp\\Sear-  
chProtocolHost.exe"]
```

Nakonec dojde k vytvoření úloh v „Plánovači úloh“ MS Windows, tak aby se každý spustitelný soubor malwaru spouštěl po spuštění systému a v náhodných intervalech.

Poté došlo k ukončení procesu 2sample02.exe, ostatní aktivita již byla z procesů nově vytvořených souborů malwaru, které byly spuštěny pomocí „Plánovače úloh“. Každý z těchto spuštěných nových souborů malwaru si vytvoří klíč a binární hodnoty následujících registrů:

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
EnableFileTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
EnableAutoFileTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
EnableConsoleTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
FileTracingMask
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
ConsoleTracingMask
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
MaxFileSize
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASAPI32\  
FileDirectory
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASMANS\  
EnableFileTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASMANS\  
EnableAutoFileTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASMANS\  
EnableAutoFileTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASMANS\  
EnableConsoleTracing
```

```
HKLM\SOFTWARE\Microsoft\Tracing\_RASMANS\  
ConsoleTracingMask
```

```
HKLM\SOFTWARE\Microsoft\Tracing\<>Název_souboru>_RASMANCS\  
MaxFileSize
```

```
HKLM\SOFTWARE\Microsoft\Tracing\<>Název_souboru>_RASMANCS\  
FileDirectory
```

Také byly vytvořeny textové soubory v následující cestě:

```
C:\Users\works\Local\Microsoft\CLR_v4.0\UsageLogs\<>název_souboru>.exe.log.
```

Jedná se o soubory rozhraní .NET Framework, nikoliv o log malwaru.

Síťová komunikace u vzorku malwaru 2sample01.exe proběhla na doménu universalword-press.site na port 80 TCP.

Jiná činnost tohoto vzorku malwaru nebyla nalezena.

7.3.1.2 Vzorky malwaru 2sample02.exe až 2sample05.exe

Vzorky druhé rodiny malwaru s názvy 2sample2.exe až 2sample05.exe se chovaly stejně jako vzorek 2sample01.exe. Rozdíly mezi vzorky jsou uvedeny níže.

U vzorku 2sample03.exe, 2sample04.exe a 2sample05.exe proběhl zápis všech vytvořených spustitelných souborů malwaru do registru

```
HKCU\7eb7864ddc553aaeca47ebca47ebc7fd26701d8085ad46,  
REG_SZ s názvem cc74265a6eec88d6b0ee84d08a70c17c439984e9,
```

kde byly tyto informace uloženy pomocí kódování base64. Tento klíč registru zůstal stejný i při opakovaném spuštění vzorků. Dále byla zjištěna síťová komunikace na doménu „bigboxt5.beget.tech“ na port 80 TCP.

7.3.1.3 Detekce druhé rodiny malwaru

Z provedených analýz první rodiny malwaru je možné stanovit jednoznačný indikátor kompromitace operačního systému touto rodinou malwaru.

- HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell, kdy se v tomto klíči nacházejí jiné více hodnot, než jen hodnota explorer.exe.

- HKCU\SOFTWARE\<40_znakový_klíč>\ kde je REG_SZ <40_znakový_název>=<seznam_souborů_malwaru_kódovaný_v_base64>, který po dekodování base64 zobrazí seznam všech souborů malwaru.

Pro detekci této rodiny malwaru lze využít i importního hashe:

```
imphash, f34d5f2d4577ed6d9ceec516c1f5a744
```

Tento importní hash je jedinečný pro tuto rodinu malwaru.

Pro účely detekce bylo napsáno následující YARA pravidlo, které vychází z nalezených řetězců v souborech první rodiny malwaru.

```
rule Win_DCRat {  
  
  meta:  
  
    author = "Vaclav Hess"  
  
    data = "2022-05-05"  
  
    description = "DCRat"  
  
  strings:  
  
    $mz = { 4D 5A }  
  
    $dc1 = "DCRat.Code" ascii wide  
  
    $dc2 = "DCRat-Log#" ascii wide  
  
    $logo1 = " _____  
_____" ascii wide  
  
    $s1 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdef-  
ghijklmnopqrstuvwxyz0123456789" ascii wide  
  
    $s2 = "AutoLoginUser" ascii wide  
  
    $s3 = "clipboard" ascii wide nocase  
  
    $s4 = "CreateEncryptor" ascii  
  
    $s5 = "DownloadData" ascii  
  
    $s6 = "exeToGenerate" wide  
  
    $s7 = "geoplugin_city" wide
```

```
$s8 = "HttpRequest" ascii
```

```
$s9 = "aHR0cHM6Ly9pcGluZm8uaW8vanNvbG==" ascii
```

wide

condition:

```
( $mz and ($dc1 or $dc2) and $logo1 and all of ($s*))
```

```
} [zdroj vlastní]
```

Toto YARA pravidlo kontroluje přítomnost jedinečných řetězců v testovaném souboru. Následně bylo toto pravidlo vyzkoušeno na všech vzorcích druhé rodiny malwaru, které identifikoval.

8 MNOŽINA CHARAKTERISTIK SOUČASNÉHO MALWARU, KTERÉ MAJÍ DETEKČNÍ POTENCIÁL

Statickou analýzou bylo zjištěno, že detekce rodiny malwaru na základě hodnoty hash souboru je v dnešní době nedostatečná. Každý testovaný vzorek měl vlastní hodnotu hash SHA265 souboru. Proto byly zvoleny jiné, níže popsané metody detekce rodin moderního malwaru.

Každá z testovaných rodin malwaru má jedinečné vložené řetězce, které mohou napovědět o činnosti malwaru. Na základě těchto jedinečných vložených řetězců bylo možné detekovat a identifikovat tyto rodiny malwaru. Proto byly sestaveny YARA pravidla, pro každou rodinu malwaru bylo sestaveno vlastní pravidlo. Tyto pravidla byly uvedeny v kapitole s názvem „Detekce první rodiny malwaru“ a v kapitole s názvem „Detekce druhé rodiny malwaru“. Tyto pravidla byly následně otestovány na jednotlivých vzorcích rodiny malwaru, vždy s pozitivním výsledkem, viz. Obrázek 70.

```
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample01.exe
Trojan_Doina sample01.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample02.exe
Trojan_Doina sample02.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample03.exe
Trojan_Doina sample03.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample04.exe
Trojan_Doina sample04.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample05.exe
Trojan_Doina sample05.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample06.exe
Trojan_Doina sample06.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample07.exe
Trojan_Doina sample07.exe
remnux@remnux:/media/sf_MALWARE/doina$ yara -r yara_doina.yar sample08.exe
Trojan_Doina sample08.exe
```

Obrázek 70. Ukázka detekce první rodiny malwaru pomocí sestaveného pravidla YARA [zdroj vlastní]

Na dalším obrázku je vidět, detekce a identifikace druhé rodiny malwaru pomocí sestaveného YARA pravidla.

```
remnux@remnux:/media/sf_MALWARE/RAT$ yara -r yara_rat2.yar 2sample01.exe
Win_DCRat 2sample01.exe
remnux@remnux:/media/sf_MALWARE/RAT$ yara -r yara_rat2.yar 2sample02.exe
Win_DCRat 2sample02.exe
remnux@remnux:/media/sf_MALWARE/RAT$ yara -r yara_rat2.yar 2sample03.exe
Win_DCRat 2sample03.exe
remnux@remnux:/media/sf_MALWARE/RAT$ yara -r yara_rat2.yar 2sample04.exe
Win_DCRat 2sample04.exe
remnux@remnux:/media/sf_MALWARE/RAT$ yara -r yara_rat2.yar 2sample05.exe
Win_DCRat 2sample05.exe
remnux@remnux:/media/sf_MALWARE/RAT$
```

Obrázek 71. Ukázka detekce druhé rodiny malwaru pomocí sestaveného pravidla YARA [zdroj vlastní]

Statickou analýzou obou rodin malwaru bylo zjištěno, že každá rodina malwaru má jedinečný importní hash tzv. imphash, který je společný pro každou rodinu malwaru. Importní hash první rodiny malwaru je 801793b2be29822524e8824fc3c47535 a importní hash druhé rodiny malwaru je f34d5f2d4577ed6d9ceec516c1f5a744. Tento importní hash lze získat například pomocí nástroje PEframe.

Jedinečnost hodnoty importního hash lze vztáhnou na jakoukoli rodinu malwaru a díky ní lze detekovat a identifikovat rodiny malwaru.

Statickou analýzou obou rodin malwaru bylo zjištěno, že nástrojem ssdeep, který umí vypočítat Context Triggered Piecewise Hashes (CTPH), známého jako fuzzy hashes nelze přesně určit rodinu malwaru. U vzorku 2sample01.exe byl tento fuzzy hashes velmi rozdílný oproti ostatním souborům shodné rodiny.

Na základě dat získaných z dynamických analýz obou rodin vzorku lze sestavit jednoznačné identifikátory kompromitace operačního systému rodinou malwaru. Jedná se například o unikátní klíče a hodnoty v registru MS Windows nebo jedinečné soubory anebo například jedinečné „Plánované úlohy“ MS Windows.

ZÁVĚR

Knihy, které byly součástí zadání této diplomové, jsou z pohledu začínajícího analytika malwaru velmi dobrými literárními prameny. Poskytují mnoho informací, které jsou důležité pro statickou, dynamickou analýzu vzorku malwaru, ale také disassembling/debugging souboru. Nejvíce autora diplomové práce zaujala kniha „Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware“, která obsahuje všechny potřebné informace pro analytiku malwaru. Vše je doplněno mnoha obrázky a příklady. Informace v ní uvedeny jsou napsány čtivou formou a jsou určeny pro začátečníky v daném oboru. Další v pořadí knih, které stojí za prostudování je kniha „Mastering Malware Analysis: The complete malware analyst’s guide to combating malicious software, APT, cybercrime, and IoT attacks“. Tato kniha poskytuje pokročilé informace o možnostech analýzy malwaru.

Následně v této práci byly popsány techniky statické, dynamické analýzy, disassembling/debugging a analýza výpisu obsahu paměti. Tyto postupy byly dále použity v praktické části diplomové práce.

Nástroje popsané v této diplomové části byly rozděleny do několika skupin. Na nástroje pro statickou analýzu, hexadecimální editory, nástroje pro dynamickou analýzu, nástroje pro zachycení výpisu obsahu paměti, nástroje pro analýzu výpisu paměti a disassemblery/debuggery. Z nástrojů pro statickou analýzu byl nástroj YARA největším přínosem, kdy pomocí sestavených pravidel bylo možné jednoduše detekovat a identifikovat vzorky malwaru. Z nástrojů pro dynamickou analýzu byl největším přínosem nástroj ProcDot, který z logů ostatních nástrojů použitých při dynamické analýze, dokázal jednoduše vyfiltrovat události spojené se vzorkem malwaru. Činnost tohoto malwaru poté byla graficky zobrazena, včetně časové osy.

V praktické části byly analyzovány dvě rodiny malwaru. První rodina obsahovala osm vzorků malwaru a druhá obsahovala pět vzorků malwaru. Všechny tyto vzorky malwaru měli unikátní hash SHA256.

Byla provedena statická i dynamická analýza obou rodin malwaru.

Při statické analýze první rodiny malwaru bylo zjištěno 126 unikátních řetězců, které se vyskytují ve všech vzorcích první rodiny malwaru. Z těchto řetězců bylo možné predikovat

chování této rodiny malwaru. Část těchto řetězců byla následně využita pro sestavení YARA pravidla, pomocí kterého lze detekovat a identifikovat tuto rodinu malwaru.

V první rodině malwaru byly dále nalezeny řetězce, pomocí kterých bylo možné identifikovat servery, se kterými malware komunikuje.

Nástrojem PEframe byl zjištěn importní hash, tzv. imphash, který je pro tuto rodinu malwaru jedinečný a na jeho základě lze tuto rodinu malwaru také detekovat a identifikovat.

Dynamickou analýzou první rodiny malwaru bylo zjištěno chování této rodiny v reálném čase. Všechny vzorky první rodiny malwaru vytvářely „Plánovanou úlohu“ MS Windows, pomocí které je zajištěna persistence po spuštění operačního systému.

Na základě informací z dynamické analýzy byl stanoven identifikátor kompromitace této rodiny malwaru. Díky němu lze stanovit, zda byl operační systém infikovaný touto rodinou malwaru.

Vzorky malwaru sample01.exe až sample08.exe byly postupně načteny do disassembleru/debuggeru Ghidra. Bylo zjištěno, že vzorky obsahují obfuskaci a velké množství importů a funkcí. Proto se nepodařilo touto analýzou získat smysluplné informace o těchto vzorcích malwaru.

Antivirovou kontrolou ClamAV bylo zjištěno, že ve všech vzorcích první rodiny malwaru byl detekován malware s názvem Win.Malware.Doina-9878360-0.

Statická analýza druhé rodiny malwaru přinesla informace o použitém frameworku Microsoft .NET a dále bylo nalezeno 1461 jedinečných řetězců, které se vyskytovaly ve všech vzorcích této rodiny malwaru. Tyto řetězce predikovaly činnost malwaru. Mezi těmito vloženými řetězci bylo nalezeno textové logo Dark Crystal RAT.

Z jedinečných vložených řetězců, včetně části textového loga bylo vytvořeno YARA pravidlo, pomocí kterého lze jednoduše detekovat a identifikovat tuto rodinu malwaru.

Dále byl zjištěn jedinečný importní hash, tzv. imphash všech vzorků druhé rodiny malwaru, kdy na základě této charakteristiky malwaru, lze tuto rodinu také detekovat a identifikovat.

Dynamickou analýzou bylo zjištěno chování této druhé rodiny malwaru v reálném čase. Na základě informací získaných touto analýzou byl stanoven jednoznačný identifikátor

kompromitace touto rodinou malwaru. Díky tomu je možné stanovit, zda byl operační systém infikovaný tímto malwarem.

Pomocí nástroje ILSpy byly převedeny soubory druhé rodiny malwaru do zdrojového kódu (zdrojový kód obsahuje kolem 35 000 řádků kódu), který byl následně analyzován pomocí MS Visual Studia. I přes veškerou snahu se nepodařilo touto analýzou získat smysluplné informace o těchto vzorcích malwaru.

Antivirovou kontrolou ClamAV byl ve všech vzorcích druhé rodiny malwaru detekován Win.Packed.Passwordstealera-9917697-0.

Na základě statické a dynamické analýzy byly vytvořeny jedinečné charakteristiky moderního malwaru, které mají detekční potenciál.

Detekce a identifikace současných rodin malwaru je časově náročná a klade na analytika velké nároky. Tak jak se vyvíjí malware, vyvíjí se nástroje a postupy pro jejich analýzu. V praktické části této diplomové práce bylo ověřeno, že i u současného malwaru lze nalézt různé charakteristiky, pomocí nich je možné rodiny malwaru detekovat a identifikovat.

SEZNAM POUŽITÉ LITERATURY

- [1] *Zpráva o stavu kybernetické bezpečnosti České republiky za rok 2020* [online]. NÁRODNÍ ÚŘAD PRO KYBERNETICKOU BEZPEČNOST. 39 [cit. 2022-05-02]. Dostupné z: https://www.nukib.cz/download/publikace/zpravy_o_stavu/Zprava_o_stavu_KB_2020.pdf
- [2] NORMAN, Jeremy. "Brain", the First PC Virus Epidemic, Created in Lahore, Pakistan. Exploring the History of Information and Media through Timelines [online]. 2022 [cit. 2022-05-02]. Dostupné z: <https://www.historyofinformation.com/detail.php?id=1676>
- [3] KLEYMENOV, Alexey a Amr THABET. *Mastering Malware Analysis: The complete malware analyst's guide to combating malicious software, APT, cybercrime, and IoT attacks*. Birmingham: Packt Publishing. Birmingham: Packt Publishing, 2019. ISBN 1789610788.
- [4] MONNAPPA, K A. *Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware*. Birmingham: Packt Publishing, 2018. ISBN 9781788392501.
- [5] KIM, Peter. *The Hacker Playbook 3: Practical Guide To Penetration Testing*. USA: Secure Planet, 2018. ISBN 978-1980901754.
- [6] MALIN, Cameron, Eoghan CASEY a James AQUILINA. *Malware Forensics: Investigating and Analyzing Malicious Code* [online]. United States of America: Elsevier, Inc., 30 Corporate Drive, Burlington, MA 01803, 2008 [cit. 2021-12-06]. ISBN 9780080560199. Dostupné z: <https://www.elsevier.com/books/malwareforensics/malin/978-1-59749-268-3>
- [7] MALIN, Cameron, Eoghan CASEY a James AQUILINA. *Malware Forensics Field Guide for Windows Systems: Digital Forensics Field Guides* [online]. United States of America: Elsevier, Inc., 30 Corporate Drive, Burlington, MA 01803, 2012 [cit. 2021-12-06]. ISBN 9781597494731.
- [8] *Malware. ESET Progress. Protected.* [online]. Praha: Eset, 2021 [cit. 2022-05-03]. Dostupné z: <https://www.eset.com/cz/malware/>
- [9] Upozornění na výskyt nového destruktivního malware typu wiper. *Národní úřad pro kybernetickou a informační bezpečnost* [online]. Brno: Národní úřad pro kybernetickou bezpečnost, 2022, 25.2.2022 [cit. 2022-05-03]. Dostupné z:]

- <https://www.nukib.cz/cs/infoservis/hrozby/1813-upozorneni-na-vyskyt-noveho-destruktivniho-malware-typu-wiper>
- [10] CVE MITRE. *CVE* [online]. McLean: MITRE, 2021, 12.10.2021 [cit. 2022-05-03]. Dostupné z: <https://cve.mitre.org/index.html>
- [11] *CVE* [online]. McLean: MITRE, 2022 [cit. 2022-05-03]. Dostupné z: <https://www.cve.org>
- [12] Slabina vs. zranitelnost a jaký je mezi nimi vztah. *Clever and Smart* [online]. Dolní Břežany: Miroslav Čermák, 2021, 12.7.2018 [cit. 2022-05-03]. Dostupné z: <https://www.cleverandsmart.cz/slabina-vs-zranitelnost-a-jaky-je-mezi-nimi-vztah/>
- [13] Co to je IoT?. *IoT PORT* [online]. Praha: České Radiokomunikace a.s, 2020, 27.1.2020 [cit. 2022-05-03]. Dostupné z: <https://www.iotport.cz/iot-novinky/ostatni-clanky-o-iot/co-to-je-iot>
- [14] Malware Analysis Overview. *N-ABLE* [online]. Austin, USA: N-ABLE, 2019, 13.8.2019 [cit. 2022-05-03]. Dostupné z: <https://www.n-able.com/blog/malware-analysis-steps>
- [15] Mohu v počítači používat více antivirových programů?. *ESET Progress. Protected.* [online]. Praha: Eset, 2020, 14.05.2020 [cit. 2022-05-03]. Dostupné z: <https://servis.eset.cz/Knowledgebase/Article/View/318>
- [16] Terms of Service. *Virustotal* [online]. California USA: Virustotal, 2021, 27.1.2021 [cit. 2022-05-03]. Dostupné z: <https://support.virustotal.com/hc/en-us/articles/115002145529-Terms-of-Service>
- [17] An In-Depth Look into the Win32 Portable Executable File Format. *MSDN* [online]. Redmont, USA: Microsoft, 2010 [cit. 2022-05-03]. Dostupné z: [https://docs.microsoft.com/en-us/previous-versions/bb985992\(v=msdn.10\)?redirectedfrom=MSDN](https://docs.microsoft.com/en-us/previous-versions/bb985992(v=msdn.10)?redirectedfrom=MSDN)
- [18] Obfuscated Strings from Malware using the FireEye Labs Obfuscated String Solver (FLOSS). *Mandiant* [online]. Alexandria, US: Mandiant, 2016 [cit. 2022-05-03]. Dostupné z: <https://www.mandiant.com/resources/automatically-extracting-obfuscated-strings>
- [19] ProGuard – obfuskace kódu v praxi. *MoroSystems Blog* [online]. Praha: MoroSystems, 2011 [cit. 2022-05-03]. Dostupné z: <https://blog.morosystems.cz/2011-08/odborne/java/proguard-obfuskace-kodu-v-praxi/>

- [20] BRAY, Shannon. *Implementing cryptography using python: br. 1*. Indianapolis: John Wiley, 2020. ISBN 9781119612216.
- [21] File(1) - Linux manual page. *Linux/UNIX system programming training* [online]. Man7.org, 2022 [cit. 2022-05-04]. Dostupné z: <https://man7.org/linux/man-pages/man1/file.1.html>
- [22] *TrID - File Identifier* [online]. Marco Pontello, 2022 [cit. 2022-05-04]. Dostupné z: <https://mark0.net/soft-trid-e.html>
- [23] Yara-rules base64. *GitHub* [online]. Redmont, USA: Microsoft, 2020, 8.7.2020 [cit. 2022-05-03]. Dostupné z: <https://github.com/Yara-Rules/rules/blob/master/utils/base64.yar>
- [24] Yara-Rules. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/Yara-Rules/rules>
- [25] Detect-It-Easy. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/horsicq/Detect-It-Easy>
- [26] ExifTool. *ExifTool* [online]. ExifTool, 2022 [cit. 2022-05-04]. Dostupné z: <https://exiftool.org/#features>
- [27] MYTOOLZ. *Luigi Auremma* [online]. Luigi Auremma, 2022 [cit. 2022-05-04]. Dostupné z: <https://aluigi.altervista.org/mytoolz.htm>
- [28] Identifying almost identical files using context triggered piecewise hashing. *ScienceDirect* [online]. Alpharetta, USA: Elsevier, 2022 [cit. 2022-05-03]. Dostupné z: <https://www.sciencedirect.com/science/article/pii/S1742287606000764?via%3Dihub>
- [29] Bulk_extractor. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: https://github.com/simsong/bulk_extractor/
- [30] Manalyze. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/JusticeRage/Manalyze>
- [31] Strings(1) - Linux manual page. *Linux/UNIX system programming training* [online]. Man7.org [cit. 2022-05-04]. Dostupné z: <https://man7.org/linux/man-pages/man1/strings.1.html>
- [32] Stringsifter. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/fireeye/stringsifter>

- [33] Flare Floss. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/mandiant/flare-floss>
- [34] PEframe. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/digitalsleuth/peframe>
- [35] PE Tree. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: https://github.com/blackberry/pe_tree
- [36] PEDump. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-03]. Dostupné z: <https://github.com/zed-0xff/pedump>
- [37] Pecheck.py Version 0.7.10. *Didier Stevens* [online]. Didier Stevens, 2020 [cit. 2022-05-03]. Dostupné z: <https://blog.didierstevens.com/2020/03/15/pecheck-py-version-0-7-10/>
- [38] Base64dump. *Didier Stevens* [online]. Didier Stevens, 2020 [cit. 2022-05-03]. Dostupné z: <https://blog.didierstevens.com/2020/07/03/update-base64dump-py-version-0-0-12/>
- [39] XORSearch & XORStrings. *Didier Stevens* [online]. Didier Stevens, 2020 [cit. 2022-05-04]. Dostupné z: <https://blog.didierstevens.com/programs/xorsearch/>
- [40] Capa. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-04]. Dostupné z: <https://github.com/fireeye/capa>
- [41] Malware Initial Assesment: PEStudio. *Malware Initial Assesment* [online]. Winitor, 2017 [cit. 2022-05-04]. Dostupné z: <https://www.winitor.com/>
- [42] YARA. *Virustotal* [online]. California USA: Virustotal, 2022 [cit. 2022-05-04]. Dostupné z: <https://support.virustotal.com/hc/en-us/articles/115002178945-YARA>
- [43] Xxd - Unix, Linux Command. *Tutorials Point* [online]. Telangana, Indie: Tutorials Point, 2022 [cit. 2022-05-04]. Dostupné z: https://www.tutorialspoint.com/unix_commands/xxd.htm
- [44] WxHexEditor. *WxHexEditor* [online]. wxHexEditor, 2017 [cit. 2022-05-04]. Dostupné z: <https://www.wxhexeditor.org/home.php>
- [45] TCPDUMP. *Tcpdump* [online]. Luis MartinGarcia, 2022 [cit. 2022-05-04]. Dostupné z: <https://www.tcpdump.org/index.html>
- [46] Wireshark: Documentation. *Wireshark* [online]. Wireshark, 2022 [cit. 2022-05-04]. Dostupné z: <https://www.wireshark.org/docs/>

- [47] INetSim: Documentation. *INetSim* [online]. INetSim, 2020 [cit. 2022-05-04]. Dostupné z: <https://www.inetsim.org/documentation.html>
- [48] Accept-all-ips. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-04]. Dostupné z: <https://github.com/REMnux/distro/blob/master/files/accept-all-ips>
- [49] Process Monitor v3.89. *Microsoft Sysinternals Documentation* [online]. Redmont, USA: Microsoft [cit. 2022-05-04]. Dostupné z: <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>
- [50] Process Hacker. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-04]. Dostupné z: <https://github.com/processhacker/processhacker/>
- [51] Regshot. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-04]. Dostupné z: <https://github.com/Seabreg/Regshot>
- [52] ProcDot: Documentation. *ProcDot* [online]. Vídeň, Rakousko: ProcDot, 2022 [cit. 2022-05-04]. Dostupné z: <https://www.procdot.com/onlinedocumentation.htm>
- [53] FTK Imager. *AccessData* [online]. Beaverton, USA: AccessData, 2022 [cit. 2022-05-04]. Dostupné z: <https://accessdata.com/product-download/ftk-imager-version-4-5>
- [54] Volatility3.plugins.package. *Readthedocs.io* [online]. readthedocs.io, 2022 [cit. 2022-05-04]. Dostupné z: <https://volatility3.readthedocs.io/en/v1.0.1/volatility3.plugins.html>
- [55] Ladění a testování programů. *FIT VUT Brno* [online]. Brno: FIT VUT, 2012 [cit. 2022-05-04]. Dostupné z: <https://www.fit.vutbr.cz/~martinek/clang/debug.html>
- [56] GDB: The GNU Project Debugger: GDB Documentation. *Free the Software!* [online]. GNU, 2022 [cit. 2022-05-04]. Dostupné z: <https://www.sourceware.org/gdb/>
- [57] Cutter 2.0.5 documentation: User Documentation. *The Cutter* [online]. The Cutter Developers, 2022 [cit. 2022-05-04]. Dostupné z: <https://cutter.re/docs/user-docs.html>
- [58] NationalSecurityAgency / ghidra. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-04]. Dostupné z: <https://github.com/NationalSecurityAgency/ghidra>
- [59] IDA Help: The Interactive Disassembler Help Index. *Hex-rays* [online]. Lutych, Belgie: hex-rays, 2022 [cit. 2022-05-04]. Dostupné z: <https://hex-rays.com/products/ida/support/idadoc/>

- [60] ILSpy. *GitHub* [online]. Redmont, USA: Microsoft, 2022 [cit. 2022-05-06]. Dostupné z: <https://github.com/icsharpcode/ILSpy>
- [61] BEGGS, Robert W. *Mastering Kali Linux for Advanced Penetration Testing* [online]. Birmingham, UK: Packt Publishing, 2014 [cit. 2022-05-05]. ISBN 9781782163121. Dostupné z: <https://subscription.packtpub.com/product/networking-and-servers/9781782163121>
- [62] Masm32 - Úvod. *Programujte.com* [online]. 2005 [cit. 2022-05-05]. Dostupné z: <http://programujte.com/clanek/2005110702-masm32-uvod/>
- [63] Background Activity Moderator Driver - Windows 10 Service. *Batcmd* [online]. 2022 [cit. 2022-05-05]. Dostupné z: <http://batcmd.com/windows/10/services/bam/>
- [64] Dark Crystal RAT. *Logshop: Crypto Market* [online]. Logshop [cit. 2022-05-05]. Dostupné z: <https://logshop.top/dark-crystal-rat-stiler/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

NÚKIB	Národní úřad pro kybernetickou a informační bezpečnost
Dos/DDoS	Denial of Service/Distributed Denial of Service.
DOS	Disk Operating Systém.
FAT	File Allocation Table.
PE	Portable Execution
CVE	Common Vulnerabilities and Exposures.
IoT	Internet of Thing
RAM	Random Access Memory
GNU	GNU's Not Unix!
C&C	Command and Control server
ASCII	American Standard Code for Information Interchange.
URL	Uniform Resource Locator
CTPH	Context Triggered Piecewise Hashes.
SMTP	Simple Mail Transfer Protocol
SMTPS	Simple Mail Transfer Protocol Secure
IP	Internet Protocol
IPSec	IP security
ISAKMP	Internet Security Association and Key Management Protocol
SSL	Secure Sockets Layer
TLS	Transport Layer Security
WPA	Wi-Fi Protected Access
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
html	Hypertext Markup Language

ESMTP	Extended Simple Mail Transfer Protocol
POP3	Post Office Protocol
POP3S	Post Office Protocol Secure
DNS	Domain Name System
FTP	File Transfer Protocol
SFTP	Secure FTP
FTPS	FTP over SSL/TLS
IRC	Internet Relay Chat
NTP	Network Time Protocol
GLP	General Public License
LGPL	Lesser General Public License
MS	Microsoft
ISC	Internet Systems Consortium
FTK	Forensic Toolkit
CD	Compact Disc
DVD	Digital Versatile Disc
USB	Universal Serial Bus
NSA	National Security Agency
ELF	Executable and Linkable Format

SEZNAM OBRÁZKŮ

Obrázek 1. Počet signatur antiviru ClamAV [zdroj vlastní].....	10
Obrázek 2. Záhloví PE souboru v hexadecimálním editoru xxd [zdroj vlastní].....	23
Obrázek 3. Záhloví ELF souboru v hexadecimálním editoru xxd [zdroj vlastní]	23
Obrázek 4. Ukázka kódovni base64 [zdroj vlastní].....	32
Obrázek 5. Ukázka operace XOR textu password pomocí klíče 0x50hex v nástroji CyberChef [zdroj vlastní].....	33
Obrázek 6. Nástroj file [zdroj vlastní]	35
Obrázek 7. Nástroj TrID [zdroj vlastní].....	35
Obrázek 8. Nástroj Yara-Rules – malware trickbot [zdroj vlastní]	36
Obrázek 9. Nástroj Detect It Easy – malware HermeticWiper [zdroj vlastní]	38
Obrázek 10. Nástroj exiftool – malware HermeticWiper [zdroj vlastní]	39
Obrázek 11. Nástroj signsrch – malware WannaCry [zdroj vlastní]	40
Obrázek 12. Nástroj ssdeep – malware tricbot [zdroj vlastní].....	41
Obrázek 13. Nástroj bulk_extractor – malware HermeticWiper	41
Obrázek 14. Nástroj manalyze (část výstupu) – malware HermeticWiper [zdroj vlastní].....	42
Obrázek 15. Nástroj strings (část výstupu) – malware HermeticWiper [zdroj vlastní]	43
Obrázek 16. Nástroj stringsifter (část výstupu) – malware HermeticWiper [zdroj vlastní].....	44
Obrázek 17. Nástroj floss (část výstupu) – malware HermeticWiper [zdroj vlastní].	45
Obrázek 18. Nástroj PEframe (část výstupu) – malware HermeticWiper [zdroj vlastní]	46
Obrázek 19. Nástroj PE Tree – malware trickbot [zdroj vlastní]	47
Obrázek 20. Nástroj PEDump (část výstupu) – malware HermetiWiper [zdroj vlastní]	48
Obrázek 21. Nástroj pecheck (část výstupu) – malware HermeticWiper [zdroj vlastní]	49
Obrázek 22. Nástroj base64dump – malware tricbot [zdroj vlastní]	50
Obrázek 23. Nástroj xorsearch (část výstupu) – malware HermeticWiper [zdroj vlastní].....	51
Obrázek 24. Nástroj capa – malware HermeticWiper [zdroj vlastní].....	52

Obrázek 25. Nástroj PESTudio – malware HermeticWiper [zdroj vlastní].....	53
Obrázek 26. Hexadecimální editor xxd – malware HermeticWiper [zdroj vlastní] ...	56
Obrázek 27. Hexadecimální editor wxHexEditor – malware trickbot [zdroj vlastní]	57
Obrázek 28. Nástroj tcpdump [zdroj vlastní].....	59
Obrázek 29. Nástroj Wireshark [zdroj vlastní].....	60
Obrázek 30. Nástroj INetSim [zdroj vlastní].....	61
Obrázek 31. Nástroj Process Monitor [zdroj vlastní]	63
Obrázek 32. Nástroj Process Hacker [zdroj vlastní].....	64
Obrázek 33. Nástroj RegShot [zdroj vlastní].....	64
Obrázek 34. Nástroj ProcDot [zdroj vlastní]	65
Obrázek 35. Nástroj FTK Imager [zdroj vlastní].....	66
Obrázek 36. Framework Volatility3 modul plist [zdroj vlastní].....	67
Obrázek 37. Disassembler GDB [zdroj vlastní]	68
Obrázek 38. Disassembler cutter [zdroj vlastní].....	69
Obrázek 39. Disassembler Ghidra [zdroj vlastní].....	70
Obrázek 40. Disassembler IDA Free [zdroj vlastní].....	71
Obrázek 41. Nástroj file – první rodina malwaru [zdroj vlastní].....	76
Obrázek 42. Nástroj TrID – první rodina malwaru [zdroj vlastní].....	76
Obrázek 43. Nástroj signsrch – první rodina malwaru [zdroj vlastní].....	77
Obrázek 44. Nástroj manalyze – vzorek01.exe [zdroj vlastní].....	77
Obrázek 45. Nalezené řetězce pomocí nástroje base64dump – sample01.exe [zdroj vlastní].....	81
Obrázek 46. Nalezené informace pomocí nástroje Yara-rule – sample01.exe [zdroj vlastní].....	82
Obrázek 47. Nástroj capa – informace o souboru sample01.exe [zdroj vlastní]	83
Obrázek 48. Informace nalezené nástrojem PEframe – sample01.exe [zdroj vlastní]	84
Obrázek 49. Informace nalezené nástrojem pedump – sample01.exe [zdroj vlastní]	85
Obrázek 50. Nástroj Detect It Easy vzorek malwaru sample01.exe [zdroj vlastní] ...	86
Obrázek 51. Antivirová kontrola ClamAV [zdroj vlastní]	87
Obrázek 52. Nastavení laboratorního prostředí [zdroj vlastní].....	88

Obrázek 53. Přehled chování malwaru sample01.exe PID 5600 v nástroji ProcDot [zdroj vlastní]	90
Obrázek 54. Přehled chování malwaru sample01.exe PID 4600 v nástroji ProcDot [zdroj vlastní]	91
Obrázek 55. Plánovač úloh infikovaného počítače malwarem sample01.exe [zdroj vlastní]	92
Obrázek 56. Nástroj file – druhá rodina malwaru [zdroj vlastní]	96
Obrázek 58. Nástroj signsrch - 2sample01.exe [zdroj vlastní]	97
Obrázek 57. Nástroj TrID – druhá rodina malwaru [zdroj vlastní]	97
Obrázek 59. Nástroj signsrch – 2sample02.exe až 2sample05.exe [zdroj vlastní]	98
Obrázek 60. Část výstupu nástroje manalyze na vzorku malwaru sample01.exe [zdroj vlastní]	98
Obrázek 61. Nástroj manalyze - 2sample01.exe [zdroj vlastní]	99
Obrázek 62. Nástroj ssdeep – druhá rodina malwaru [zdroj vlastní]	99
Obrázek 63. Logo Dark Crystal RAT ve vložených řetězcích [zdroj vlastní]	101
Obrázek 64. Nástroj pedump – 2sample01.exe [zdroj vlastní]	102
Obrázek 65. Nástroj Detect It Easy – 2sample01.exe	103
Obrázek 66. Antivirová kontrola ClamAV [zdroj vlastní]	104
Obrázek 67. Nastavení laboratorního prostředí. [zdroj vlastní]	105
Obrázek 68. Upravený klíč registru, po spuštění malwaru 2sample01.exe [zdroj vlastní]	106
Obrázek 69. Výsledek vyhledávání hashe SHA256 nově vytvořeného spustitelného souboru malwaru na webu virustotal.com	107
Obrázek 70. Ukázka detekce první rodiny malwaru pomocí sestaveného pravidla YARA [zdroj vlastní]	114
Obrázek 71. Ukázka detekce druhé rodiny malwaru pomocí sestaveného pravidla YARA [zdroj vlastní]	115

SEZNAM TABULEK

Tabulka 1. Pravdivostní tabulka XOR.....	33
Tabulka 2. Počet nalezených řetězců v první rodině malwaru	79
Tabulka 3. Přehled zájmových řetězců nalezených v první rodině malwaru	80
Tabulka 4. Počet nalezených řetězců u druhé rodiny malwaru	99

SEZNAM PŘÍLOH

Příloha P I: Obsah přiloženého CD

PŘÍLOHA P I: OBSAH PŘILOŽENÉHO CD

- .
 - |— druha_rodina_malwaru
 - | |— druha_rodina_malwaru_README.txt
 - | |— druha_rodina_malwaru.zip
 - | |— dynamicka_analyza
 - | | |— 2sample01
 - | | | |— Procmon_2sample01.CSV
 - | | | |— regshot_2sample01.txt
 - | | | |— WinDump_2sample01.pcap
 - | | | |— 2sample01_01.pd
 - | | | |— 2sample01_02.pd
 - | | |— 2sample02
 - | | | |— Procdot_2sample02_01.pd
 - | | | |— Procdot_2sample02_02.pd
 - | | | |— Procmon_2sample02.CSV
 - | | | |— regshot_2sample02.txt
 - | | | |— WinDump_2sample02.pcap
 - | | |— 2sample03
 - | | | |— Procmon_2sample03.CSV
 - | | | |— regshot_2sample03.txt
 - | | | |— WinDump_2sample03.pcap
 - | | | |— 2sample03_01.pd
 - | | | |— 2sample03_02.pd
 - | | |— 2sample04
 - | | | |— Procdot_2sample04_01.pd

- | | | |— Procdot_2sample04_02.pd
- | | | |— Procmon_2sample04.CSV
- | | | |— regshot_2sample04.txt
- | | | |— WinDump_2sample04.pcap
- | | |— 2sample05
- | | |— Procmon_2sample05.CSV
- | | |— regshot_2sample05.txt
- | | |— WinDump_2sample05.pcap
- | | |— 2sample05_01.pd
- | | |— 2sample05_02.pd
- | |— staticka_analyza
- | | |— base64dump_2sample01.exe.txt
- | | |— base64dump_2sample02.exe.txt
- | | |— base64dump_2sample03.exe.txt
- | | |— base64dump_2sample04.exe.txt
- | | |— base64dump_2sample05.exe.txt
- | | |— Die_2sample01.exe.Entropy.txt
- | | |— Die_2sample01.exe.Hash.txt
- | | |— Die_2sample01.exe.Info.txt
- | | |— Die_2sample01.exe.Strings.txt
- | | |— Die_2sample02.exe.Entropy.txt
- | | |— Die_2sample02.exe.Hash.txt
- | | |— Die_2sample02.exe.Info.txt
- | | |— Die_2sample02.exe.Strings.txt
- | | |— Die_2sample03.exe.Entropy.txt
- | | |— Die_2sample03.exe.Hash.txt

- | | |— Die_2sample03.exe.Info.txt
- | | |— Die_2sample03.exe.Strings.txt
- | | |— Die_2sample04.exe.Entropy.txt
- | | |— Die_2sample04.exe.Hash.txt
- | | |— Die_2sample04.exe.Info.txt
- | | |— Die_2sample04.exe.Strings.txt
- | | |— Die_2sample05.exe.Entropy.txt
- | | |— Die_2sample05.exe.Hash.txt
- | | |— Die_2sample05.exe.Info.txt
- | | |— Die_2sample05.exe.Strings.txt
- | | |— floss_2sample_all_shoda.txt
- | | |— floss_2sample01.exe.txt
- | | |— floss_2sample02.exe.txt
- | | |— floss_2sample03.exe.txt
- | | |— floss_2sample04.exe.txt
- | | |— floss_2sample05.exe.txt
- | | |— manalyze_2sample01.exe.txt
- | | |— manalyze_2sample02.exe.txt
- | | |— manalyze_2sample03.exe.txt
- | | |— manalyze_2sample04.exe.txt
- | | |— manalyze_2sample05.exe.txt
- | | |— pedump_2sample01.exe.txt
- | | |— pedump_2sample02.exe.txt
- | | |— pedump_2sample03.exe.txt
- | | |— pedump_2sample04.exe.txt
- | | |— pedump_2sample05.exe.txt

- | | |— pedump_2sample06.exe.txt
- | | |— peframe_2sample01.exe.txt
- | | |— peframe_2sample02.exe.txt
- | | |— peframe_2sample03.exe.txt
- | | |— peframe_2sample04.exe.txt
- | | |— peframe_2sample05.exe.txt
- | | |— peframe_2sample06.exe.txt
- | | |— pecheck_2sample01.exe.txt
- | | |— pecheck_2sample02.exe.txt
- | | |— pecheck_2sample03.exe.txt
- | | |— pecheck_2sample04.exe.txt
- | | |— pecheck_2sample05.exe.txt
- | | |— pecheck_2sample06.exe.txt
- | | |— signsrch_2sample01.exe.txt
- | | |— signsrch_2sample02.exe.txt
- | | |— signsrch_2sample03.exe.txt
- | | |— signsrch_2sample04.exe.txt
- | | |— signsrch_2sample05.exe.txt
- | | |— ssdeep_2sample.txt
- | | |— ssdeep_2sample01.exe.txt
- | | |— ssdeep_2sample02.exe.txt
- | | |— ssdeep_2sample03.exe.txt
- | | |— ssdeep_2sample04.exe.txt
- | | |— ssdeep_2sample05.exe.txt
- | | |— strings_2sample01.exe.txt
- | | |— strings_2sample02.exe.txt

- | | |— strings_2sample03.exe.txt
- | | |— strings_2sample04.exe.txt
- | | |— strings_2sample05.exe.txt
- | | |— trid_2sample01.exe.txt
- | | |— trid_2sample02.exe.txt
- | | |— trid_2sample03.exe.txt
- | | |— trid_2sample04.exe.txt
- | | |— trid_2sample05.exe.txt
- | | |— xorsearch_2sample01.exe.txt
- | | |— xorsearch_2sample02.exe.txt
- | | |— xorsearch_2sample03.exe.txt
- | | |— xorsearch_2sample04.exe.txt
- | | |— xorsearch_2sample05.exe.txt
- | |— yara_rat2.yar
- |— fulltext.pdf
- |— prvni_rodina_malwaru
 - |— dynamicka_analyza
 - | |— sample01
 - | | |— procdot_sample01-01.pd
 - | | |— procdot_sample01-02.pd
 - | | |— procmon_sample01.CSV
 - | | |— reg_sample01.txt
 - | | |— windump_sample01.pcap
 - | |— sample02
 - | | |— procdot_sample02-01.pd
 - | | |— procdot_sample02-02.pd

- | | └─ procmon_sample02.CSV
- | | └─ reg_sample02.txt
- | | └─ windump_sample02.pcap
- | └─ sample03
- | | └─ procdot_sample03-01.pd
- | | └─ procdot_sample03-02.pd
- | | └─ procmon_sample03.CSV
- | | └─ reg_sample03.txt
- | | └─ windump_sample03.pcap
- | └─ sample04
- | | └─ procdot_sample04-01.pd
- | | └─ procdot_sample04-02.pd
- | | └─ procmon_sample04.CSV
- | | └─ reg_sample04.txt
- | | └─ WinDump_sample04.pcap
- | └─ sample05
- | | └─ procdot_sample05-01.pd
- | | └─ procdot_sample05-02.pd
- | | └─ procmon_sample05.CSV
- | | └─ regshot_sample05.txt
- | | └─ WinDump_sample05.pcap
- | └─ sample06
- | | └─ procmon_sample06.CSV
- | | └─ regshot_sample06.txt
- | | └─ sample06_01.pd
- | | └─ sample06_02.pd

- | | └─ WinDump_sample06.pcap
- | └─ sample07
- | | └─ procdot_sample07-01.pd
- | | └─ procdot_sample07-02.pd
- | | └─ procmon_sample07.CSV
- | | └─ reg_sample07.txt
- | | └─ windump_sample07.pcap
- | └─ sample08
- | | └─ procdot_sample08-01.pd
- | | └─ procdot_sample08_02.pd
- | | └─ procmon_sample08.CSV
- | | └─ reg_sample08.txt
- | | └─ windump_sample08.pcap
- └─ prvni_rodina_malwaru_README.txt
- └─ prvni_rodina_malwaru.zip
- └─ staticka_analyza
- | | └─ base64dump_sample01.exe.txt
- | | └─ base64dump_sample02.exe.txt
- | | └─ base64dump_sample03.exe.txt
- | | └─ base64dump_sample04.exe.txt
- | | └─ base64dump_sample05.exe.txt
- | | └─ base64dump_sample06.exe.txt
- | | └─ base64dump_sample07.exe.txt
- | | └─ base64dump_sample08.exe.txt
- | | └─ Die_sample01.exe.Entropy.txt
- | | └─ Die_sample01.exe.Hash.txt

- | └─ Die_sample01.exe.Info.txt
- | └─ Die_sample01.exe.Strings.txt
- | └─ Die_sample02.exe.Entropy.txt
- | └─ Die_sample02.exe.Hash.txt
- | └─ Die_sample02.exe.Info.txt
- | └─ Die_sample02.exe.Strings.txt
- | └─ Die_sample03.exe.Entropy.txt
- | └─ Die_sample03.exe.Hash.txt
- | └─ Die_sample03.exe.Info.txt
- | └─ Die_sample03.exe.Strings.txt
- | └─ Die_sample04.exe.Entropy.txt
- | └─ Die_sample04.exe.Hash.txt
- | └─ Die_sample04.exe.Info.txt
- | └─ Die_sample04.exe.Strings.txt
- | └─ Die_sample05.exe.Entropy.txt
- | └─ Die_sample05.exe.Hash.txt
- | └─ Die_sample05.exe.Info.txt
- | └─ Die_sample05.exe.Strings.txt
- | └─ Die_sample06.exe.Entropy.txt
- | └─ Die_sample06.exe.Hash.txt
- | └─ Die_sample06.exe.Info.txt
- | └─ Die_sample06.exe.Strings.txt
- | └─ Die_sample07.exe.Entropy.txt
- | └─ Die_sample07.exe.Hash.txt
- | └─ Die_sample07.exe.Info.txt
- | └─ Die_sample07.exe.Strings.txt

- | └─ Die_sample08.exe.Entropy.txt
- | └─ Die_sample08.exe.Hash.txt
- | └─ Die_sample08.exe.Info.txt
- | └─ Die_sample08.exe.Strings.txt
- | └─ exiftool_sample01.exe.txt
- | └─ exiftool_sample02.exe.txt
- | └─ exiftool_sample03.exe.txt
- | └─ exiftool_sample04.exe.txt
- | └─ exiftool_sample05.exe.txt
- | └─ exiftool_sample06.exe.txt
- | └─ exiftool_sample07.exe.txt
- | └─ exiftool_sample08.exe.txt
- | └─ floss_sample01.exe.txt
- | └─ floss_sample02.exe.txt
- | └─ floss_sample03.exe.txt
- | └─ floss_sample04.exe.txt
- | └─ floss_sample05.exe.txt
- | └─ floss_sample06.exe.txt
- | └─ floss_sample07.exe.txt
- | └─ floss_sample08.exe.txt
- | └─ pedump_sample01.exe.txt
- | └─ pedump_sample02.exe.txt
- | └─ pedump_sample03.exe.txt
- | └─ pedump_sample04.exe.txt
- | └─ pedump_sample05.exe.txt
- | └─ pedump_sample06.exe.txt

- | └─ pedump_sample07.exe.txt
- | └─ pedump_sample08.exe.txt
- | └─ peframe_sample01.exe.txt
- | └─ peframe_sample02.exe.txt
- | └─ peframe_sample03.exe.txt
- | └─ peframe_sample04.exe.txt
- | └─ peframe_sample05.exe.txt
- | └─ peframe_sample06.exe.txt
- | └─ peframe_sample07.exe.txt
- | └─ peframe_sample08.exe.txt
- | └─ pecheck_sample01.exe.txt
- | └─ pecheck_sample02.exe.txt
- | └─ pecheck_sample03.exe.txt
- | └─ pecheck_sample04.exe.txt
- | └─ pecheck_sample05.exe.txt
- | └─ pecheck_sample06.exe.txt
- | └─ pecheck_sample07.exe.txt
- | └─ pecheck_sample08.exe.txt
- | └─ ssdeep_sample01.exe.txt
- | └─ ssdeep_sample02.exe.txt
- | └─ ssdeep_sample03.exe.txt
- | └─ ssdeep_sample04.exe.txt
- | └─ ssdeep_sample05.exe.txt
- | └─ ssdeep_sample06.exe.txt
- | └─ ssdeep_sample07.exe.txt
- | └─ ssdeep_sample08.exe.txt

- | └─ strings_sample01.exe.txt
- | └─ strings_sample02.exe.txt
- | └─ strings_sample03.exe.txt
- | └─ strings_sample04.exe.txt
- | └─ strings_sample05.exe.txt
- | └─ strings_sample06.exe.txt
- | └─ strings_sample07.exe.txt
- | └─ strings_sample08.exe.txt
- | └─ yara-rules_sample01.exe.txt
- | └─ yara-rules_sample02.exe.txt
- | └─ yara-rules_sample03.exe.txt
- | └─ yara-rules_sample04.exe.txt
- | └─ yara-rules_sample05.exe.txt
- | └─ yara-rules_sample06.exe.txt
- | └─ yara-rules_sample07.exe.txt
- | └─ yara-rules_sample08.exe.txt
- └─ yara_doina.yar