Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

Analýza škodlivého kódu ve virtuálním prostředí

Diplomová práce

Bc. Jaroslav Kovář

Školitel: Ing. Petr Břehovský

České Budějovice 2019

Bibliografické údaje

Kovář, J., 2019: Analýza škodlivého kódu ve virtuálním prostředí. [Malware analysis in virtual environment. Mgr. Thesis, in Czech.] – 72 p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic.

Annotation:

The diploma thesis deals with analysis of malware that attempts to evade many kinds of analyses in virtual environments (so-called evasive malware). The thesis includes a designed implementation of the analysis environment, experiments and assessment of the malware analysis in virtual environment, which validates the benefits of protective measures making the analysis environment less obvious to evasive malware. An important source of inspiration for creating the basis of some of the protective measures were findings about biological viruses.

Keywords: malware, malware analysis, dynamic analysis, evasive malware, virtualization, biological viruses

Anotace:

Diplomová práce se zabývá analýzou škodlivého kódu, který se analýze ve virtuálních prostředích vyhýbá (tj. evasivním malware). Práce obsahuje navrhnutou implementaci analytického prostředí, experimenty a vyhodnocení analýzy škodlivého kódu ve virtuálním prostředí, které ověřilo přínos provedených opatření na snížení odhalitelnosti analytického prostředí ze strany evasivního malware. Významným zdrojem inspirace pro některé postupy a opatření byly poznatky o výzkumu biologických virů.

Klíčová slova: škodlivý kód, malware analýza, dynamická analýza, evasivní malware, virtualizace, biologické viry

Prohlašuji, že svoji diplomovou práci jsem vypracoval/a samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své diplomové práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích,

dne 11. 12. 2019 .....

Podpis autora práce

Poděkování

Děkuji tímto způsobem všem členům mé rodiny, Markétě Coufalové a mým přátelům za povzbuzování během vzniku diplomové práce.

Děkuji svému školiteli Ing. Petru Břehovskému za odborné vedení diplomové práce, cenné připomínky a trpělivost během tvorby.

Obsah

1. Zadání diplomové práce

Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

ZADÁVACÍ PROTOKOL MAGISTERSKÉ PRÁCE

Student: Bc. Jaroslav Kovář
(jméno, příjmení, tituly)

Obor – zaměření studia: **Aplikovaná informatika – Informační systémy a technologie**

Katedra/ústav, kde bude práce vypracovávána: Ústav aplikované informatiky

Školitel: Ing. Petr Břehovský
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Garant z PřF: .....
(jméno, příjmení, tituly, katedra – jen v případě externího školitele)

Školitel – specialista, konzultant: .....
(jméno, příjmení, tituly, u externího š. název a adresa pracoviště, telefon, fax, e-mail)

Téma magisterské práce: Analýza škodlivého kódu ve virtuálním prostředí
.....

Cíle práce:

- Vytvoření prostředí pro analýzu škodlivého kódu ve virtuálním prostředí volně navazujícím na předchozí práci a rozšíření prostředí, zejména o opatření proti evasivnímu malware.
- Některé způsoby maskování analytických nástrojů, případně prostředí, napodobují postupy známé z biologie.
- Vyhodnocení opatření analýzami škodlivého kódu a testovacími evasivními programy ve virtuálním prostředí.

Základní doporučená literatura:

HOOPES, J. Virtualization for Security. Syngress, 2009. ISBN 978-1-59749-305-5.

SIKORSKI, M., HONIG, A. Practical malware analysis. No Starch Press, 2012. ISBN 978-1-59327-290-6.

WEBSTER, R.Q., GRANOFF, A. Encyclopedia of virology. Academic Press, 1999. ISBN 978-0-122270307.

2. Úvod

Žijeme ve světě plném kybernetických hrozeb. Do roku 2020 má být až třetina organizací celého světa narušena nějakou skupinou kybernetických zločinců nebo aktivistů.[1] Útočníci své nástroje a metody dlouhodobě zdokonalují. Nakolik je potom překvapivé, když prolomí a překonají zabezpečení, které je mnohdy zastaralé, nesourodé a nekoordinované.[2]

S pronikáním ICT do stále více oblastí lidské činnosti výrazně roste škoda způsobená škodlivými kódy. Výdaje na kybernetickou bezpečnost stoupají – v roce 2017 celosvětově přesáhly 89 miliard USD.[3] Navzdory vynaloženým částkám, vlna úspěšných útoků pokračuje a zasahuje prakticky všechna odvětví hospodářství, včetně sektoru dodavatelů zabezpečení. Pro ilustraci, že se skutečně odhodlaní útočníci nezastaví před ničím, lze připomenout nejméně půl roku trvající infiltraci společnosti Kaspersky škodlivým kódem Duqu 2.0[4,5]. Ten byl společností odhalen v roce 2015 během testování nové metody detekce skrytého škodlivého kódu.[6]

V práci se při analýzách ve virtuálním prostředí zaměřuji právě na skrytý škodlivý kód, tzv. evasivní malware. Jak analyzovat škodlivý kód, který se analýze snaží vyhnout? Vytvořením nového zamaskovaného analytického prostředí. Po úvodním přehledu problematiky pokračuje práce tvorbou sandboxu k analýze škodlivého kódu ve virtuálním prostředí posíleném o anti-evasivní opatření. Ověřuji vedle tradičních opatření také využitelnost některých poznatků z oboru biologie pro maskování analytických nástrojů. Jedná se například o napodobení mutace a obecně o použití technik útočníka (malware) proti němu samotnému. Spolu s implementací popisuji přínos opatření, pomocí kterých dokáže prostředí čelit evasivním technikám škodlivého kódu. Vyhodnocením pomocí testovacích evasivních programů ukazuji, jak na rozdíl od obyčejného virtuálního prostředí klesá detekovatelnost maskovaného prostředí se skrytými analytickými nástroji. Následně provádím experimenty s analýzou vzorků škodlivého kódu.

---

[1] Gartner Security & Risk Management Summit 2015.
[2] THREATCONNECT. Threat Intelligence Platforms.
[3] Gartner Forecasts Worldwide Security Spending Will Reach $96 Billion in 2018, Up 8 Percent from 2017. Dostupné z: https://www.gartner.com/newsroom/id/3836563.
[4] Kaspersky Lab. Duqu 2.0: Technical Details. Dostupné z: https://web.archive.org/web/20171123190036/https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyberespionage_actor_returns.pdf.
[5] PAGANINI, Pierluigi. Duqu 2.0: The Most Sophisticated Malware Ever Seen.
[6] Použitá metoda: alpha verze anti-APT. Dostupné z: https://blog.kaspersky.com/kaspersky-statement-duqu-attack/.

2.1 Cíle práce a jejich popis

CÍL 1. Vytvoření prostředí pro analýzu škodlivého kódu ve virtuálním prostředí volně navazujícím na předchozí práci a rozšíření prostředí, zejména o opatření proti evasivnímu malware.

Vznik analytického prostředí, které aspoň částečně navazuje na bakalářskou práci, adaptovaného na analýzu škodlivého kódu, který svou analýzu detekuje a škodlivé aktivity skrývá. Během realizace práce je ověřována úspěšnost úprav prostředí, které mají snížit jeho detekovatelnost škodlivým kódem. Operační systém infikovatelných virtuálních strojů je Windows. Úpravy prostředí proběhnou ve dvou etapách: nejprve „technická opatření“, pak „opatření inspirovaná biologií“. Z technických opatření jsou virtuálnímu prostředí upravovány:

Ovladače, Registry, Síť, Systémový čas, Soubory, Procesy, Generování uživatelské aktivity.

Prostředí bude po nějakou dobu běžně používáno, aby vznikly stopy „opotřebování uživatelem“. Účelem opatření je omezit 4 hlavní metody, kterými malware uniká analýze ve virtuálním analytickém prostředí: Srovnávání virtuálního prostředí s fyzickým; Matení automatizovaných nástrojů; Detekce založené na časování (srovnání systém. času, malware čeká aktivitu, apod.); Obfuskace (zamaskování škodlivého kódu obtížně čitelným kódem).

Výsledné prostředí slouží IT správcům a malware analytikům k analýzám škodlivých kódů.

Vstupem analytického prostředí jsou soubory libovolného malware. Pro účely práce byla z několika testovacích evasivních programů, resp. vzorků škodlivého kódu, vytvořena sbírka splňující požadavky na kvalitu (uvedení zdroje, čas přidání vzorku). Kvalitní analýzy vzniknou dodáním vzorků formátů podporovaných prostředím (Windows PE, VBscript, PDF, Office).

Výstupem analytického prostředí jsou reporty o analyzovaném škodlivém kódu, které obsahují běžně zjišťované věci o malware (Registry, Soubory, Procesy, Síťová aktivita). Zjištění poslouží například k lepšímu porozumění o co škodlivý kód usiluje, odhalení útočníka, ladění bezpečnostních nástrojů i jako podněty pro další anti-evasivní úpravy prostředí. Provádění anti-evasivních úprav analytického prostředí bude manuální.

CÍL 2. Některé způsoby maskování analytických nástrojů, případně prostředí, napodobují postupy známé z biologie.

V práci jsou uvedeny analogie mezi počítačovými a biologickými viry. Některé způsoby maskování analytických nástrojů napodobují známé postupy z biologie jako například mutace. Zamaskování nástrojů typických pro malware analýzu slouží ke snížení detekovatelnosti analytického prostředí.

CÍL 3. Vyhodnocení opatření analýzami škodlivého kódu a testovacími evasivními programy ve virtuálním prostředí.

V práci vytvořená sbírka malware resp. testovacích evasivních programů bude použita jako referenční měřítko vyhodnocující úspěšnost úprav prostředí. Například o kolik více nebo méně evasivních technik se projevilo/detekovalo analýzu po úpravách prostředí. Zjištění nejčastějších detekčních/evasivních technik vzorků poslouží jako zpětná vazba na co se zaměřit při dalších úpravách prostředí. Proběhne postup 'testy testovacími programy – zjištění o detekcích – úpravy prostředí' a 'analýza malware – reporty z analýzy – (úpravy prostředí)'. Nejprve bude (varianta 0) klasické virtuálním prostředí bez úprav doplněno (variantou 1) technickými úpravami a zároveň budou k technickým úpravám přidána opatření inspirovaná poznatky z biologie (varianta 2). Proběhne srovnání testů detekce a jak se podařilo úpravami zmást testovací evasivní programy reprezentující vzorky malware.

2.2 Definice problému

Narůstá množství škodlivého kódu, který obchází a klame „obranu" zabývající se jeho analýzou, tzv. *evasivního malware*, jenž detekuje analytická prostředí, virtualizaci nebo analytické nástroje a poté mění, případně zcela skrývá, své škodlivé aktivity. To potenciálně vede ke zkresleným či jinak znehodnoceným reportům z malware analýzy v analytickém prostředí. Analýza ve virtuálním prostředí je výhodná[7] časově, náročností a vhodná i pro méně zkušené junior malware analytiky, ovšem nyní může být problémem její *spolehlivost*.

[7] PALKMETS, Lauri, Cosmin CIOBANU, Yonas LEGUESSE, Christos SIDIROPOULOS. ENISA – Building artifact handling and analysis environment. Dostupné z: https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/documents/building-artifact-handling-and-analysis-environment-toolset.

Evasivní malware je rozšířený. Odhadem aspoň jednu evasivní techniku obsahuje od 50 % (Minerva Labs, 2017)[8] do 70 % (Lastline, 2017)[9] malware. Navíc existuje značná rozmanitost, *diverzita*, tohoto malware, který se rychle vyvíjí, *evolvuje*, a tím je obtížné na něj virtuální analytická prostředí adaptovat, a to i jen zpětně. Nároky jsou proto vysoké, jak na analytická prostředí (požadavkem adaptace těchto prostředí na vývoj škodlivého kódu), tak na následné testování účinnosti opatření. V neposlední řadě jsou značné nároky kladeny na samotné malware analytiky.

Shrnutí, čím je vzhledem k analýze škodlivého kódu problémový evasivní malware:

- Zkreslené analýzy malware (nejen ve virtuálních prostředích).
- Rozšířenost evasivních škodlivých kódů není zanedbatelná.
- Diverzita/rychlost jeho evoluce.
- Obtížná předvídatelnost nového malware a jím aplikovaných evasivních technik.

[8] Minerva Labs Research Report: 2017 Year in Review. Dostupné z: https://l.minerva-labs.com/hubfs/Minerva 2017 Yearly Report_FINAL.pdf.
[9] Lastline Enterprise. Evasive Malware Defeats Advanced Detection Tools. Dostupné z: https://www.infosecurityeurope.com/__novadocuments/357217?v=636295319723800000.

3. Informace a její škodlivé podoby

3.1 Škodlivý kód

Škodlivý kód je český ekvivalent pro malware (zkratka z angl. *malicious software*). Škodlivý kód „počítačový virus“ definoval Fred Cohen[10] v 80. letech jeho schopností šířit se.

„We define a computer ‚virus‘ as a program that can ‚infect‘ other programs by modifying them to include a possibly evolved copy of itself. With the infection property, a virus can spread throughout a computer system or network using the authorizations of every user using it to infect their programs. Every program that gets infected may also act as a virus and thus the infection grows.“[11]

Za klíčovou vlastnost virů považoval Cohen jejich schopnost replikace a již tehdy zmínil několik dodnes platných konceptů jako například viry imitující legitimní programy nebo navrhoval obranu proti nim pomocí úprav vlastností prostředí, v kterém se viry nacházejí a jsou spouštěny. Definice, co je škodlivé není zcela přímočará, ovšem můžeme například sledovat legitimní aktivitu po určitou dobu, pozorovaná data zobecnit do modelu a ten následně porovnávat s reálným aktuálním stavem (tzv. anomaly based detection).

V roce 2016 byly systémy Windows hlavním terčem dvou třetin z odhadovaného celkového množství 600 miliónů útoků škodlivým kódem.[12] Z toho byly zasaženy téměř výhradně pouze jejich 32-bitové verze (99,7 %).[13] Četnost speciálně 64-bitového malware pro Windows v roce 2016 podle zprávy AV-TEST spíše klesala. Škodlivý kód, který funguje na 32-bitových verzích, bývá schopen úspěšně napadnout také 64-bitové verze Windows. Zřejmě proto byl v roce 2016 pouze 64-bitový škodlivý kód ještě značnou raritou (0,3 %). Softwarový ekosystém Microsoftu zasáhlo v letech 2016 i 2017 nejvíce útoků obecně a též byl napaden

[10] COHEN, Fred. Computer viruses.
[11] Tamtéž. „Definujeme počítačový ‚virus‘ jako program, který dokáže ‘nakazit‘ jiné programy tím, že je upraví tak, aby zahrnovaly jeho eventuálně zdokonalenou kopii. Schopností ‘nakazit‘ se virus dokáže šířit počítačovým systémem nebo sítí – virus využije každého oprávnění uživatelů, kteří jej použijí, aby nakazil jejich programy. Každý program, jenž byl nakažen, může také působit jako virus a tím nákaza narůstá.“
[12] Srov. v r. 2006 byly zaznamenány „jen“ necelé 3 milióny vzorků škodlivého kódu. AV-TEST. Dostupné z: https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2015-2016.pdf.
[13] AV-TEST. Dostupné z: https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2015-2016.pdf.

největším počtem vzorků škodlivého kódu ze všech dodavatelů operačních systémů.[14] Celosvětově byly v Q1 2019 podílem na trhu systémy Windows 7 a Windows 10 zastoupeny oba kolem 40 %[15], s převahou 64-bitových verzí[16]. Aktuální statistiky o škodlivém kódu potvrzují jeho nepřestávající zaměření na operační systémy Windows také v průběhu 2019.[17]

Typickým vektorem (cestou), jak malware napadá koncové uživatele je elektronická pošta: Buď je vzorek škodlivého kódu nějakým způsobem přítomen v samotné zprávě (jako příloha), anebo zpráva obsahuje škodlivý URL odkaz. V roce 2017 došlo k poklesu podílu emailů se škodlivými vzorky, ale zato prudce vzrostl podíl emailů se škodlivými odkazy a zvednul se o 10,7 procentních bodů na 12,3% z celkových více než 2 miliard emailů denně filtrovaných společností Symantec.[18]

Škodlivá aktivita (Malicious technique) je charakteristika (činnosti programu), která je obecně považována za projev škodlivého kódu. Ne všechen software uplatňující takovou techniku je ovšem nutně škodlivý.

3.2 Evasivní malware

Pojmem evasivní malware je souhrnně označován škodlivý kód, který používá techniky vyhýbání se detekci a analýze. Takový malware uniká pozornosti tím, že například skrývá své škodlivé aktivity v závislosti na prostředí (angl. *environment-aware malware*).[19] Vývoj evasivního malware stupňují závody ve zbrojení mezi autory škodlivého kódu a jeho analytiky.

Chen et al.[20] zkoumali evasivní techniky (anti-virtuální a anti-debugovací) ve více než sedmnácti tisících vzorcích obecného a speciálně zacíleného (angl. *advanced persistent threat*) škodlivého kódu shromážděného v letech 2009-2014. Autoři zjistili, že evasivní techniky, resp. anti-analytické techniky, byly během zkoumaného období postupně přítomny v čím dál více

[14] AV-TEST. Dostupné z: https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2016-2017.pdf.
[15] Desktop Windows Version Market Share Worldwide (March 2017 - March 2018). Dostupné z: http://gs.statcounter.com/windows-version-market-share/desktop/worldwide.
[16] Share of Steam users in March 2018, by operating system. Dostupné z: https://www.statista.com/statistics/265033/proportion-of-operating-systems-used-on-the-online-gaming-platform-steam/.
[17] Malware Statistics & Trends Report | AV-TEST. Dostupné z: https://www.av-test.org/en/statistics/malware/.
[18] Symantec. Dostupné z: https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf.
[19] ROYAL, Paul. Entrapment: Tricking Malware with Transparent, Scalable Malware Analysis. Black Hat 2012.
[20] XU CHEN, , Jon ANDERSEN, Z. Morley MAO, Michael BAILEY a Jose NAZARIO. Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware.

vzorcích. Rostoucí přítomnost evasivních technik pak negativně korelovala s detekovatelností vzorků antivirovými nástroji.

Ve studii[21] z roku 2018 byl sbírán škodlivý kód, který prošel několika fázemi automatické analýzy, při kterých nebylo rozhodnuto, zda je vzorek legitimní nebo škodlivý. Bylo zjištěno, že 98 % vzorků škodlivého kódu, které se dostaly až do poslední fáze testování, používalo aspoň jednu evasivní technik a z toho 32 % vzorků používalo dokonce více než 6 evasivních technik. Také bylo zjištěno, že 27 % vzorků uniklo detekci, když byly zkoumány pouze v jednom analytickém prostředí (sandboxu). Úspěšná detekce vyžadovala zkoumání v nejméně dvou prostředích, která měla různý operační systém nebo internetový prohlížeč.

3.2.1 Hlavní evasivní techniky škodlivého kódu

Anti-Debugování (Anti-Debugging)

Techniky škodlivého kódu zaměřené na narušení činnosti debuggeru[22] (případně procesu debugování). Škodlivý kód tím brání analytikům získávat o něm poznatky prostřednictvím „ladění" jeho zdrojového kódu, kdy například mohou kontrolovat a upravovat části škodlivého kódu za chodu, postupovat programem kódu řádek po řádku či sledovat jeho proměnné a vlákna.

Debugger (debugování) používají analytici v rámci tzv. reverzního inženýrství[23] škodlivého kódu (například IDA Pro[24], OllyDbg, WinDbg, r2, Immunity Debugger a jiné).

Anti-Disassemblování (Anti-Disassembling)

Techniky škodlivého kódu zaměřené na narušení činnosti disassembleru[25] (případně procesu disassemblování). Škodlivý kód tím brání či znesnadňuje analytikům v převedení jeho strojového (binárního) kódu na kód v jazyce Assembler, který je pro analytiky lépe čitelný.

[21] STEFNISSON, Siggi. Evasive Malware Now a Commodity. Dostupné z: https://www.securityweek.com/evasive-malware-now-commodity.
[22] GAJDOŠOVÁ, Markéta. Reverzní inženýrství malwaru pomůže v aktivní ochraně. Dostupné z: https://computerworld.cz/securityworld/reverzni-inzenyrstvi-malwaru-pomuze-v-aktivni-ochrane-49491.
[23] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online]. „Reverse-Engineering", s. 67.
[24] IDA Pro – at the cornerstone of IT security. Dostupné z: https://www.hex-rays.com/products/ida/ida-executive.pdf.

Mnozí analytici při reverzním inženýrství čelí technikám anti-disassemblování opět pomocí nástroje IDA Pro, který funguje také jako disassembler.[26]

Obfuskace (Obfuscation)

Techniky maskování zdrojového kódu souborů se škodlivým kódem. Tím je znesnadněna tvorba signatur („otisků“), kterými by bylo možné škodlivý vzorek identifikovat. Příklady obfuskace jsou: *base64* (vícenásobné) kódování, *gzinflate*, skrývání škodlivého kódu do více vrstev kódování nebo vyhýbání se antivirovým regex (regulární výrazy) vyhledávacím vzorům.

Anti-Virtuální (Anti-VM, Anti-Virtual Machine)

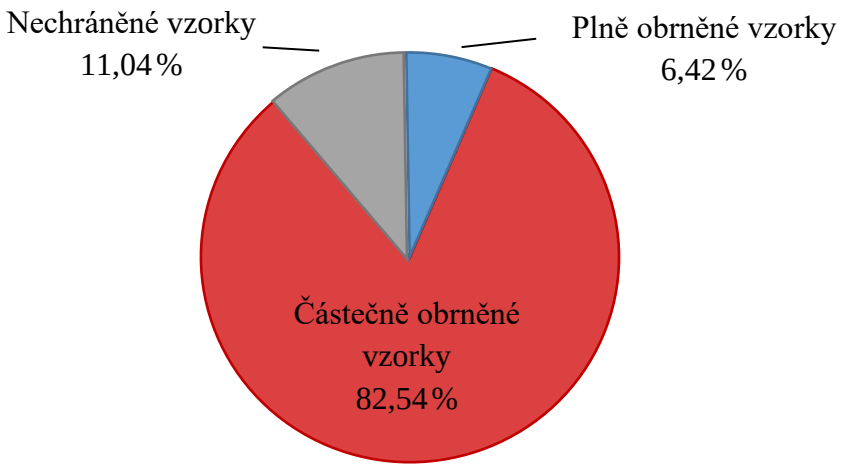
Techniky škodlivého kódu zaměřené na detekci virtuálního prostředí, vyhýbání se analýze v něm a jiné způsoby narušení činnosti (analytického) virtuálního stroje. Detekuje-li škodlivý kód přítomnost virtuálního prostředí, zamaskuje svoje škodlivé aktivity, například tím, že je vůbec nespustí. Prodloužením doby do první detekce získává škodlivý kód delší dobu k provozování nerušené škodlivé aktivity na počítačích ostatních obětí.

České ekvivalenty jsou převzaty ze *Slovníku kybernetické bezpečnosti*[27], který zároveň obsahuje český výklad výše uvedených pojmů v souvislostech.

[26] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online]. „Anti-Disassembly“, s. 360–383.
[27] JIRÁSEK, Petr, Luděk NOVÁK a Josef POŽÁR. *Výkladový slovník kybernetické bezpečnosti* [online].

3.2.2 Zastoupení evasivních technik ve škodlivém kódu

Zastoupení evasivních technik škodlivého kódu je vysoké. Branco et al.[28] vyhodnotili ve dvou studiích v letech 2012 a 2014, více než 4 milióny vzorků a více než 12 miliónů vzorků škodlivého kódu.

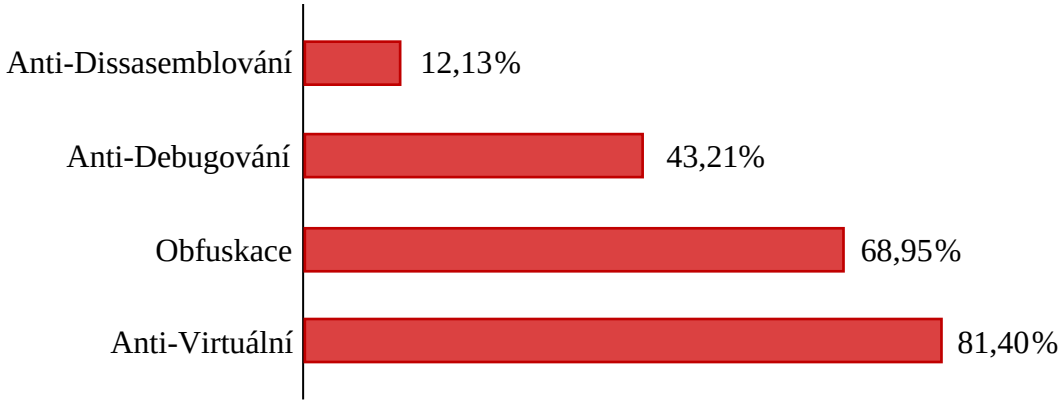


Obrázek 1: Podíl škodlivého kódu plně a částečně obrněného evasivními technikami

Vzorky škodlivého kódu obsahovaly v 89% případů alespoň jednu z hlavních evasivních technik (anti-dissasemblování, anti-debugování, obfuskace, anti-virtuální). V 6,4% případů obsahovaly vzorky škodlivého kódu dokonce všechny 4 evasivní techniky, tyto vzorky byly označeny jako tzv. *plně obrněné vzorky* (full-armored samples). Vzorky obsahující 1 až 3 evasivní techniky jsou označeny jako částečně obrněné.

[28] BRANCO, Rodrigo a Gabriel BARBOSA. Scientific but Not Academical Overview of Malware Anti-Debugging, Anti-Disassembly and Anti- VM Technologies.

Nejčastější evasivní technika použitá ve škodlivém kódu zmíněné studie je anti-virtuální technika, kterou obsahovalo 81 % vzorků[29]. Pro dynamickou analýzu škodlivého kódu, který anti-virtuální techniky obsahuje, je vhodné zamaskovat virtuální prostředí. Implementace maskovaného virtuálního prostředí je součástí této práce.



Obrázek 2: Zastoupení evasivních technik

3.2.2.1 Škodlivý kód schopný strojového učení

Škodlivý kód se schopností učit se a zdokonalovat pomocí vlastních úspěchů patří dle Mankyho z Fortinetu k závažným hrozbám, kterým bude zapotřebí od roku 2018 čelit.[30] Také Chan[31], z Fortinetu APAC, varuje, že útočníci brzy technologie umělé inteligence uplatní více. Zatímco botnety nadále představují závažné nebezpečí, považují oba jmenovaní za daleko ničivější hrozbu koncept tzv. Hivenetu poháněného umělou inteligencí. Každá jednotka hivenetu by byla složena z tzv. swarmbotů, které se mohou samostatně rozhodovat a spojit do větších autonomně jednajících ("myslících") sítí. Hivenet řízený AI bude dost chytrý na to, aby si zjistil, zda (a jakou) používá organizace obranu. Namísto neustálého útočení hrubou silou na organizaci, která nějakou kybernetickou obranou disponuje, mohou swarmboti zjistit, jestli by se nenašla snadnější cesta. Následně mohou být všechny útoky směrovány a tunelovány zpět prostřednictvím takto prolomeného slabého vstupního bodu. Nadcházející škodlivý kód generovaný a vybavený umělou inteligencí bude vnímavý vůči prostředí a schopen pozměnit sám sebe (adaptovat se). Namísto vykonávání sady předem naprogramovaných instrukcí si takový kód sám zvolí vhodné cíle (angl. Targets-of-Opportunity), posoudí jejich slabiny, vypracuje plán útoku i zamaskování stop. A dokáže se „inteligentně“ rozhodnout o tom, jaké

[29] BRANCO, Rodrigo a Gabriel BARBOSA. Scientific but Not Academical Overview of Malware Anti-Debugging, Anti-Disassembly and Anti- VM Technologies.
[30] MANKY, Derek. Cybersecurity 2018.
[31] LEE LI YING, By. Cybersecurity chief on attacks in Singapore in 2017.

informace vytěží a kdy. Aneb jak zdůrazňuje Israeli, zakladatel Illusive Networks: „Jedna z hlavních odlišností malware řízeného umělou inteligencí spočívá v jeho potenciálu prostupovat sítí (mnohem) rychleji."[32]

3.3 Analýza škodlivého kódu (malware)

Motivací proč analyzovat škodlivý kód bývá pro analytiky snaha zjistit o něm více podrobností: jak se dostal do systému, čím je charakteristický, jak doplnit obranu proti znovu napadení, odhadnout jaké představuje riziko, o co škodlivý kód usiluje a jak se ho zbavit. Navíc nabývá na významu otázka, kdo je původcem, případně autorem, škodlivého kódu. Tedy je zájem o využití analýzy k odhalení útočníka (attacker-centered threat intelligence).

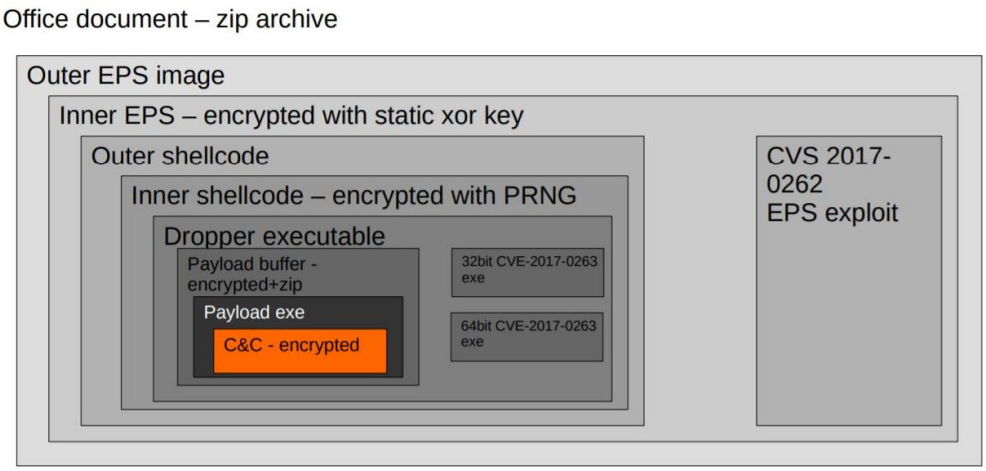
"If you were infected with malware, you'd want to know how this malware found its way past your defenses. What risks does it pose? What are the adversary's objectives? Answering these questions can take a lot of time."[33]

3.3.1 Statická analýza

Vzhledem k tomu, že statická analýza nezahrnuje spuštění škodlivého souboru, mohla by být podle Sikorskiho et al.[34] provedena v jakémkoli prostředí bez rizika vystavení systému infekci škodlivým kódem. Jako nevýhodu uvádí, že program může být obfuskován malware autorem. K obfuskaci může dojít „zabalením" škodlivého kódu pomocí packeru, přičemž je pak komprimován a nevykazuje jeho skutečné chování, dokud není dekomprimován během spuštění škodlivého kódu.[35] Extrakce (unpacking) takového kódu nemusí být vždy možná,

[32] WILLIAMS, Alice. Why your next malware infection might be AI-controlled. Dostupné z: https://venturebeat.com/2017/05/11/why-your-next-malware-infection-might-be-ai-controlled/.
[33] SZOR, Peter. *The art of computer virus research and defense*. „Pokud byste byli napadeni škodlivým kódem, budete chtít vědět, jak se škodlivý kód dostal skrze vaší obranu. Jaké riziko představuje? Jaké jsou cíle a záměry útočníka? Odpovědi na tyto otázky mohou zabrat mnoho času."
[34] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online]. „Basic Static Analysis", s. 2.
[35] Tamtéž. "Packers and Unpacking", s. 383-401.

což limituje možnosti statické analýzy. Pokud se navíc autorům škodlivého kódu podaří proměnit reverzní inženýrství na prolamování kryptografie, ztrácí statická analýza na významu:



Obrázek 3: Příklad architektury vícenásobně šifrovaného škodlivého kódu [Zdroj: cert.lv [36]]

Techniky obfuskace a šifrovaní malware jsou jedněmi z důvodů, proč se statická analýza kombinuje s dynamickou analýzou a bývá jí mnohdy nahrazena.[36]

3.3.2 Dynamická analýza

Dynamická analýza vyžaduje bezpečné prostředí pro práci. Toho se obvykle dosahuje použitím virtuálního stroje, často softwarově emulovaného operačního systému se schopností vrátit se k předchozímu stavu pomocí snímků. Taková schopnost umožňuje analytikovi spustit spustitelný soubor ve virtualizovaném operačním systému při sledování systému a vrátit se zpět do stavu před infikováním systému po dokončení analýzy. Monitorování, ke kterému dochází při spuštění malware, zahrnuje sledování síťového provozu, kontrolu přidaných či odstraněných souborů, prohlížení úprav v registru a podobně.[37] Výhodou tohoto přístupu je, že s největší pravděpodobností, pokud je spustitelný soubor škodlivý, zavede změny systému (např. se zaregistruje v autostartu). Dynamická analýza je prostředkem ke sledování těchto změn. Balené soubory (packed files) mohou být zpracovány stejně jako všechny ostatní soubory, neboť chování při rozbalení se projeví při spuštění. Rizika s použitím dynamické analýzy na rozdíl od statické analýzy zahrnují to, že systém, který je virtuálním strojem spuštěn,

[36] PODINS, Karlis. Prezentacija CERT Latvia. Dostupné z: https://cert.lv/uploads/Kibershahs/Prezentacijas/Karlis_Podins_Kibersahs-2018.pdf
[37] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online]. „Dynamic Analysis“, s.39-62.

je vystaven malware. Je možné, že malware unikne virtuálnímu počítači tím, že využije zranitelnosti ve virtuálním počítači, a proto je třeba důsledně aktualizovat virtualizační nástroj.

Tak jako malware analytici zlepšují své metody detekce malware, autoři malware zase neustále hledají nové způsoby, jak zabránit detekci a analýze jejich škodlivých kódů. Jedním ze způsobů, jak toho malware autoři dnes dosahují je použití nejrůznějších druhů evasivních technik, a proto může být vhodné čerpat inspiraci z oboru biologie, pro který je rozmanitost normou.

3.4 Biologické viry

Odhady původu biologických virů kladou jejich počátek nejméně dvě až čtyři miliardy let nazpátek[38]. Lze předpokládat, že během tak dlouhého období prošly značným vývojem. Zároveň jsou biologické viry systematicky studovány již od konce 19. století.[39] Nabízí se otázka, jak by inspirace z oboru virologie mohla obohatit analýzu škodlivého kódu?

Biologické viry představují odhadem skoro třetinu všech kompletně sekvenovaných genomů[40]. Pro úplnost jsou zde stručně uvedeny některé základní poznatky z biologické virologie. Viry jsou nebuněčné formy organismů, jsou mnohem menší a jednodušší než buňka. Jejich rozmnožování je možné jedině v hostitelské buňce, protože nemají vlastní metabolismus. Viry jsou proto popsatelné jako takzvaní nitrobuněční parazité, z nichž někteří způsobují nežádoucí onemocnění.

Viry jsou rozlišovány podle typu nukleové kyseliny na RNA viry obsahující pouze RNA, DNA viry obsahující pouze DNA a Duplo-RNA viry obsahující dvoušroubovici RNA.

Podle typu hostitele lze viry dělit na *bakteriofágy* napadající bakterie, *fytopatogenní viry* napadající rostliny, *zoopatogenní viry* napadající živočichy a *mykoviry*, které napadají houby. Speciálním typem virů jsou *retroviry*, což jsou RNA viry, které donutí hostitelskou

[38] GRANOFF, Allan. a Robert WEBSTER. *Encyclopedia of virology*.
[39] LUSTIG, A a A LEVINE. One hundred years of virology. (Zakladatelé oboru virologie zejm. – Ivanovsky, 1892; Beijerinck, 1898.)
[40] https://www.ncbi.nlm.nih.gov/genome/browse/#!/overview/

buňku přepsat RNA kód viru na DNA. Tato DNA se následně včlení do chromozomální DNA buňky.[41]

Virová infekce čili rozmnožování viru probíhá v několika fázích. Nejprve virová částice, složená z jedné nebo více molekul, přilne na povrch buňky. Poté buď celá virová částice nebo pouze nukleová kyselina se „zdrojovým kódem" viru pronikne do hostitelské buňky. Nukleová kyselina v buňce vyvolá podle svých genů syntézu bílkovin (enzymů), které jsou nutné pro pomnožení viru. Buňka poté na základě enzymu replikuje z vlastních zdrojů virovou nukleovou kyselinu a syntetizuje bílkovinu pláště viru. Sestaví se nové virové částice. Tyto částice se z buňky uvolní a mohou napadnout další buňky. Virové částice mohou napadenou buňku opustit bez poškození anebo také způsobit její zánik (tzv. lýzu buňky).

Existuje mnoho forem virové infekce. Postupné napadení biologickým virem probíhá v následujících fázích: První stadium je persistence, při které vir pronikne do buňky a přetrvá v ní, aniž by se množil. Dalším stadiem je latentní infekce, což je forma virové infekce, při které se vir množí pomalu, buňka jím v tomto stadiu není omezena. *Virogenie*[42] je forma virové infekce, kdy se virový genom začlení do genomu buňky. V dalším stadiu nazvaném transformace dochází ke změně virového genomu. Během posledního stadia virové infekce rozplétá virus cytoplazmatickou membránu a napadá další buňky. Vzniká nekrotické ložisko, které způsobí lavinové šíření infekce. Vir se například dostává do krevního oběhu a může být roznesen po celém napadeném organismu.

V neposlední řadě je významné, že patrně všechny známé biologické viry uplatňují nějaký způsob úniku před obrannou reakcí organismu hostitele. Stejně jako virové genomy potřebují molekulární nástroje k patřičné interakci s hostitelskou buňkou vedoucí k replikaci viru, tak potřebují také způsoby, jak se vyhnout odezvě hostitele a tyto způsoby bývají obdobně početné a rozmanité jako viry a hostitelé samotní.[43] Jinými slovy, viry vykazují *evasivní techniky biologického škodlivého kódu*.

[41] GRANOFF, Allan. a Robert WEBSTER. *Encyclopedia of virology*.
[42] GRANOFF, Allan. a Robert WEBSTER. *Encyclopedia of virology*.
[43] CISALPINO, Daniel a Eric BORTZ. Quora.

3.5 Analogie mezi škodlivým kódem a biologickými viry

Již Fred Cohen ve své práci[44] poukazoval na analogie[45] mezi biologickými a počítačovými viry (škodlivým kódem) a některé koncepty detekce počítačových virů navrhnul právě podle nich. V řadě charakteristik vykazují počítačové škodlivé kódy i biologické viry značnou podobnost. Existují ovšem rysy, v nichž se počítačové škodlivé kódy a biologické viry (aspoň zatím) výrazně odlišují:

Počítačové škodlivé kódy jsou dosud ve velké míře vytvářeny lidmi a existuje autor (jednotlivec nebo skupina), se znalostí zdrojového kódu.[46,47] Škodlivé kódy jsou neprostorové a digitální. Ačkoliv je jejich existence podmíněná existencí na nějakém (fyzickém) nosiči dat, můžeme na ně nahlížet jako na informaci, tedy *„strukturu uspořádávající hmotné objekty"*.[48] Ojedinělý výskyt počítačových virů, které lze označit za „užitečné" je zanedbatelný.

Biologické viry jsou vytvářeny biologickou evolucí, sekvence nových virů není známa, jsou vždy v prostorové 3D formě, jsou materiální, složené z molekul, některé viry mají užitečný vliv pro hostitele.

Shrnutí základních podobností v chování počítačového škodlivého kódu a biologických virů:

Škodlivý kód	**Biologické viry**
napadá specifické soubory, resp. programy hostitelského systému (*.exe, *.dll, *.pdf, aj.)	napadají často specifické buňky hostitelského organismu (červené krvinky, nervové buňky, aj.)
modifikuje programy systému	modifikuje genetickou informaci buňky
je parazitický, pro existenci a reprodukci potřebuje hostitelský systém (program)	jsou parazitické, pro život a reprodukci potřebují buňku
infikovaný systém vytváří kopie škodlivého kódu	infikovaná buňka vytváří kopie viru

[44] COHEN, Fred. Computer viruses.
[45] Ve smyslu existující nebo zjištěné shodnosti některých vlastností mezi netotožnými objekty, jevy ap. Dostupné z: http://scs.abz.cz/web.php/hledat?cizi_slovo=analogie
[46] Resp. existuje autor nástroje, kterým byl škodlivý kód vygenerován. Nicméně se již, zejména ve výzkumných projektech, objevují malware vzorky vygenerované umělou inteligencí a je otázkou, komu patří jejich autorství. Např. minulý rok v EndGame upravili OpenAI Framework, aby generoval malware; DARPA *CyberGrandChallenge*, apod.
[47] ANDERSON, Hyrum. Bot vs. Bot: Evading Machine Learning Malware Detection.
[48] MAREŠ, Milan. *Základy teorie informace*.

některé systémy dokáží být imunní	některé buňky mohou být imunní
vývoj a doba od vydání do aplikace záplaty (patche) na nové zranitelnosti (včetně zero-day) může zabrat týdny[49]	vývoj a doba od vydání do aplikace léku nebo vakcíny na nové virové infekce (včetně emergentních nákaz) může zabrat několik let
infikovaný systém může dlouho fungovat bezchybně	infikovaný organismus muže být dlouho bez symptomů
po spuštění se může projevit fatálně	po inkubační době se může projevit fatálně
zřídka infikuje stejný systém dvakrát	zřídka infikují stejnou buňku dvakrát
může mutovat	může mutovat
velikost typicky v intervalu 15–1500 kB[50]	DNA typického malého viru ~ 1–150 kB

Tabulka 1: Analogie mezi škodlivým kódem a biologickými viry

Ochrana před biologickými viry souvisí s „uvědoměním si“ infekce. Pro ilustraci lze uvést koncept buněčných receptorů schopných rozpoznávat vzory. Receptory PRRs existují prakticky ve všech typech buněk a mají řadu biologických funkcí – pro organismy je nejdůležitější funkce rozpoznávání patogenních molekulárních vzorů, tedy například virů. Nutno podotknout, že ne všechny biologické viry mají integrativní vlastnosti. To znamená, že jen některé viry usilují o to, modifikovat hostitelský genom a následně se do něj včlenit. Například existuje biologický vir Dengue, který se nevčleňuje do hostitelského genomu, dokonce se k hostitelské DNA ani nepřibližuje, ani nemá DNA zprostředkovatele. A přesto buňky dokáží infekci detekovat právě pomocí receptorů schopných rozpoznávat vzory. Viry mají pak zase způsoby, jak se vyhnout odezvě hostitele narušením klíčových bodů signalizace souvisejících s antivirální odezvou buňky.[51] Na určité rovině obecnosti řeší jak buňky, tak počítačové systémy analogický problém rozpoznání vzoru infekce. Navíc se ukazuje, že je možné zakódovat počítačový škodlivý kód do DNA a použít ji k útoku na zranitelnosti počítače analyzujícího sekvence této DNA.[52]

[49] NopSec State of Vulnerability Risk Management Report. Dostupné z: http://info.nopsec.com/rs/736-UGK-525/images/NopSec_StateofVulnRisk_WhitePaper_2015.pdf.
[50] POSTON, Robert. How large is malware? D. z: https://nakedsecurity.sophos.com/2010/07/27/large-piece-malware/.
[51] CISALPINO, Daniel a Eric BORTZ. Quora.
[52] TIMMER, John. Researchers encode malware in DNA, compromise DNA sequencing software.

4. Analýza a návrh

V této kapitole autor prozkoumal několik populárních malware analyzátorů, popsal sdílení dat o hrozbách (Threat Intelligence), sestavil přehled zdrojů vzorků škodlivého kódu (internetové sbírky) a sbírku programů na testování anti-evasivní úrovně prostředí. Autor navrhl vlastní řešení na analýzu škodlivého kódu ve virtuálním prostředí s anti-evasivními opatřeními.

4.1 Analýza existujících řešení

V následující části jsou uvedena hlavní současná existující řešení (Hybrid Analysis, Virus Total, JoeSandbox, Cuckoo Sandbox). Autor se s uvedenými řešeními podrobně seznámil, zjistil některé výhody i nevýhody a komunikoval s analytiky.

4.1.1 Malware skenery a analyzátory

V současné době existuje mnoho automatizovaných online malware skenerů a analyzátorů. Online analyzátory, zejména jejich veřejné bezplatné varianty obvykle zveřejňují reporty s výsledky analýzy a často samotné vzorky na internetu. To představuje značnou nevýhodu a riziko pokud soubor, který má být analyzován, obsahuje citlivé informace. Určité omezení představuje též limit na velikost nahrávatelných souborů, typicky v řádu desítek MB.[53]

4.1.1.1 Hybrid Analysis

Tento automatizovaný malware analyzátor je poskytován, jak bezplatně v omezené verzi online na doménách Hybrid-Analysis.com a Reverse-It.com, tak placené plné verzi, která je provozována též online anebo jako lokální (standalone) instalace u zákazníka. Původně německé řešení vlastní od minulého roku společnost CrowdStrike (USA). Hybrid Analysis poskytuje detailní reporty. Výhodou jsou podrobné reporty o aktivitě analyzovaného vzorku. Jen je třeba počítat s občasným výskytem false-positive anebo false-negative v reportech.

Od ledna 2018 byla autorem[54] opakovaně zaznamenána chyba, kdy Hybrid Analysis vykázal v reportu o analýze testovacímu vzorku síťovou aktivitu, ačkoliv vlastní vzorek

[53] Jednou z možností, jak online analyzátorům předložit soubor přesahující limit je extrahovat jej a poslat přípustné části. Například celou řadu souborů včetně Windows PE (.exe, .dll, aj.) lze extrahovat programem 7-Zip.
[54] Autorem této diplomové práce.

(jednoduchý .VBS skript) ve svém zdrojovém kódu o cca 50 řádcích rozhodně žádnou síťovou aktivitu naprogramovanou neměl (obsahoval jen textové if-podmínky a `msgbox`). Nabízí se vysvětlení, že k chybnému reportu Hybrid Analysis došlo tím, že do aktivit připisovaných v reportu samotnému vzorku byl nesprávně zahrnut nesouvisející šum z prostředí, v kterém byl vzorek dynamicky analyzován. Placená verze služby Hybrid Analysis začíná na $6,000 za rok.

4.1.1.2 VirusTotal

VirusTotal[55] představuje jeden z nejpopulárnějších online multi-antivirových skenerů malware a podezřelých domén. VirusTotal vznikl před 15 lety a v roce 2012 jej převzal Google. Společnost Alphabet[56] plošně používá VirusTotal ve svých produktech (Google, Gmail, aj.). Od ledna 2018 změnil VirusTotal vlastníka a byl zahrnut do Chronicle[57], nově založené dceřiné firmy Alphabetu zaměřené výhradně na kybernetickou bezpečnost. Chronicle (původně projekt v rámci Alphabet X) uvádí na svém webu záměr aplikovat umělou inteligenci a další technologie, vycházející z Google, na VirusTotal databázi vzorků.[58]

VirusTotal má zřejmě nejrozsáhlejší databázi malware vzorků indexovaných hash řetězci a tagy. Oficiální web VirusTotal uvádí, že databáze obsahuje více než 1,5 miliardy vzorků a další 2 miliardy síťových položek (URL, IP adres, domén a souvislostí jako například passive DNS). VirusTotal používá pro dynamickou malware analýzu zejména řešení Cuckoo Sandbox a čínský sandbox Tencent HABO.[59] Bezplatně VirusTotal skenuje pouze soubory o velikosti do 64 MB. Pro účely této práce není veřejné API rozhraní s omezeným přístupem do VirusTotal databáze příliš využitelné. Placené privátní API by vedle stahování více vzorků umožnilo pokročilé vyhledávání, přístup k platformě VirusTotal Intelligence zobrazující trendy v malware podložené daty z VirusTotal databáze, (praktickou) absenci limitu na API requesty a další výhody.

4.1.1.3 JoeSandbox

Online analyzátor malware umožňuje automatizovanou analýzu, jak na virtuálních strojích, tak na fyzických strojích (bare-metal analysis). V neplacené variantě umožňuje

[55] VirtusTotal. Dostupné z: https://www.virustotal.com.
[56] Alphabet. Dostupné z: https://abc.xyz/.
[57] Chronicle Security. Dostupné z: https://chronicle.security/.
[58] GILLET, Stephen. Chronicle Blog. 2018.
[59] MARTINEZ, Emiliano. Malware analysis sandbox aggregation: Welcome Tencent HABO!. Dostupné z: https://blog.virustotal.com/2017/11/malware-analysis-sandbox-aggregation.html.

JoeSandbox Cloud Basic pouze omezenou analýzu, kterou navíc podmiňuje souhlasem se zveřejněním analýzy na internetu. Pro praktické použití je mnoho funkcí jako například analýza HTTPS provozu dostupných teprve v placené variantě Cloud Pro, eventuálně v rámci 30denní zkušební verze. Provozovatelem tohoto řešení je společnost Joe Security se sídlem ve Švýcarsku. Joe Security založil v roce 2011 Stefan Bühlmann, který již 2008 vydal nástroj JoeBox. JoeBox byl ve své době jeden z prvních kernel-mode malware analyzátorů.[60] Z těchto východisek vznikl komerční online analyzátor JoeSandbox.

V dubnu 2018 vydala společnost Joe Security nový produkt JoeSandbox A1, což je standalone řešení pro malware analýzu poskytované na hardware NUC vybaveném Joe Sandbox variantami Desktop pro automatizovanou malware analýzu. Takové zařízení by bylo vhodné prozkoumat, ovšem výkonný ředitel Joe Security odmítnul autorovi této diplomové práce poskytnout A1 pro výzkumné účely do České republiky a navíc sdělil, že cena vysoce přesahuje zdroje investovatelné do této práce.

4.1.1.4 Cuckoo Sandbox

Jedná se o populární open-source malware analyzátor. Online služba Cuckoo Sandbox provádějící analýzy na Windows byla do konce roku 2017 provozována na doméně malwr.com.[61] Ve stejném roce vznikla varianta online Cuckoo Sandbox vyhrazená analýze Linux malware.[62] V současné době je možné Cuckoo Sandbox zprovoznit vlastnoručně jako standalone malware analyzátor lokálně na vlastním systému.

Právě kvůli značné popularitě se mnohé škodlivé kódy při detekci, vyhýbání se a maření své analýzy zaměřují cíleně na Cuckoo Sandbox. Například příspěvek Virus Bulletin 2016[63] uvádí celou řadu postupů, jak detekovat Cuckoo Sandbox. Bacurio a Low z Fortinetu zase uvádí aktuální postup z ledna 2018, jak obejít user-mode API háky, pomocí, kterých Cuckoo provádí monitorování.[64,65]

---

[60] *Joe Security - About Company* [online]. Dostupné z: https://www.joesecurity.org/company-joe-security.
[61] Alternativní http://sandbox.pikker.ee/ bývá z ČR bez proxy nedostupné (HTTP 503 Service Unavailible).
[62] Linux Cuckoo. Dostupné z: https://web.archive.org/web/20170809220228/https://linux.huntingmalware.com/.
[63] VB 2016. Dostupné: https://www.virusbulletin.com/uploads/pdf/magazine/2016/VB2016-Chailytko-Skuratovich.pdf.
[64] BACURIO, Floser, Wayne LOW. Prevalent Threats Targeting Cuckoo Sandbox Detection and Our Mitigation. Dostupné z: https://www.fortinet.com/blog/threat-research/prevalent-threats-targeting-cuckoo-sandbox-detection-and-our-mitigation.html.
[65] Navíc uvádí, že evasivní technika obcházející user-mode API háky je obvykle realizována zabalením malware pomocí Visual Basic packeru zvaného VBCrypter. Škodlivý kód ve Visual Basicu často cílí na Office dokumenty.

Autor kontaktoval hlavního vývojáře Cuckoo Sandbox Jurriaana Bremera[66], který potvrdil, že kernel-mode monitorování, které by zmíněnému obcházení zabránilo a je jako takové vůči evasiním technikám odolnější, je pouze v plánech budoucího vývoje (standardní Cuckoo verze 2.0.x je tudíž dosud zranitelná). Některé evasivní techniky z poslední doby byly určitým malware (např. Locky, Qbot, Ramdo, Cridex, Matsnu) cílené přímo proti Cuckoo Sandbox. Značnou komplikaci způsobují některé malware rodiny, které se nejen vyhýbají analýze, ale navíc generují falešnou informaci o sobě. Takové chování bylo pozorováno například u ransomware Locky a Ramdo. Cuckoo řídí analýzu pomocí in-guest agenta, resp. kontroléru (Python program agent.py), který musí běžet na infikovaném Guest virtuálním stroji a tím z principu roste prostor pro detekovatelnost.

(*Pozn.* Pro analýzy v maskovaných variantách prostředí vyvinutého v této diplomové práci bylo vyřešeno řízení bez in-guest agenta a řízení probíhá z vně virtuálního prostředí pomocí nativních funkcí VirtualBoxu.)

4.2 Threat Intelligence

Threat Intelligence představuje sdílení dat o kybernetických hrozbách (angl. Cyber Threat Intelligence, CTI), které probíhá více či méně veřejně. Mnoho dat o hrozbách, včetně těch o škodlivém kódu, je čerpáno formou tzv. zpravodajství z otevřených zdrojů (angl. Open Source Intelligence, OSINT). Některé CTI a OSINT zdroje byly rovněž použity v této práci.

Typickými zástupci CTI platforem jsou: *VirusTotal Intelligence*, *Malware Information Sharing Platform* (MISP používá organizace FIRST sdružující CSIRT/CERT týmy, další neveřejnou instanci MISP rovněž používá NATO[67]), *Talos* (Cisco)[68], *LogRhythm*, *MANTIS* (Siemens)[69], *NetWitness* (RSA), *iSight* (FireEye), *AlienVault* (od 2018 součást společnosti AT&T), *CRITS* (MITRE)[70], *RiskIQ*[71] a další uvádí včetně rozboru problematiky CTI dokument

[66] Dne 30. listopadu 2018 přes IRC kanál #cuckoosandbox.
[67] Malware Information Sharing Platform - NATO. Dostupné z: https://www.ncia.nato.int/Documents/Agency%20publications/Malware%20Information%20Sharing%20Platform%20(MISP).pdf#page=2.
[68] Cisco. Talos. Dostupné z: https://talosintelligence.com a https://talosintelligence.com/software.
[69] GROBAUER, Bernd. Siemens. MANTIS. Dostupné z: https://github.com/siemens/django-mantis
[70] MITRE. CRITS. Dostupné z: https://crits.github.io/
[71] RiskIQ PassiveTotal Threat Investigation Platform. Dostupné z: https://www.riskiq.com/products/passivetotal/.

ENISA[72]. Mnohá zahraničních řešení sdílejí informace ve formátech jako STIX/MAEC, OpenIOC a MISP. Více specializovanými zdroji (angl. feeds) jsou například *ShadowServer*[73] nebo databáze *PassiveDNS*. V českém prostředí vzniká jednak společný CTI projekt[74] organizací CZ.NIC a CESNET finančně podpořený MV ČR a garantovaný NÚKIB, a jednak rovněž MV ČR finančně podpořený[75] běžící pětiletý projekt *PROKI*[76], který systémem IntelMQ agreguje řadu zdrojů dat a zachycené IP vyhodnocuje například pomocí VirusTotalu nebo passiveDNS. Dále CESNET vyvíjí jako člen EU Horizon 2020 projektu Protective CTI platformu *NERD*, která škodlivým doménám vypočítává reputační skóre. Platforma NERD, stejně jako další moduly bezpečnostní CESNET e-Infrastruktury Warden[77] a Mentat[78] nebo jejich nadstavba SABU[79], používá ke sdílení informací o hrozbách formát IDEA[80].

4.3 Zdroje vstupních dat

V této kapitole je sestavena sbírka vzorků škodlivého kódu, sbírka vzorků legitimního kódu a sbírka vzorků na testování anti-evasion úrovně virtuálního prostředí.

4.3.1 Zdroje a sbírky vzorků škodlivého kódu

Některé zdroje vzorků jsou poskytovány ke stažení veřejně a bezplatně, například:

- **MalShare** – http://www.malshare.com/ – 2 milióny vzorků[81]

```
API klíč na bezplatné stažení 1000 vzorků denně je vygenerován po registraci
nového uživatele na webu MalShare.[82]
```

- **Malc0de** – http://malc0de.com/database – seznam škodlivých URL odkazů na malware

[72] Exploring the opportunities and limitations of current Threat Intelligence Platforms — ENISA. Dostupné z: https://www.enisa.europa.eu/publications/exploring-the-opportunities-and-limitations-of-current-threat-intelligence-platforms.
[73] Shadowserver Statistics. Dostupné z: https://www.shadowserver.org/wiki/pmwiki.php/Stats/Statistics.
[74] CESNET a CZ.NIC vybudují kybernetický bezpečnostní systém CTI. Dostupné z: https://www.cesnet.cz/sdruzeni/zpravy/tiskove-zpravy/cesnet-a-cz-nic-vybuduji-kyberneticky-bezpecnostni-system-cti/.
[75] RVVI PROKI. https://www.rvvi.cz/cep?s=rozsirene-vyhledavani&ss=detail&n=8&h=VI20152020026
[76] PRedikce a Ochrana před Kybernetickými Incidenty (PROKI). Dostupné z: https://csirt.cz/page/3586/proki/
[77] KÁCHA, Pavel, Michal KOSTĚNEC a Andrea KROPÁČOVÁ. Warden 3: Internet Threat Sharing Platform. Int. Journal of Computers. Dostupné z: http://www.naun.org/main/NAUN/computers/2016/a302007-172.pdf.
[78] KÁCHA, Pavel. Warden, Mentat. Dostupné z: https://www.cesnet.cz/wp-content/uploads/2017/02/warden-mentat.pdf.
[79] KROPÁČOVÁ, Andrea. SABU. Dostupné z: https://www.cesnet.cz/wp-content/uploads/2016/02/kropacova-sabu.pdf
[80] KÁCHA, Pavel. IDEA: Security Event Taxonomy Mapping. Circuits, Systems, Communications and Computers.
[81] Správce MalShare na webu uvádí, že ne všechny vzorky jsou nutně škodlivé. Některé prověří analýzy v této práci.
[82] MalShare API stáhne vzorky i vcelku za den. Dostupné z: https://github.com/Malshare/MalShare-Toolkit

- **Malware Capture Facility Project** – https://mcfp.felk.cvut.cz/publicDatasets/
- DasMalwerk – http://dasmalwerk.eu/
- theZoo MalwareDB – http://thezoo.morirt.com/

Některé další jsou dostupné po žádosti a opět bezplatně. Zejména bývá často uváděna možnost bezplatného zpřístupnění pro akademický výzkum. Aspoň u této práce probíhalo schválení nejdéle měsíc od zahájení komunikace s provozovateli. Následující zdroje byly zdarma zpřístupněny autorovi práce po schválení žádosti pro účely (akademického) výzkumu:

- **VirusShare** – https://virusshare.com/ – Akademický přístup ✓
- **HybridAnalysis** – https://www.hybrid-analysis.com/ – Akademický přístup ✓
- **MISP** (Malware Information Sharing Platform) – https://www.circl.lu/misp/
  - https://mispriv.circl.lu/ – aktuální instance MISP – Akademický přístup ✓
- VxHeaven – http://83.133.184.251/virensimulation.org/ – Záloha webu vxheaven.org ✓

Výše uvedené zdroje poskytují množství vzorků dalece přesahující rozsah této práce.

Zdroje vzorků placené nebo jinak komplikované, jako například VirusTotal vyžadující závazek akademického pracoviště sdílet všechny vzorky, proto mohly být ponechány stranou.

4.3.2 Správa sbírky (repozitáře) vzorků škodlivého kódu

Návrh struktury, způsobu uložení a používání sbírky škodlivých vzorků počítá s využitím existujících řešeních pro správu vzorků (např. VIPER Framework[83], sqlite3 databáze theZoo[84], aj.) dle jejich kompatibility se standardy pro malware reporty (zejména MAEC, MISP, IDEA).

Vcelku častým jevem jsou nekompatibility způsobené tím, že celá řada organizací používá vlastní schéma na repozitář vzorků. To ztěžuje efektivní sdílení analytických informací, přestože obě strany chtějí sdílet analýzy a další data. V takové situaci se jako řešení nabízí použití standardu, MAEC[85]. Schéma MAEC je vhodné jako obecný, standardizovaný, formát na zprostředkování mapování mezi různorodými repozitáři škodlivého kódu tak, aby bylo možné sdílet analytické informace uložené v těchto nesourodých repozitářích. Například výstup reportovacího Python modulu Cuckoo Sandbox 2.x je nativně ve formátu MAEC 5.0.[86]

[83] VIPER – Binary Analysis and Management Framework. Dostupné z: https://github.com/viper-framework/viper.
[84] Dostupné z: http://thezoo.morirt.com/.
[85] MAEC 5.0. Dostupné z: http://maecproject.github.io/releases/5.0/.
[86] MAEC 5.0 Cuckoo Report Module. Dostupné z: https://github.com/MAECProject/cuckoo/blob/maec5.0-cuckoo2.0/cuckoo/reporting/maecreport.py.

4.3.3 Sbírka vzorků legitimního kódu

Strukturou je sbírka podobná předchozí s tím, že tentokrát obsahuje legitimní programy. Její oblast použití je, jak případné další otestování (např. false-positive rate) již uvedených[87] malware analyzátorů, tak obdobné otestování postupu analýzy v prostředí této práce.

> 7z1801.exe (7-Zip), ~WRA0003.wbk (dočasný soubor Microsoft Word), calc.exe, chrome.exe, cmd.exe, libvlccore.dll (knihovna VLC Player), readerdc_install.exe (Adobe Reader), VZ_JB.ppt, winlaunch.bat (spustí doplněk SQL Lite v Exploreru) ...

4.3.4 Sbírka na testování anti-evasion úrovně virtuálních prostředí

Jedná se o neškodné testovací programy napodobující detekční techniky malware. Tyto programy umožňují ověřit detekovatelnost virtuálního prostředí a detekovatelnost přítomnosti analytických nástrojů.

> al-khaser_x86.exe, al-khaser_x64.exe – https://github.com/LordNoteworthy/al-khaser
>
> InviZzzibleSandboxEvasion.exe – https://github.com/CheckPointSW/InviZzzible
>
> pafish.exe – https://github.com/a0rtega/pafish (Paranoid Fish)
>
> SEMS.exe – https://github.com/AlicanAkyol/sems

Testovací sbírka byla následně předložena populárnímu online malware analyzátoru Hybrid Analysis. Testovací programy jsou navrženy tak, že výsledky jejich zjištění během analýzy jsou obvykle čitelné v reportu analyzátoru, kterému byl vzorek standardně předložen (například pafish.exe a sems.exe vytvoří soubory s názvy detekovaných příznaků analýzy). Tím byla pro účely této práce v roce 2018 prakticky otestována aktuální anti-evasion úroveň automatizovaného online malware analyzátoru Hybrid Analysis.

[87] Viz §4.1.1.

Autor zkoumal detekovatelnost populárního prostředí a v něm probíhající analýzy:

	Hybrid Analysis (https://www.hybrid-analysis.com)
al-khaser_x86	**Analýza detekována** ✓ ▪ *Počet **CPU jader** < **2*** ▪ *CPUID hypervisor vendor odpovídá VirtualBoxu* ▪ ***MAC vendor** = 0A:00:27 odpovídá Hybrid Analysis* ▪ *Sériové číslo v BIOSu, model (VirtualBox) a výrobce počítače (Oracle) (zjištěno pomocí WMI[88,89])* ▪ *NTEventLog obsahuje „vboxvideo“ (dle WMI)* ▪ *VirtualBox firmware v ACPI tabulkách*[90] HA Report 6b6de10...
al-khaser_x64	**Analýza detekována** ✓ ▪ *Obdobná zjištění jako výše al-khaser_x86* ▪ *MAC, Sériové číslo BIOSu, model a výrobce počítače...* ▪ *VirtualBox firmware ACPI* HA Report 33a7ea4...
InviZzzibleSE	**Analýza nedetekována** × HA Report 93674c7...
Pafish	**Analýza detekována** ✓ ▪ *CPUID hypervisor vendor odpovídá VirtualBoxu* ▪ *CPU časování = VM*[91,92]

[88] Windows Management Instrumentation (WMI) poskytuje prostředky pro správu dat a řízení operačních systémů založených na Windows. Více o WMI například zde: STEMP, Greg, Dean TSALTAS, Bob WELLS a Ethan WILANSKY. WMI Scripting Primer: Part 1.
[89] Například sériové číslo z BIOSu počítače lze vypsat z příkazové řádky Windows: C:\wmic bios get serialnumber. V případě testovacího programu al-khaser jsou dotazovány mnohé WMI třídy, například W32_ComputerSystem.
[90] VirtualBox User Manual – Chapter 3. System Settings. Advanced Configuration and Power Interface (ACPI). „VirtualBox poskytuje ACPI Guest operačnímu systému defaultně. ACPI je standard umožňující operačním systémům rozpoznat hardware, konfigurovat základní desku (motherboard) i jiná zařízení a spravovat napájení.“ Dostupné z: https://www.virtualbox.org/manual/ch03.html
[91] CPU virtuálního stroje lze detekovat rozdíly v časech opakovaného vykonání instrukce RDTSC oproti fyzickému CPU. Více o instrukci RDTSC (Read Timestamp-Counter) v manuálu Intel 64 a IA-32 architekury pro vývojáře: https://software.intel.com/sites/default/files/managed/39/c5/325462-sdm-vol-1-2abcd-3abcd.pdf#page=1667
[92] KANKOWSKI, Peter. Performance measurements with RDTSC. Dostupné z: https://www.strchr.com/performance_measurements_with_rdtsc.

	▪ *Neaktivní myš uživatele* HA Report 2180f4a...
SEMS[93]	**Analýza detekována** ✓ ▪ *Počet CPU jader = 1* ▪ *Pevný disk <= 60 GB* ▪ *Registry klíče VirtualBox*[94] HA Report 59bb9d0...

Tabulka 2: Reporty malware analyzátoru Hybrid Analysis o testovací anti-evasion sbírce

Na populární analytická prostředí bývá malware cíleně připraven dopředu. Pro ilustraci je vhodné uvést, že na průzkum například Hybrid Analysis zevnitř postačí skripty .bat, .vbs o 5 až 10 řádcích. Skript autor práce předložil zmíněnému online analyzátoru. Ve veřejném reportu tím vyšlo najevo, že zmíněný malware analyzátor je virtualizován pomocí VirtualBoxu, název virtuálního počítače je PSPUBWS=PC, má přidělenou IP tvaru 192.168.56.x (typické pro VirtualBox) a na strojích určených k infikování je nainstalován Wireshark. Stačilo analyzátoru předložit skripty s příkazy "cmd /c set, výpisem motherboardu pomocí "wmic" nebo ipconfig /all. Mnozí tvůrci malware mohou podobným způsobem snadno zjistit charakteristiky populárních malware analyzátorů a cíleně upravit své škodlivé kódy, aby tato řešení detekovala a klamala.

Proti analýze ve zcela novém prostředí, například sandboxu Javeman vytvořeném v této práci, se však tvůrce malware z principu nemá šanci cíleně připravit.

[93] https://github.com/AlicanAkyol/sems/blob/master/cuckoo_detection/Functions.cpp
[94] Například Windows Registry klíč „System\\CurrentControlSet\\Enum\\IDE\\DiskVBOX_HARDDISK“

4.4 Návrh analýzy škodlivého kódu v prostředí rozšířeném o anti-evasion opatření

Na jedné straně jsou škodlivé kódy útočníků záměrně různorodé (obfuskace, polymorfní a metamorfní varianty, rychlé originální inovace), na druhé straně jsou nástroje obránců výrazně stejnorodější (oproti malware) až uniformní, jak celkově nižším počtem variant nástrojů, tak zejména pokud jde o jednotlivé instance stejného programu, kdy má množství kopií stejný otisk. Toho dlouhodobě malware zneužívá. Jakým způsobem autor navrhuje tomu čelit?

Pokud je úspěch evasivního malware založen na jeho úspěšné schopnosti detekovat přítomnost obranných nástrojů svých obětí, nabízí se myšlenka použít proti malware jeho vlastní zbraně a detekci mu ztížit. Svým způsobem se karta obrací a nyní obě strany usilují o detekci s kým a čím se potýkají. Tradiční schéma obránce detekuje malware, doplňuje zrcadlově schéma, kdy malware detekuje, na koho útočí, a zda jde o (bezbrannou) oběť anebo obránce schopného malware „vyřadit z provozu“. Odlišné prostředí poskytne nové možnosti odhalit škodlivé aktivity vzorku, který zatím unikal úspěšné analýze (stávajícími prostředími) a „vyřadit“ jej následným sdílením reportu (celého nebo tzv. Indicators-of-Compromise). Reporty z analýz pak může analytik samostatně sdílet například do platforem Threat intelligence komunitám „obránců“ (VirusTotal, AlienVault, MISP, CESNET CTI či NERD).

Bude zajímavé implementovat do obranných nástrojů obdobné techniky jakými malware brání detekci, různé mutace, polymorfismus (metamorfismus, aj.) a vyhodnotit přínos proti detekci. Popsaný návrh bude experimentálně ověřen v práci úpravami virtuálního analytického prostředí a testy funkčnosti opatření maskujících prostředí před malware na analýzách sbírky vzorků.

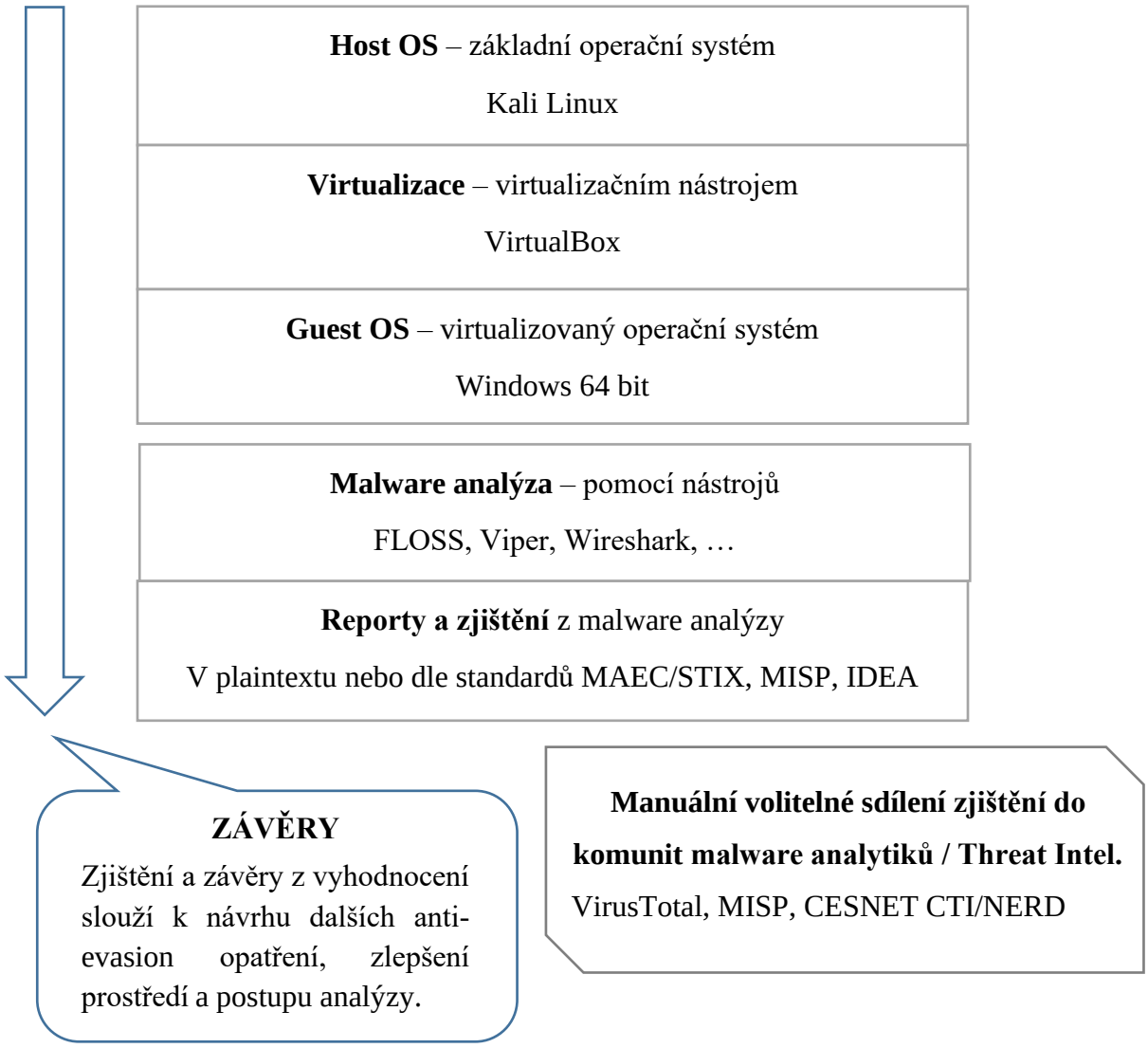
Malware analýza v prostředí bude dynamická s prvky statické analýzy. V reportech analýzy bude vedle typicky zjišťovaných vlastností (Soubory, Procesy, Síťová aktivita apod.) věnována pozornost technikám detekce virtuálního stroje prostředí („VM“) i anti-VM evasion a doporučení o spolehlivosti analýzy VM. Prototyp prostředí bude izolován od internetu a síťové služby budou simulované[95]. Proběhne (aspoň částečně automatizovaná) analýza vzorků

[95] Jako prevence vzniku škod analýzou vzorků a také, aby prostředí mohlo sloužit uživatelům, kteří na sebe nechtějí upozornit tím, že by malware vzorku dovolili připojit se na Internet a tzv. „zavolat domů“.

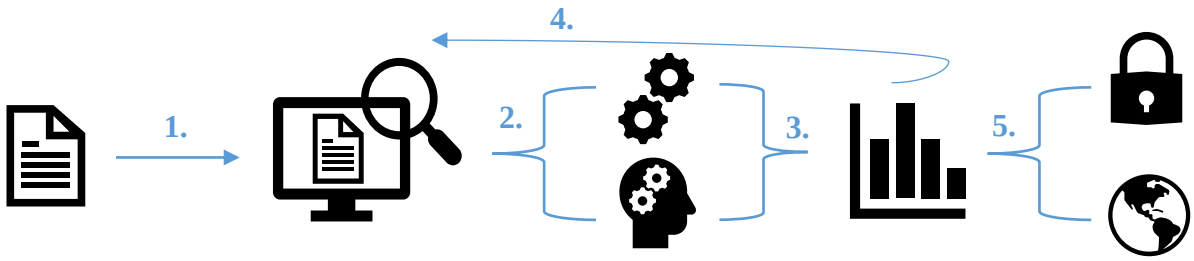
ze sbírky (v prostředí) a zjištění (základní) budou shrnuta do reportu o analýze. Reporty může analytik sdílet do Threat Intelligence v podobě blízké standardům anebo po úpravě přímo ve formátech MAEC/STIX, MISP a IDEA.

Přínosem vzniklého prostředí bude ověření úspěšnosti návrhu, jak zlepšit virtuálnímu prostředí odolnost proti detekci a evasivním technikám malware a tím potenciálně dosáhnout spolehlivějších analýz. Tím, že poznáme, kdy virtuální analýza stačí, protože vzorek neuplatňuje VM evasivní techniky, je přínosem úspora času a úsilí, které by jinak padlo na fyzickou analýzu. Podle časových možností budou provedeny experimenty s analýzou škodlivých kódů v prostředí rozšířeném o technická a bio-inspirovaná anti-evasivní opatření.

Posloupnost návrhu rozšířeného prostředí ilustruje následující schéma:



Obrázek 4: Schéma analýzy škodlivého kódu v prostředí rozšířeném o anti-evasion opatření

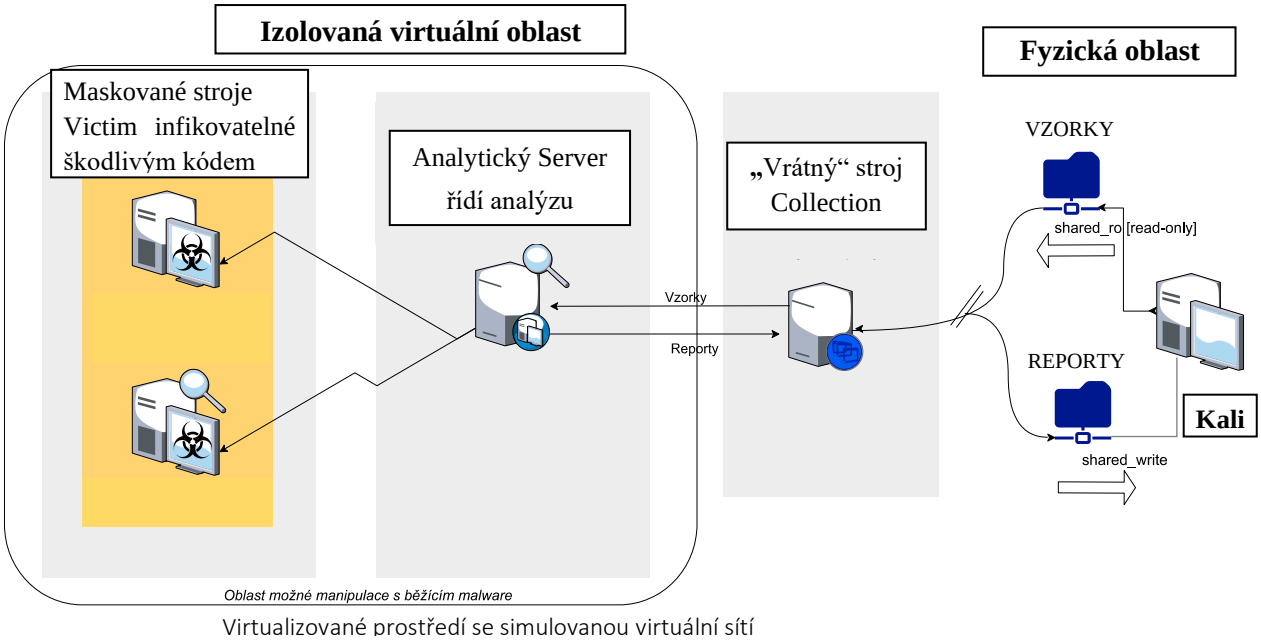


Obrázek 5: Schéma vstupu vzorku do analýzy, analýza a výstup případným sdílením zjištění

Analýza v prostředí této práce probíhá v několika zobecněných krocích (Obrázek 5): Vzorek je přesunut do analytického prostředí (1). Analýzu je provedena, buď automatizovanou nástroji, anebo manuálně, případně jejich kombinací (2). Podle zjištění z analýzy (3) se lze rozhodnout pro opakovanou analýzu vzorku (4). Výsledná zjištění pak mohou zůstat soukromá (užitečné jde-li o neveřejné informace) anebo jsou veřejně sdíleny, zejména s komunitami analytiků (5).

4.5 Shrnutí požadavků na prostředí analýzy škodlivého kódu

- **Sbírky** testovacích evasivních programů a vzorků škodlivého kódu (popsaného hash otiskem, datem přidání, atd.)
- **Rozšířené prostředí** o anti-evasion opatření s automatizovaným nebo manuálním řízením
- **Varianty** virtuálního analytického prostředí (popsané níže v tabulce Tab. 3)
  - Varianta 0. **Klasická verze** bez maskovacích anti-evasivních opatření
  - Varianty 1-2. **Anti-evasion verze** obsahují maskovací opatření spočívající v úpravě vlastností virtuálního systému, analytických nástrojů a generování uživatelské aktivity.
- **Vyhodnocení** (ne)detekovatelnosti těchto variant testovacími evasivními programy.
- **Analýza škodlivého kódu** zacílená na evasivní malware poskytne **zjištění** o jeho aktivitě, které buď zůstanou soukromé anebo mohou být sdíleny.



Obrázek 6:Schéma návrhu analytického prostředí

„#0“ GuestAdd	Nezamaskované virtuální prostředí.
	Klasický virtuální stroj s Windows 7, který obsahuje doplňky VirtualBox Guest Additions. Tato varianta demonstruje, že lze dosáhnout rychlého vytvoření stroje, ale vzniklé prostředí je poté snadno detekovatelné. Stroj Victim pouze [w7cv-0-guestadd].
„#1“ Fresh	Mírně maskované virtuální prostředí.
	Vlastní vytvořené stroje Victim instalací Windows 7 jsou na rozdíl od předchozí varianty bez Guest doplňků a s parametry HW upravenými tak, aby více odpovídaly parametrům reálného počítače. Stroje Victim [w7cv-1-fresh] [w7av-1-fresh].
„#2“ AntiEvasion	Zamaskované virtuální prostředí se zamaskovanými analytickými nástroji.
	Opět vlastní vytvořené stroje Victim s Windows 7, cíleně maskované proti detekci virtualizace a analytických nástrojů (anti-evasivními opatřeními). Určené na analýzy škodlivého kódu. Stroje Victim [w7cv-2-antievasion] [w7av-2-antievasion].

Tabulka 3: Varianty analytického prostředí dle detekovatelnosti a účelu

5. Implementace analýzy v maskovaném prostředí

Projekt maskoveného prostředí využívá verzovací systém GitLab.com a byl označen pracovně názvem JAVEMAN (Junior Adaptive Virtual Environment for Malware Analysis). Repozitář projektu GitLab.com bude pro účely obhajoby dočasně zpřístupněn jako veřejný.[96]

Virtuální prostředí obsahuje několik typů virtuálních strojů, každý se specifickým účelem:

- **Clean Victim VM** představuje „oběť“ na které budou spouštěny vzorky škodlivého kódu. Guest OS tohoto virtuálního stroje jsou Windows 7 64-bit. Stroj záměrně postrádá typické analytické nástroje kromě těch, které jsou standardní součástí Windows.
- **Analytic Server VM** simuluje oběti síťové služby, monitoruje veškerou komunikaci s ní a obsahuje zejména statické analytické nástroje i nástroje na vizualizaci logů z analýz. Vzorky malware přijímá z Collection VM. Hostovaným OS tohoto stroje je Remnux v6.
- **Collection VM** představuje bránu prostředí a slouží k distribuci vzorků na Analytic Server a případnému sdílení reportů na hostitelský OS. Stroj běží jen krátce a po přesunutí souboru je ihned vypnut. Pouze tento stroj má povolený přístup do sdílené složky hostitelského OS. Uvedenému účelu stroje plně vyhovuje minimalistický systém Alpine Linux.
- **Analytic Victim VM** představuje „oběť“ vybavenou typickými analytickými nástroji pro malware analýzu na Windows. Stejně jako stroj „Clean Victim“ má také stroj „Analytic Victim“ síťové služby simulované od stroje „Analytic Server“. Oba stroje Victim jsou maskovány před detekcí evasivními škodlivými kódy (opatřeními z §5.4).

5.1 Kali Linux jako podstavba pod virtualizovaným prostředím

Hostitelským OS fyzického stroje této práce, na kterém je virtualizované prostředí umístěno, je Kali Linux[97]. Použití Linuxu jako základu vychází z doporučované zásady („best practice“) mít na fyzickém stroji jiný druh operačního systému než ve virtuálním stroji, na kterém jsou spouštěny malware vzorky.[98] Jde o opatření, které má omezit riziko kontaminace fyzického stroje ze strany malware, který se zaměřuje převážně na Windows a také v této práci

[96] Git repozitář této práce: https://gitlab.com/jarkovar/javeman
[97] http://www.kali.org
[98] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online].

je spouštěn na virtualizovaných Windows (stroje Victim). Autorovi se osvědčilo zprovoznit plnohodnotný Kali Linux na USB flashdisku (nikoli pouze Live). To usnadnilo vytváření a používání variant analytických prostředí z několika různých USB na stejném počítači.

5.1.1 Plná instalace Kali Linuxu na USB a zprovoznění VirtualBoxu

Úspěšný postup plné instalace na USB zahrnoval využití dvou flashdisků.[99] První pomocné USB je zde značeno „A“ a jeho velikost stačí 4GB. Druhé hlavní USB je značené „B“ a jeho doporučená velikost je aspoň 64GB. Cílem je plná instalace Kali na hlavní USB „B“.

Nejprve bylo nutné zajistit, aby se Kali Linux z pomocného USB „A“ nainstaloval jako plný systém na druhé hlavní USB „B“ do správné sekce /dev/sdX, konkrétně /dev/sda. Tedy, aby byl výsledný systém Kali Linux nainstalovaný na druhém cílovém USB „B“ pokládán za primární systém a po instalaci se bez potíží spustil. Zápis obrazů operačních systémů z .iso souborů lze nahrát z Windows programem Rufus (autor použil verzi 2.18) a metodou zápisu dd.

- Nejprve byl na USB „A“ nahrán obraz kali-linux-light-2019.3-amd64.iso.
- Na USB „B“ byl nahrán pomocný obraz jiné live distribuce linuxového systému, konkrétně Fedora-Security-Live-x86_64-27-1.6.iso. Účelem tohoto pomocného live systému bylo, že jeho přítomnost na USB „B“ oproti prázdnému flashdisku způsobila, že se během instalace Kali Linuxu z USB „A“ na „B“ v instalačním průvodci cílové USB „B“ rozpoznalo jako /dev/sda. Bez triku s pomocným systémem by USB „B“ bylo rozpoznáno jako /dev/sdb a následné bootování do takto umístěného systému by bylo obtížné. Shrnutí nahraných obrazů:

Cílové USB „B“	Fedora-Security-Live-x86_64-27-1.6.iso[100]
Zdrojové USB „A“	kali-linux-light-2019.3-amd64.iso[101]

- Následně bylo po vložení obou USB do PC výhodné před instalací upravit v BIOSu pořadí boot options a paradoxně nastavit jako první volbu USB „B“ s Fedorou a až poté USB „A“ s Kali. Boot menu lze typicky vyvolat při startu počítače klávesami <F8> až <F12> nebo <Esc>:

```
BIOS boot options:
   1. USB „B“ - Live Fedora
   2. USB „A“ - Live Kali Linux
```

[99] Postup je aplikovatelný rovněž na externí disk, stačí jej v textu dosadit namísto USB „B“. Uvedený postup je nadbytečný, pokud se uživatel rozhodne pro dual-boot nebo instalaci pouze Kali na interní disk.
[100] https://download.fedoraproject.org/pub/alt/releases/27/Labs/x86_64/iso/Fedora-Security-Live-x86_64-27-1.6.iso
[101] http://archive.kali.org/kali-images/kali-2019.3/kali-linux-light-2019.3-amd64.iso

Nyní je po restartu počítače doporučeno znovu vyvolat Boot menu a zvolit volbu spuštění z pomocného USB „A“ s Live Kali Linux. Tím se dosáhne vhodného mapování /dev/sdX:

Cílové USB „B“ (dosud s Live Fedora)	/dev/sda
Zdrojové USB „A“ (Live Kali Linux)	/dev/sdb

- Nyní byla provedena instalace Kali Linuxu standardním postupem na cílové USB „B“ (mapované na /dev/sda) volbou „Install“ v menu Live Kali Linuxu. Systému bylo nastaveno několik přizpůsobení – hostname: javeman, domain: kali, vlastní root heslo: Javeman2018. Přihlašovací údaje do administrátorského účtu analytického prostředí jsou tudíž:

  Login: root | Heslo: Javeman2018

- Po dokončení instalace bylo zdrojové USB „A“ vyjmuto a v PC ponecháno jen USB „B“, nyní již s plnohodnotnou instalací Kali Linuxu.
- Systém bylo nakonec ještě třeba přidat a povolit v BIOSu. Do BIOS nastavení SecureBoot byla proto přidána položka „kali“ povolující zavedení systému Kali ze souboru /EFI/boot/bootx64.efi umístěného na USB „B“.

Po úspěšném vykonání uvedených kroků bylo spuštěnému systému Kali nakonfigurováno síťové připojení v /etc/network/interfaces a systém byl aktualizován příkazem apt-get upgrade.

Jak bylo zmíněno dříve, VirtualBox vyžaduje update kernelu systému a to bylo díky plnohodnotné instalaci nyní již proveditelné posloupností následujících příkazů:

```
sudo apt-cache search linux-headers
## (byla vybrána nejvyšší verze linux-headers, obdobně pro linux-image)
sudo apt-get -y install linux-headers-5.2.0-kali2-all
sudo apt-get -y install linux-image-5.2.0-kali2-all
sudo apt-get -y install build-essential dkms
sudo apt-get -y install virtualbox
```

Dále byla provedena multi-boot úprava umožňující spuštění plnohodnotných Kali z několika různých USB na stejném počítači. V linuxovém boot manageru GRUB byly každému systému přidány do boot menu položky ostatních systémů na USB rozlišených root_uuid identifikátory. Tím autor vyřešil omezení zaplněného pevného disku na laptopu (kapacita SSD disků často jen 256 GB) a mohl vyvíjet několik variant analytického prostředí bez vzájemného ovlivňování. Testy spuštění plnohodnotné instalace Kali z USB i následně VirtualBoxu (po

zapnutí Intel podpory virtualizace VTx, VTd v BIOSu) proběhly úspěšně na několika počítačích (desktopové Dell, HP a notebooky Acer, HP) s Intel CPU i3 a i5 od 3. až do 7. generace těchto procesorů a blíží se Live systému. Podmínkou spuštění Kali z USB byl UEFI mód bootování BIOSu a vyřešení akceptace EFI grubx64.efi bootloaderu v SecureBoot[102] nebo jeho vypnutí (Kali je založeno na distribuci Debian, kde je plná podpora UEFI SecureBoot ve fázi vývoje[103]):

```
grub-install --target=x86_64-efi --efi-directory=/boot/efi --bootloader-
id="$device_name" --recheck --debug /dev/sda
sed -i '$ s/.$//' /boot/efi/EFI/"$device_name"/grubx64.efi
update-grub && efibootmgr # Setup grub boot-menu and show current boot order
```

Nastavení bootování a vytvoření menu pro více USB s analytickým prostředím Javeman na Kali pomocí práce s příkazy `grub` a úpravami souvisejících souborů (grubx64.efi a šablon pro grub.cfg) je zahrnuto v instalačním multiboot skriptu prostředí Javeman `setup-multiboot.sh`.[104]

5.1.2 Doplnění Linuxu softwarovými balíčky pro analytické prostředí

Z návrhu prostředí vyplynulo přidání mnohých softwarových balíčků (packages) umožňujících zejména virtualizaci prostředí, jeho maskování, řízení analýzy a úpravy. Následuje přehled několika hlavních doplněných softwarových balíčků, přičemž úplný seznam obsahuje instalační skript prostředí Javeman `setup-javeman-on-kali.sh`[105]:

- Oracle VM VirtualBox 6.x: `virtualbox` (viz §5.1.1.)
- `git, xdotool, p7zip, leafpad, vim-gnome, pymisp, clamav, ufw ...`
- Python 2.x & Python 3.x (na Kali jsou Python 2 a 3 defaultně) byly doplněny moduly:`python-pip, python3-pip, pyvenv, python-colorama, python-requests`
- Knihovny, Python moduly a jiné požadavky polymorfních/metamorfních enginů (pro maskování analytických nástrojů): `metame, cmake, pefile, pydasm, capstone, keystone, radare2 (r2), simplejson` (instalováno Python venv).

Pozn. Mnoho nástrojů pro malware analýzu je naprogramováno v Pythonu, případně má pro Python interface, například nástroje VIPER[106], PyMISP, radare2 a poly-/metamorfní enginy.

[102] SecureBoot – Debian Wiki. Dostupné z: https://wiki.debian.org/SecureBoot.
[103] LARABEL, Michael. Debian Making Progress On UEFI SecureBoot Support In 2018.
[104] Git repozitář této práce. https://gitlab.com/jarkovar/javeman/blob/master/control/setup-multiboot.sh
[105] Git repozitář této práce. https://gitlab.com/jarkovar/javeman/blob/master/control/setup-javeman-on-kali.sh
[106] VIPER Binary Analysis and Management Framework. Dostupné z: https://viper.li/en/latest/

5.1.3 Persistentní Live USB s Kali Linux neposkytovalo virtualizaci

Pro úplnost uvádí autor záznam o použití distribuce Kali Linux Light 2018.1 64-bit vydané 26. ledna 2018[107] z počátku vývoje této práce. Použití Live Linuxu se však neosvědčilo.

Kali Linux byl nejprve spouštěn z persistentní Live distribuce na USB flashdisku. Kali Linux nabízí v live boot menu dvě volby umožňující tzv. persistenci – trvalé uchovování dat a změn provedených v systému Kali Linux i po opětovném restartu Live USB. Dokumentace Kali pro Live USB s persistencí doporučuje paměťové médium s kapacitou aspoň 8 GB, autor použil USB 3.0 Kingston 32GB. Z oficiálních stránek byl stažen soubor[108] s ISO obrazem systému Kali Linux Light 2018.1 a na USB byl zapsán programem Win32DiskImager[109] z Windows. V práci bylo tímto způsobem vytvořeno několik Live USB. Postup přípravy příkazem `dd` z Linuxu, resp. OS X je uveden v online dokumentaci Kali.[110] Alternativou je například příprava Live USB programem Rufus (oficiálně spustitelný pouze na Windows). Program Rufus umí live boot oddíl Linuxu vytvořit na FAT32 filesystému, který je čitelný i pro Windows a uživatel tím má možnost používat Kali Live USB rovněž jako běžný flashdisk. Dále bylo potřeba na počítači v BIOSu povolit boot option z USB a po vložení Live USB do počítače po restartování stisknout klávesu F12 pro boot menu (případně F8 aj. dle modelu PC). Defaultní přihlašovací údaje Live Kali Linuxu byly ponechány beze změn (Login: root | Heslo: toor). Příprava persistentního oddílu byla již prováděna z Linuxu. Live systémem z jednoho flashdisku byl konfigurován souborový systém jiného flashdisku. Zájemce o reprodukci postupu již nějaký Linux přímo na pevném disku (např. jako dual-boot) vystačí si s jedním flashdiskem, kde založí oddíl příkazem:

```
start=2g; end=32g; parted /dev/sdb mkpart primary $start $end
```

Zápis ext3 filesystému na oddíl disku "persistence" a vytvoření souboru *persistence.conf*:

```
mkfs.ext3 -L persistence /dev/sdb3
e2label /dev/sdb3 persistence && mkdir -p /mnt/liveusb
mount /dev/sdb3 /mnt/liveusb
echo "/ union" > /mnt/liveusb/persistence.conf && umount /dev/sdb3
```

Více podrobností persistence oddílu a šifrováném disku je uvedeno v dokumentaci Kali.[111]

[107] Společnost Offensive Security vydávala také v roce 2018 aktualizované verze Kali Linuxu každé čtvrtletí (2018.1 – 2018.4). V době odevzdání této diplomové práce byl proto používán již Kali Linux Light 2019.3.
[108] http://cdimage.kali.org/kali-2018.1/kali-linux-light-2018.1-amd64.iso
[109] https://launchpad.net/win32-image-writer
[110] https://docs.kali.org/downloading/kali-linux-live-usb-install
[111] https://docs.kali.org/downloading/kali-linux-live-usb-persistence

Použitá Live Kali Linux verze 2018.x obsahovala 4.x kernel[112]. Nové jádro způsobilo, že mnohé dříve vydané programy včetně virtualizačního nástroje VirtualBox na něj nebyly odladěny. Nový kernel přináší několik vlastností pro virtualizaci výhledově zajímavých. Jmenovitě například podporu šifrování DRAM paměti nebo podporu 5-úrovňového stránkování umožňující adresovat až 4 PB fyzické operační paměti a 128 PB virtuální paměti. S ohledem na bezpečnost prostředí určeného k manipulaci se škodlivým kódem bylo žádoucí použít nejaktuálnější verzi VirtualBoxu a tou byl pro platformu Linux v době psaní této části práce VirtualBox 5.2.8 – r121009 [113]. Standardní instalace softwarového balíčku VirtualBox

```
apt-get update & apt-get install virtualbox
```

však vedla k chybové hlášce (*LSB kernel modules failure*), která byla podle celé řady příspěvků v komunitních fórech Kali, Virtualbox[114] a dalších, způsobená zmíněnou nekompatibilitou kernelu systému s balíčkem virtualbox. Opakované pokusy odladit instalaci VirtualBoxu na Live systému selhaly. Ukázalo se, že podmínkou VirtualBoxu je update kernelu systému, který ovšem Live systém reálně neposkytuje.

Z těchto důvodů bylo rozhodnuto použít ke zprovoznění virtualizace plnou instalaci Kali Linuxu, která byla rovněž umístěna na USB (viz §5.1.1.). Systém s virtualizačním nástrojem byl dále vybaven vhodnými virtuálními stroji,

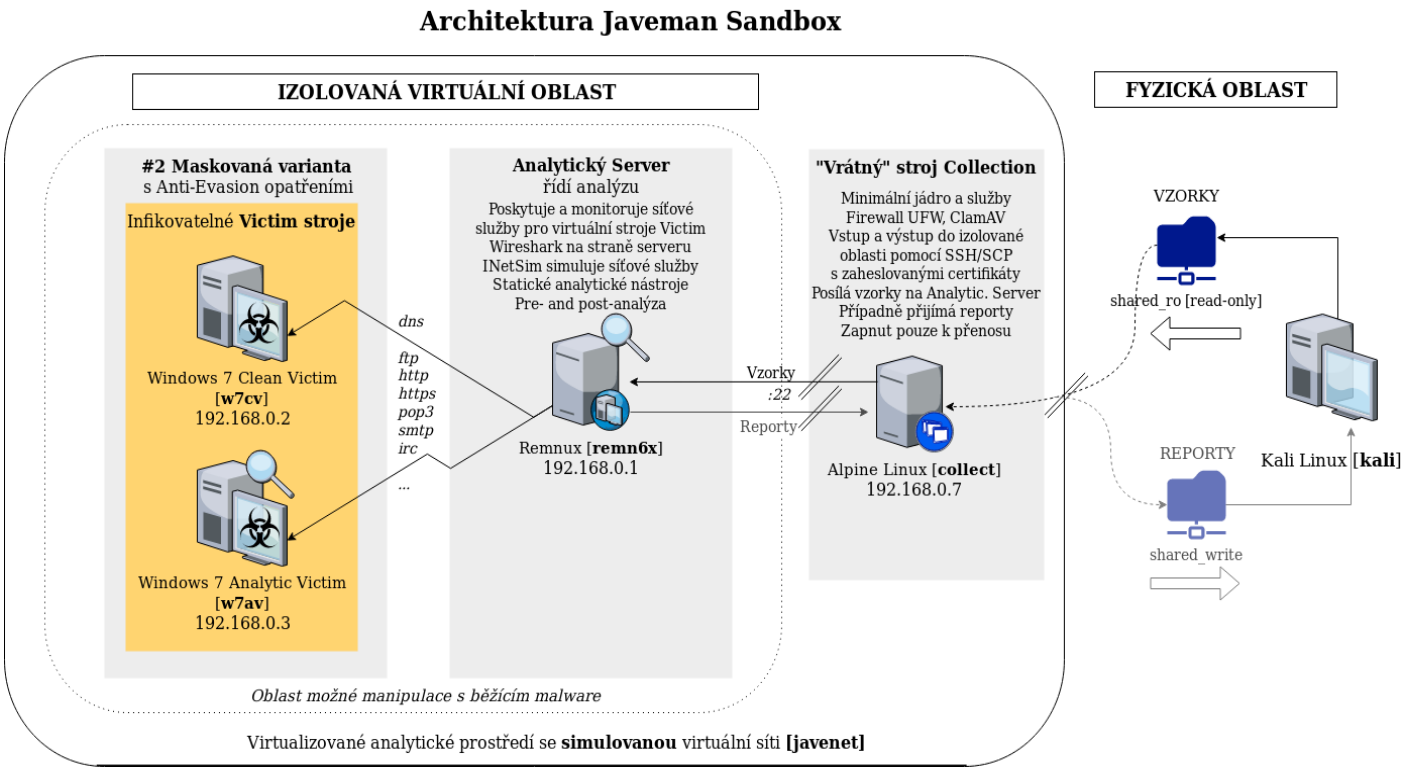
5.2 Virtualizované stroje

Na analýzu škodlivého kódu v izolovaném prostředí je využíváno několik typů virtuálních strojů propojených virtuální síti. Platformou virtualizace je Oracle VM VirtualBox[115].

Sestava nejdůležitějších strojů (Analytic Server, Victim a Collection) je umístěna v šifrovaném cloudovém úložišti ve formátu .OVA (Open Virtual Appliance)[116]. Virtuální stroje z úložiště automaticky stáhne instalační skript prostředí Javeman, naimportuje je do VirtualBoxu a nakonfiguruje k dalšímu použití (včetně některých maskovacích opatření).

[112] https://www.kali.org/news/kali-linux-2018-1-release/
[113] https://download.virtualbox.org/virtualbox/5.2.8/virtualbox-5.2_5.2.8-121009~Debian~stretch_amd64.deb
[114] Diskuze ohledně Virtualbox vboxdrv.Dostupné z: https://forums.virtualbox.org/viewtopic.php?f=7&t=83943
[115] Oracle VM VirtualBox Datasheet. 2019. Dostupné z:
http://www.oracle.com/us/technologies/virtualization/oraclevm/oracle-vm-virtualbox-ds-1655169.pdf
[116] Overview of the Open Virtualization Format. https://support.citrix.com/article/CTX121652

Virtuální stroje analytického prostředí Javeman jsou umístěny ve virtuální síti javenet, přičemž infikovatelným strojům Victim jsou běžné síťové služby simulované strojem Analytic Server:



Obrázek 7: Schéma architektury analytického prostředí se simulovanou virtuální sítí

V této kapitole následují dále typy virtuálních strojů a za jejich popisem je uveden přehled analytických a souvisejících nástrojů, kterými byl stroj dovybaven a jaký byl jejich význam pro malware analýzu. Kromě samotných nástrojů malware analýzy, vyžadovaly některé nástroje speciální runtime prostředí. Nativní nástroje Windows zase posloužily k realizaci anti-evasion opatření uvnitř strojů Victim či usnadnění analýzy. Z důvodu úspory místa na disku bylo u strojů Victim využíváno sdílení jednoho základu více virtuálními stroji („shared virtual disk VDI“)[117].

5.2.1 VM „Clean Victim“ [w7cv]

Stroje typu Victim jsou v této práci nazývány virtuální stroje s Windows určené pro infikaci škodlivým kódem. Victim strojům jsou poskytovány síťové služby z jiného stroje Analytic Server, který případnou komunikaci zároveň zachytává. Při analýze tímto způsobem nemá škodlivý kód příliš možností zaznamenat analytické nástroje. Operačním systémem strojů Victim jsou Windows 7, které jsou na desktopech podle několika dostupných statistik dosud

[117] Special Image Write Modes. Dostupné z: https://www.virtualbox.org/manual/ch05.html#hdimagewrites

značně celosvětové rozšířené.[118,119,120] Systém Windows 7 s časově omezenou platností lze legálně stáhnout na oficiálních stránkách Microsoftu přinejmenším dvěma způsoby: Buď jsou Windows ke stažení jako již nainstalované v 90denní zkušební VM s prohlížečem od Microsoftu[121], anebo jako instalační .ISO po zadání platného produktového klíče do příslušného online formuláře[122]. Navíc mají studenti technických oborů často přístup k projektu MSDNAA poskytujícího bezplatně vybraný software včetně operačních systémů pro akademické účely. Další konfigurace strojů Victim je již závislá na variantě analytického prostředí (#0, #1 nebo #2).

I na virtuálním stroji Clean Victim napodobujícím bezbrannou oběť mohou být nástroje využitelné pro malware analýzu, pokud jsou typické pro běžné uživatele a nikoli analytiky. K tomu jsou vhodné zejména nativní administrátorské nástroje, které jsou standardní součástí Windows.[123] Některé běžné programy poskytují užitečné funkce, jako např. 7-Zip, který umí rozbalit libovolný binární Windows PE (.EXE, .DLL, aj.) soubor a byly proto doinstalovány. Takové „obyčejné" nástroje uniknou detekci od malware i pokud by malware cíleně po analytických nástrojích pátral. Na velmi jednoduchou analýzu lze přímo na Clean Victim použít:

- PowerShell[124,125,126], Services.msc, Msconfig.exe, Eventvwr.msc, Taskschd.msc
- 7-Zip[127], CMD[128] – icacls, netstat, SC, TaskList, type, SigVerif[129]…
- WMI (Windows Management Instrumentation) – WMIC[130], wbemtest.exe…

[118] ADAMS, Matthew. Windows 10 hits 35% user base, Windows 7 takes the crown with 43%.
[119] Net Applications – podíl počítačů s Win7 ~42% (Q3 2018). https://www.netmarketshare.com/operating-system-market-share.aspx?id=platformsDesktopVersions
[120] StatCounter – více než třetina počítačů s Win7 (Q3 2018). http://gs.statcounter.com/os-version-market-share/windows/desktop/.V Číně dokonce podíl Windows 7 přesahuje 55%. http://gs.statcounter.com/windows-version-market-share/desktop/china.
[121] https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/
[122] https://www.microsoft.com/en-us/software-download/windows7
[123] MARAK, Victor. Leveraging the command line for windows: malware analysis and forensics.
[124] SAJEEV, Nair. Live Response Using PowerShell. Dostupné z: https://www.sans.org/reading-room/whitepapers/forensics/live-response-powershell-34302.
[125] LOMBARDI, Mike. Rolling Your Own Incident Response with PowerShell - SANS Blue Team Summit 2018. Dostupné z: https://www.sans.org/cyber-security-summit/archives/file/summit-archive-1524594407.pdf.
[126] https://github.com/mattifestation/PowerShellArsenal
[127] Use 7-Zip to explore & extract EXE file contents. Dostupné z: https://qtechbabble.wordpress.com/2016/11/07/use-7-zip-to-explore-exe-file-contents/.
[128]Například pro podrobné výpisy: C:\>Tasklist /M – všechny DLL moduly využívané běžícími procesy; C:\>icals – změny ACL oprávění systémových souborů. C:\>sc queryex – služby/ovladače běžící na pozadí včetně flags…
[129] MUELLER, John Paul. *Příkazový řádek Windows pro Windows Vista, 2003, XP a 2000*. SigVerif vypíše podpisy (systémových) ovladačů do textového logu a umožňuje tak základní kontrolu bezpečnosti těchto ovladačů.
[130] MARAK, Victor. *Command Line for Windows Malware and Forensics* [online]. Popis využití konzole WMIC v příkazovém řádku pro základní statickou analýzu, debugování spustitelného souboru a dílčí forenzní analýzu.

5.2.2 VM „Analytic Server“ (Remnux) [remn6x]

Stroj typu Analytic Server simuluje oběti Victim síťové služby, umožňuje analyzovat malware linuxovými nástroji statické analýzy (např. framework VIPER) a zachytávat síťovou komunikaci v rámci dynamické analýzy. Virtuální stroj je založen na linuxuovém systému Remnux, který vytvořil Lenny Zeltser, původně pro výukové účely a kurzy malware v rámci organizace SANS Institute.[131] Mnohé nástroje Analytického Serveru [remn6x] bylo třeba vhodně nakonfigurovat a přizpůsobit automatizované analýze, jak pro simulaci síťových služeb a jejich sledování, tak pro samotný průběh analýzy řízené z vně prostředí:

- Síťová aktivita: Wireshark, INetSim
- Statická a dynamická analýza: VIPER, ProcDOT[132],Volatility (analýza RAM)

Stroj byl vybaven zaheslovanými SSH certifikáty pro zabezpečený přenos vzorků. Již existující analytické nástroje byly v této práci aktualizovány a rozšířeny, například o univerzální disassembler/debugger IDA v7.0 Free for Linux[133] včetně pluginů stealth a findcrypt[134].

5.2.3 VM „Collection“ (Alpine Linux) [collect]

Stroj typu Collection zajišťuje distribuci vzorků na Analytic Server po simulované virtuální síti. Požadavkům na účel, bezpečnost a ideálně minimalismus vhodně vyhovuje Alpine Linux.[135] Instalační ISO Alpine Linux 64-bit v úpravě pro virtuální prostředí má minimální velikost pouhých 32 MB.[136] První přihlášení na uživatele `root` nevyžaduje heslo. Samotná instalace proběhla spuštěním skriptu ‘setup-alpine’ a byla zvolena možnost `sys mode`, aby se data systému ukládala na virtuální disk. Z nabízených balíčků byl vybrán pouze `openssh`, jenž poskytne příkazy `scp` a `ssh` pro přenos vzorků a reportů mezi stroji Collection a Analytic Server. Síťová konfigurace rozhraní eth0 byla nejprve ponechána na DHCP, aby měl Alpine Linux během instalace přístup k repozitářům umístěným na internetu. Systému bylo nastaveno:

```
### hostname: collect
### root heslo: Javeman2018
```

[131] Version 6 Release of REMnux Linux. Dostupné z: https://zeltser.com/remnux-v6-release-for-malware-analysis.
[132] WOJNER, Christian. ProcDOT. Dostupné z: http://procdot.com/index.htm.
[133] IDA Support: Freeware (IDA 7 for Linux). Dostupné z: https://out7.hex-rays.com/files/idafree70_linux.run
[134] IDA Support: Download Center. Dostupné z: https://www.hex-rays.com/products/ida/support/download.shtml
[135] Alpine Linux. Dostupné z: https://alpinelinux.org/about/.
Pozn. Zpočátku byl na Collection VM používán TinyCore Linux. Ovšem po komplikacích s vlastností frugal mode (obdoba live systému) a kompilací balíčků byl včas nahrazen mnohem lépe vyhovující distribucí Alpine Linux.
[136] Alpine Linux.Dostupné z: http://dl-cdn.alpinelinux.org/alpine/v3.8/releases/x86_64/alpine-virt-3.8.1-x86_64.iso.

Pro účely doplnění sdílené složky a antivirového nástroje ClamAV byly odkomentovány znaky # v řádcích obsahujících odkazy na repozitáře pomocí `vi /etc/apk/repositories`.

```
apk update && apk upgrade
```

Příkaz `uname –a` vypsal o stroji v době vzniku, že používal verzi Alpine Linux 3.8.1 s jádrem Kernel 4.14.69-0-virt on x86_6o.[137] Systém byl mezitím aktualizován na Kernel 5.4.1-0-virt.

```
addgroup sudo
adduser term
adduser term sudo
apk add sudo
```

Nově založenému uživateli bylo nastaveno implicitní heslo `Javeman2018`. Po spuštění `visudo` byl v editovaném souboru odstraněn znak komentáře "#"z řádku pod skupinou ALL.

```
## Uncomment to allow members of group sudo to execute any command
%sudo ALL=(ALL) ALL
apk add clamav && freshclam
apk add virtualbox-guest-additions virtualbox-guest-modules-virt
reboot
modprobe –a vboxsf
mkdir /mnt/shared_ro && mkdir /mnt/shared_write
mount -t vboxsf sf_shared_ro /mnt/shared_ro
mount -t vboxsf sf_shared_ro /mnt/shared_write
```

Po instalaci bylo Collection VM v nastavení VirtualBoxu nastaveno síťové připojení na izolovanou virtuální síť. Zároveň byla uvnitř VM v Alpine Linux upravena konfigurace síťového rozhraní eth0 v souboru: `/etc/network/interfaces`

```
auto eth0
iface eth0 inet static
address 192.168.0.7/24
gateway 192.168.0.1
```

a též DNS bylo nasměrováno na Analytic Server doplněním položek do `/etc/resolv.conf`

```
nameserver 192.168.0.1
domain service.org  ## Simulated domain name
```

[137] Alpine Linux 3.8.1 Released. Dostupné z: https://alpinelinux.org/posts/Alpine-3.8.1-released.html.

Manuální přesun vzorků na Analytický Server po simulované virtuální síti lze provést příkazem:

```
scp sample.zip remnux@192.168.0.1:~/Samples
```

Vysoká bezpečnost vyžaduje minimum služeb uvádí kniha „Bezpečnost v Linuxu“.[138] Ve smyslu tohoto doporučení byly firewallem iptables Alpine Linuxu povoleny pouze SSH (TCP 22) a DNS (53) směrem do virtuální sítě a zpět. Linux kernel byl aktualizován v listopadu 2019.

5.2.4 VM „Analytic Victim“ [w7av]

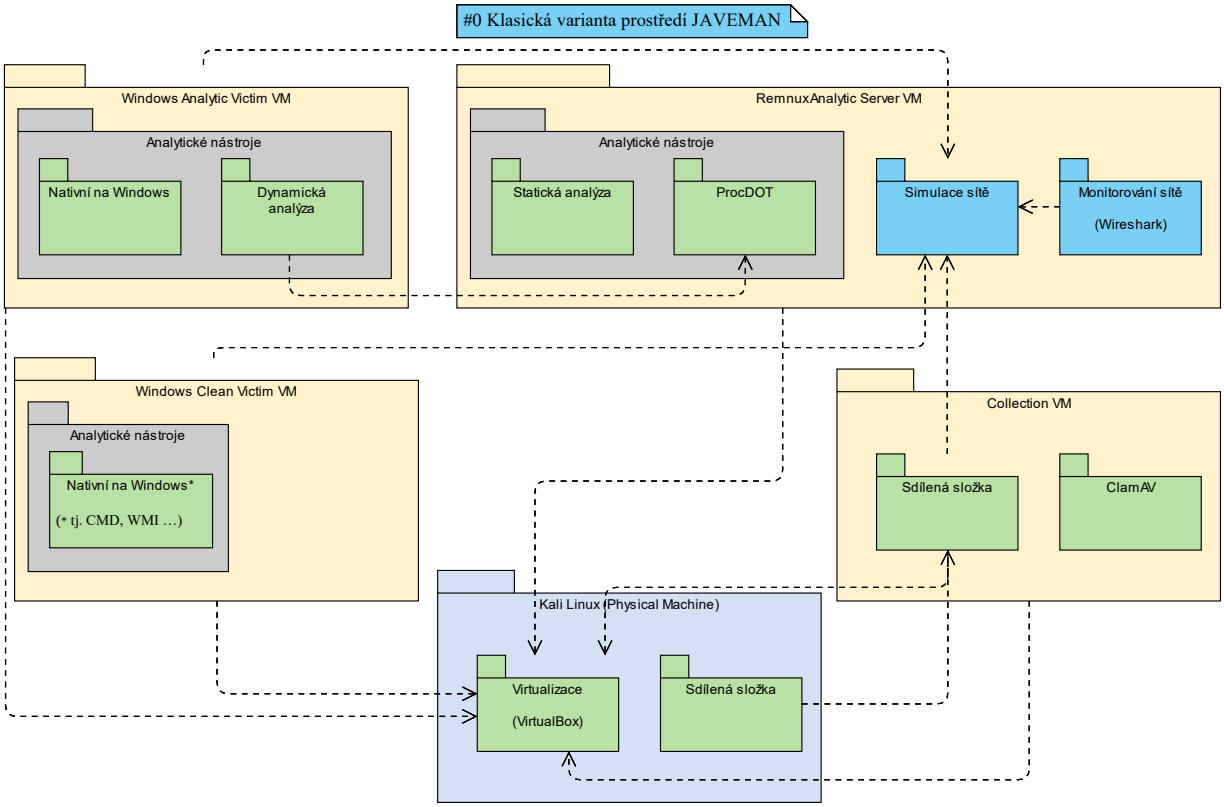
Stroj Analytic Victim vychází z Clean Victim. Odlišuje se výbavou nástroji pro analýzu. Autorovi se osvědčily zejména zamaskované analytické nástroje pro dynamickou analýzu, jmenovitě Procmon pro zachycení lokální aktivity a WinDump logující síťovou aktivitu škodlivého kódu. Jejich spuštění je při automatizované analýze prováděno z vně virtuálního prostředí originálním postupem simulace stisku kláves, čímž zároveň dochází ke generování simulované uživatelské aktivity (popis postupu řízení analýzy obsahuje kap. §6.2.2). Přehled vhodného vybavení stroje Analytic Victim:

- Python 3.x
- Univerzální nástroje
  - Eset SysInspector 1.3.5[139] (ideálně česká verze, aby zmátla malware)
- Statická analýza
  - IDA v7.0 Free for Windows[140], Immunity Debugger (Python)
  - SSMA[141] – (detekuje anti-virtualizační techniky vzorku; vyžaduje Python 3)
- Dynamická analýza
  - SysInternals[142] – Procmon 3.5 (na logy do ProcDOT), Autoruns, ProcExp.
  - WinDump[143] (na logy do ProcDOT)
- Anti-evasion (v případě maskovaných variant analytického prostředí)
  - Makin (odhalí některé evasivní techniky malware vzorku)[144]
  - Hidden (na skrývání programů a procesů analytických nástrojů „uvnitř“ W7)

[138] TOXEN, Bob. *Bezpečnost v Linuxu*. Více o vypnatí nepotřebných služeb zejména na s. 95 – 110.
[139] ESET. Free Diagnostic Tool SysInspector. Dostupné z: https://www.eset.com/int/support/sysinspector/
[140] IDA Support: Freeware (IDA 7 Windows). Dostupné z: https://out7.hex-rays.com/files/idafree70_windows.exe.
[141] Simple Static Malware Analyzer. Dostupné z: https://github.com/secrary/SSMA.
[142] Windows Sysinternals. Dostupné z: https://docs.microsoft.com/en-us/sysinternals/.
[143] WinDump: tcpdump for Windows using WinPcap. https://www.winpcap.org/windump/
[144] https://github.com/secrary/makin

5.3 Varianty prostředí pro malware analýzu

5.3.1 Varianta 0: „Klasické“ prostředí pro malware analýzu (bez opatření)



Obrázek 8: Schéma se závislostmi mezi prvky analytického prostředí klasické varianty #0

5.3.2 Nasazení statické a dynamické malware analýzy

Na počátku analýzy se jeví jako vhodné vzorek škodlivého kódu nejprve podrobit zběžné statické analýze, která často odhalí různé evasivní techniky a umožňuje je zneškodnit ještě před dynamickou analýzou ve virtuálním prostředí.[145] Nejprve je proto vzorek automatizovaně předložen na Analytic Serveru statickému analyzátoru VIPER. V případě manuální analýzy může uživatel vzorek importovat do disassembleru jako například IDA, přepsat vzorku několik bytů v úsecích, kde detekuje prostředí nebo uplatňuje techniky skrývání, a předpřipravit si tak vzorek pro účinnější analýzu ve virtuálním prostředí. To je velmi přínosné zejména v případech, kdy by nutná úprava virtuálního prostředí (s cílem oklamat evasivní techniku zkoumaného vzorku) byla z principu neproveditelná, a naopak úprava vzorku jde provést velmi dobře. Jedná se o reverzní inženýrství bez újmy na obecnosti chování vzorku.

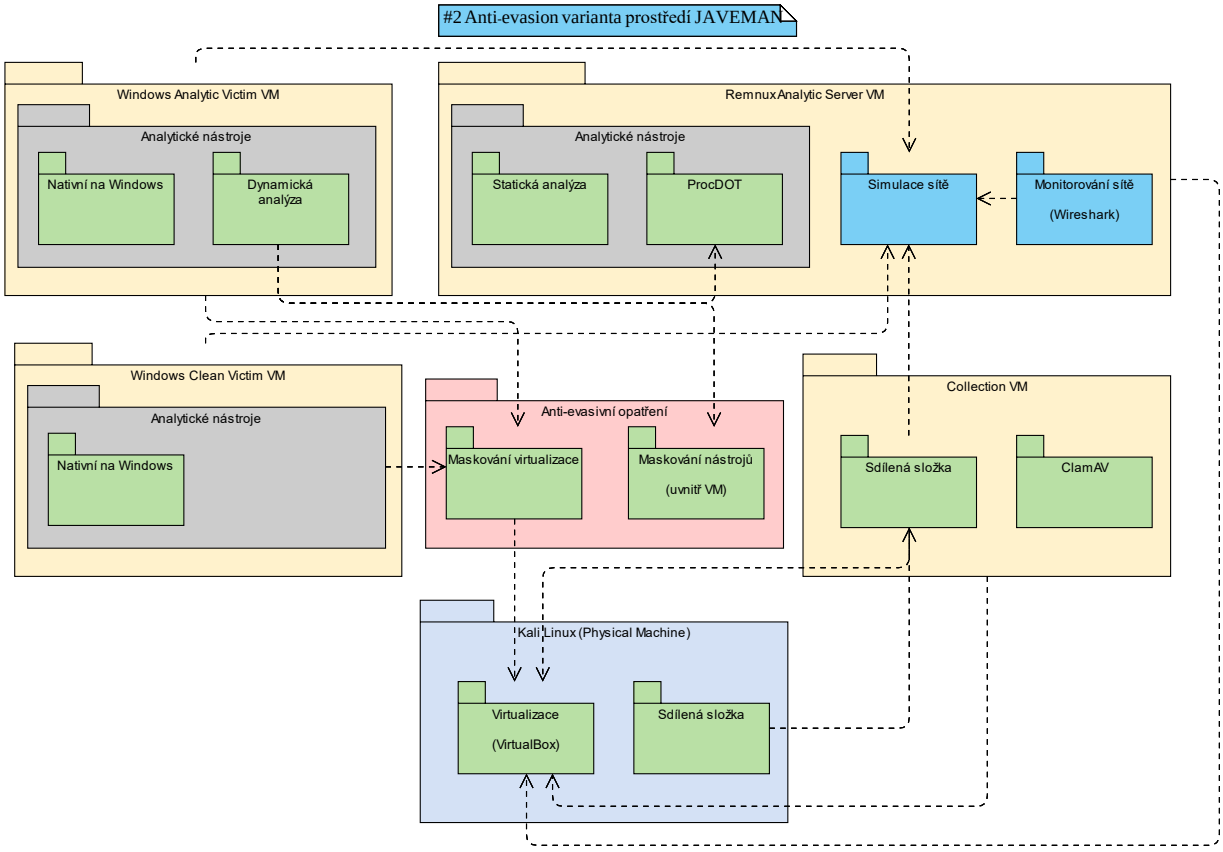
[145] SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis* [online].

Pokud by se například vzorek dotazoval před zahájením škodlivých aktivit na počet CPU a obsahoval by podmínku, že mají být přítomna aspoň 4 fyzická jádra CPU a bylo by obtížné tuto vlastnost nastavit anebo simulovat ve virtuálním prostředí, můžeme vzorku podmínku přepsat a snížit mu v ní počet fyzických CPU jader na lépe simulovatelnou hodnotu. Následně je vzorek přesunut na stroj Victim k dynamické analýze s případným spuštěním nástrojů Procmon a WinDump. K analýze i vizualizaci jejich logů poslouží ProcDOT (viz kap. §5.5.3).

5.3.3 Generování reportů o analýze škodlivého kódu

Díky tomu, že existují standardizované formáty pro reporty a mnohé jsou vzájemně převoditelné, stačí připravit nálezy z analýzy v jakékoliv podobě přijatelné jednomu z těchto formátu. Například MISP poskytuje nástroj pro převod mezi MISP a STIX formáty.[146] MISP dokonce umožňuje nahrávat zjištění z analýzy malware v libovolné formě včetně prostého textu.

5.3.4 Varianty 1-2: „Anti-evasion“ prostředí pro malware analýzu



Obrázek 9: Schéma se zobecněnými závislostmi mezi prvky anti-evasion variant #1 a #2

[146] https://github.com/MISP/MISP-STIX-Converter

5.4 Rozšíření analytického prostředí o anti-evasion opatření

Oblast VIRTUALIZACE	
Detekční technika evasivního malware	**Maskovací (anti-evasion) opatření**
Instrukce `IN (VMware magic number)` na port `0x5658 (VMware hypervisor port).`	Týká se zejména nástroje VMWare, použití jiného virtualizačního nástroje (VirtualBox).

Detekce založené na instrukci CPUID:[147]
`Malware předpokládá CPUID fyz. <> virt. Zavolání instrukce CPUID se vstupem EAX=1 vrátí jako return hodnotu popis vlastností procesoru. Na fyzickém stroji bude 31. bit registru EAX = 0, zatímco na virtuálním je hodnota registru = 1. Zavolání CPUID se vstupem EAX=40000000 vrátí v EAX, ECX, EDX tzv. vendor string`
	Nastavením: `VirtualBox -> Settings(VM) -> System -> Acceleration -> Paravirtualization Interface = „None“`
	Nastavením CPUID na bity reálného procesoru, které jsou z fyz. stroje nahrány do virt. stroje:[148]
	`vboxmanage list hostcpuids > i5cpu`
	`(Leaf__EAX_____EBX_____ECX_____EDX)`
	`00000000 00000016 756e6547 6c65746e 49656e69`
	`vboxmanage modifyvm $vm --cpuidset $line`

`Detekce výrazně abnormální HW konfigurace:`
`HDD < 100 GB ?`
`CPU < 2? ?`
`Memory < 2048 MB ?`
	Nastavením HW konfigurace ve VirtualBoxu, která napodobuje běžný low-end notebook:[149]:
	`HDD (SSD) 128 GB`[150]`, CPU 2 jádra, Operační paměť 2048 GB, GPU Intel HD Graphics.`

Detekce charakteristických názvů souborů, procesů, oken, uživatele, položek Windows Registry, síťové konfigurace (MAC,IP), např.:
`User: cuckoo, cuckoosandbox, IEUser`
`"vboxservice.exe" //VboxGuestAddProcess "\system32\drivers\VBoxGuest.sys"`
	Vynechání nebo změna názvů typicky detekovaných položek:
	`- Absence VirtualBox Guest Additions (pouze odinstalace zanechává stopy, v Guest VM vůbec k instalaci nedojde)`
	`- Změna názvů a hodnot VM (antivmdetect)`

[147] https://www.cyberbit.com/blog/endpoint-security/anti-vm-and-anti-sandbox-explained/
[148] https://superuser.com/questions/625648/virtualbox-how-to-force-a-specific-cpu-to-the-guest
[149] Vyhledáváním s těmito parametry bylo na v USA populárních e-shopech (např. Newegg.com) s elektronikou ověřeno, že se jedná o v listopadu 2018 dosud prodávané konfigurace notebooků.
[150] Virtuální disk používá dynamickou alokaci, kdy skutečná velikost na fyzickém stroji expanduje expanduje a obvykle nedosahuje velikosti nastavené pevnému disku VM. Pohledem zevnitř VM však nastavenou velikost má. Více o virtuálním disku v dokumentaci VirtualBoxu: Dostupné z: https://www.virtualbox.org/manual/ch05.html.

"08:00:27" VBox MAC address vendor"[151] C:\WINDOWS\system32\drivers\vmmouse.sys Zdroj: www.malwarestats.org/tricks.html	- Uživatel „Jave Man“ (javeman) není dosud považován za známku sandboxu - IP z rozsahu 192.168.x.x je pro běžné uživatele nacházející se za aktivním prvkem typická, proto lze ponechat. -Pojmenování českými názvy či pozpátku.
Detekce připojení na Internet: Pokus o připojení na známé IP, domény (např. google.com) anebo specifické pro malware (např. WannaCry ‚kill-switch‘ doména[152]).	Simulace sítě doplněná několika často testovanými adresami (např. Google DNS 8.8.8.8.). Ostatní budou přidány manuálně dle zachycené komunikace a analýza zopakována.
Detekování známek uživatelské aktivity: GetCursorPos, WaitForSingleObject user activity check, GetLocaleInfo, language check Zdroj: analýza bankovního malware Gozi[153]	Generování uživatelské aktivity[154]: - simulace stisků kláves z vně GuestVM: vboxmanage modifyvm keyboardputscancode[155] - simulace pohybů myši z vně GuestVM: VirtualBox API putMouseEventAbsolute[156], resp. pomocí xdotool nebo evemu[157] (případné pokročilé simulace programy uvnitř Guest VM: např. AutoIT, SikuliX)
Detekce známek používání a opotřebení: Application.RecentFiles.Count (Dridex malware) <Chrome Cache Folder><Mozilla History File> <Mozilla Cache Folder> <IE History Folder> <IE Cache Folder>	Používání systému s Windows ve virtuálním stroji po nějakou dobu jako typický uživatel: - Manipulace se soubory (tvorba, kopírování, mazání). - Instalace typických programů jako Chrome, Adobe Reader, Firefox, apod. - Používání a přizpůsobení prohlížeče

Tabulka 4: Některé techniky detekce evasivním škodlivým kódem a maskovací opatření v oblasti virtualizace

[151] IEEE Standards Association. MAC Address Block Large. Dostupné z: http://standards-oui.ieee.org/oui/oui.txt.
[152] https://umbrella.cisco.com/blog/2017/05/16/the-hours-of-wannacry/
[153] Analyzing Gozi's Anti-Analysis Tricks. Dostupné z: https://joesecurity.org/blog/5852460122427342172.
[154] V práci použitá generovaná aktivita simulace stisků kláves originálním způsobem zároveň řídí stroje a automatizuje proces analýzy (viz .
[155] Aplikace funkce keyboardputscancode z příkazové řádky je blíže popsána v §5.2. této práce.
[156] VirtualBox Main API IMouse Interface Reference. Dostupné z: https://www.virtualbox.org/sdkref/interface_i_mouse.html#a5a59889c3f92d2e4e797f15ce9385a9c.
[157] https://fedoramagazine.org/simulate-device-input-evemu/

Oblast ANALYTICKÉ NÁSTROJE	
Detekční technika evasivního malware	**Maskovací (anti-evasion) opatření**

Detekce názvů souborů, běžících procesů a jejich popisu, názvů oken, apod. populárních analytických nástrojů (zároveň jde obvykle o nástroje atypické pro obyčejné uživatele):
`"Procmon", procmon.exe, wireshark.exe...`
`"Agent.py", PEiD.exe, PEview.exe...`
`"ProcessHacker"`
	Změna názvů analytických nástrojů:
	`- Přejmenování souborů, např. pozpátku Procmon.exe -> nomcorP.exe`
	`- Změna textu popisu 'Description' zobrazujícího se u běžícího procesu a např. editací EXE reverzním inženýrstvím)`
	`- Minimum analytických nástrojů na VM k infikování malware, varianta Clean Victim VM jen s nativními Win nástroji (např.Wireshark stačí mít na Server VM)`

Detekce přítomnosti analytických nástrojů metodami statické analýzy (hash, signatury), obdobou postupů známých z antivirové detekce.
(Jedná se o autorem očekávaný směr vývoje inspirovaný z biologie, kde bývá ko-evoluce ve vztahu parazit-hostitel, predátor-kořist častá.):
`MD5 Procmon.exe: 42d273490f10fa08b60bf51bfd66ba3b`
`SHA1 Procmon.exe: 68bcdd6164f22b571149c3a95cda0a5729b3ba38`
	Maskováním analytických nástrojů před statickou detekcí (hash, signaturami) metodami známými od škodlivého kódu z uplynulé doby, konkrétně mutací nástrojů polymorfními a metamorfními enginy:
	`- Změna vnitřku nástroje (instrukcí spustitelného souboru) změní hash a unikne jednoduché detekční signatuře.`
	`peCloakCapstone.py ../Procmon.exe`
	`metame.py -d -i Procmon.exe -o nmP.exe`
	`(0x401008 (xor eax,eax) -> sub eax,eax)`

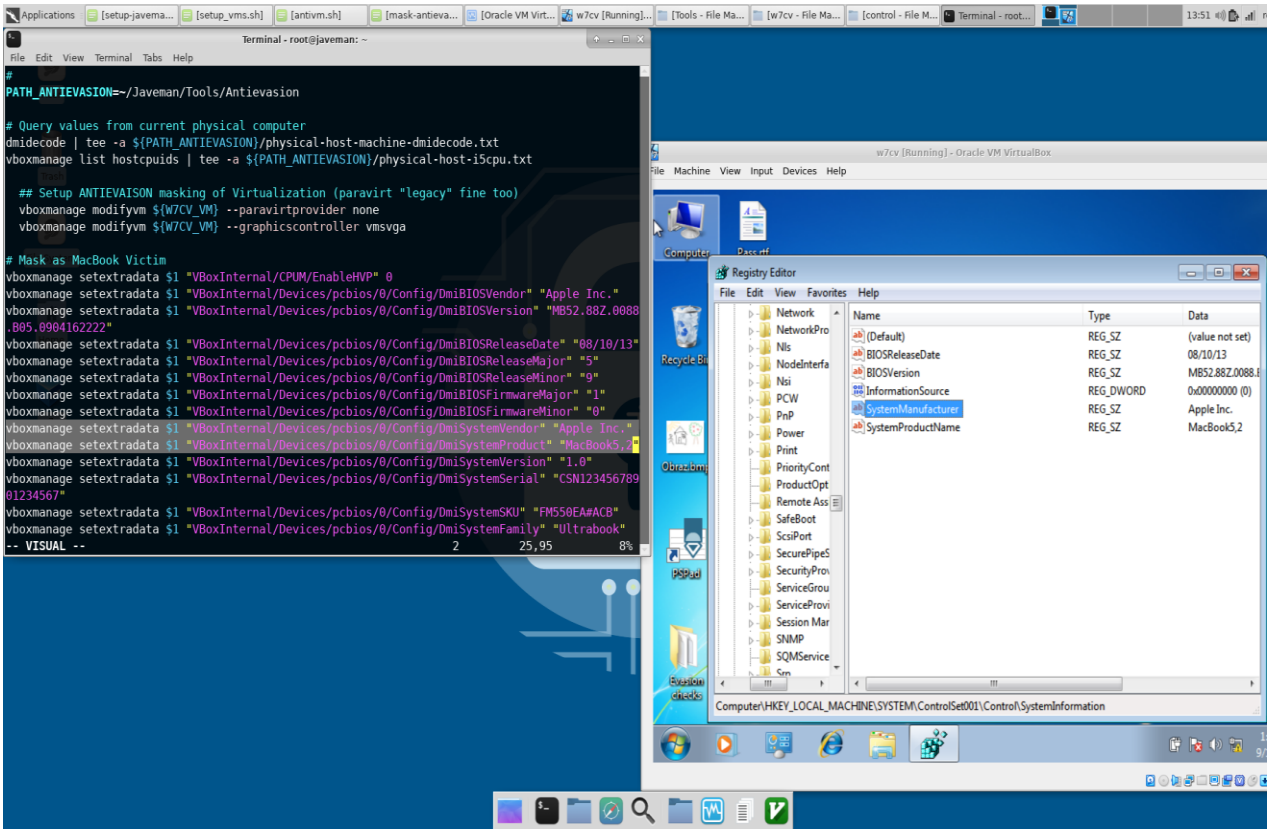
Tabulka 5: Některé techniky detekce evasivním škodlivým kódem a maskovací opatření analytických nástrojů

Maskování virtualizace obecně

Maskování virtualizace bylo provedeno na fyzickém Kali Host po vytvoření stroje Victim, avšak před jeho prvním spuštěním, zejména úpravami charakteristik ovladačů systému. Maskování virtuálních strojů Victim provádí autorem napsaný instalátor prostředí Javeman[158] a Bash skript

[158] Git repozitář této práce: https://gitlab.com/jarkovar/javeman/blob/master/control/setup-javeman-on-kali.sh

`mask-antievasion-vm.sh`[159]. Některými upravenými hodnotami je stroj Victim úspěšně maskován za hardware společnosti Apple (Obrázek 10).



Obrázek 10: Maskování stroje v registrech Windows za Apple (dle SystemManufacturer)

Maskování časování virtualizace pomocí úprav „RTSC“[160]

Jde o obranu proti detekci virtualizace pomocí časování RTSC (tzv. *Real Time Stamp Counters).*

```
sudo vboxmanage setextradata "W7CV" VBoxInternal/TM/TSCMode RealTSCOffset
sudo vboxmanage setextradata "W7CV" VBoxInternal/CPUM/SSE4.1 1
sudo vboxmanage setextradata "W7CV" VBoxInternal/CPUM/SSE4.2 1
```

Úpravy zvyšující odolnost virtuálního prostředí vůči detekcím rozdílů v časování Guest/Host jsou zahrnuty v maskovacím skriptu této práce. Vhodné další úpravy poskytne zkoumání zdrojového kódu VirtualBoxu, konkrétně prvku TSCMode třídy Time Manager.[161] Autor empiricky ověřil, že aktuální verze VirtualBoxu bohužel již nepodporuje nastavení parametru VBoxInternal/TM/UseRealTSC[162] za účelem vypnutí emulace instrukce RDTSC.

[159] Git repozitář této práce: https://gitlab.com/jarkovar/javeman/blob/master/control/mask-antievasion-vm.sh
[160] VirtualBox User Manual – Chapter 9. Advanced. https://www.virtualbox.org/manual/ch09.html#changetscmode.
[161] VirtualBox SVN. Dostupné z: https://www.virtualbox.org/svn/vbox/trunk/src/VBox/VMM/VMMR3/TM.cpp
[162] VirtualBox. Disable rdtsc emulation. Dostupné z: https://forums.virtualbox.org/viewtopic.php?f=1&t=60980

5.5 Rozšíření analytického prostředí o bio-inspirovaná opatření

5.5.1 Polymorfní a metamorfní zamaskování analytického nástrojů

Zamaskování programu polymorfními či metamorfními generátory (enginy) nemusí být nutně jen doménou malware autorů. Škodlivý kód, který usiluje o detekci obranných nástrojů, vstupuje do problematiky detekce, která byla donedávna vyhrazená analytikům. Takový škodlivý kód se dostává do role jakéhosi anti-antiviru. Lze tudíž proti malware autorům, kteří během uplynulých let aktivně vynalézali, jak úspěšně zmařit detekci vybraného kódu, použít jejich vlastní metody, včetně polymorfismu. Některé poly- a metamorfní enginy, využitelné k zamaskování analytických nástrojů, jsou založené na disassembly frameworku Capstone[163]:

peCloakCapstone[164] – nástroj v Pythonu zamaskuje spustitelné Windows soubory (PE), aby unikly detekci signatur, kterou typicky používají antiviry. V roce 2015 tím bezpečnostní výzkumník skryl několik testovacích škodlivých kódů před většinou tehdy rozšířených AV.[165]

pymetamorph[166] – metamorfní engine spustitelných souborů Windows (napsaný v Pythonu).

Resonance[167] – velmi jednoduchý polymorfní engine napsaný v jazyce C.

Evoasm[168] – engine generující kód metodou genetického programování, tzv. AIMGP (Automatic Induction of Machine Code by Genetic Programming).[169]

metame[170] – metamorfní engine napsaný v Pythonu a používající sadu pro reverzní inženýrství radare2[171] umí zamaskovat 32-bitové i 64-bitové spustitelné soubory (ověřeno na Windows PE).

Detekci signatur (charakteristického otisku) typicky používají antiviry. Škodlivé kódy se před detekcí skrývaly, například polymorfními technikami. Nyní ovšem základní detekci pomocí signatur (často hash) uplatňují některé škodlivé kódy k detekování analytických nástrojů.

[163] http://www.capstone-engine.org/
[164] https://www.securitysift.com/download/peCloak.py resp. https://github.com/v-p-b/peCloakCapstone
[165] CZUMAK, Mike. *PeCloak.py – An Experiment in AV Evasion* [online].
[166] https://github.com/JuanJMarques/pymetamorph/
[167] https://github.com/krystalgamer/Resonance
[168] https://github.com/evoasm/evoasm.rb
[169] Metoda AIMGP funguje tak, že se jí nejprve předloží skupina příkladů zadání, tedy několik párů vstup/výstup popisujících žádoucí chování programu (specifikaci), a metoda poté pomocí genetického programování navrhne několik krátkých programů zapsaných ve strojovém kódu, které specifikaci splňují. Evoasm obsahuje kompilátor JIT (just-in-time), který spouští generovaný strojový kód za běhu. Podporována je Intel 64-bit architektura (x86-64).
[170] https://github.com/a0rtega/metame
[171] Blíže viz https://www.radare.org/r/ a https://github.com/radare/radare2.

Částečně se tím prohodily jejich úlohy. Někteří autoři malware nyní převzali techniky obránců k detekování jejich anti-malware. Naproti tomu jsou v této práci některé techniky převzaty od malware autorů na skrytí anti-malware nástrojů před škodlivým kódem a provedení jeho analýzy.

5.5.2 Implementace maskování analytických nástrojů

Vybrané polymorfní a metamorfním enginy zamaskovaly nástroje dynamické analýzy.

peCloakCapstone – Nejprve byl mimo virtuální prostředí na fyzickém stroji zprovozněn polymorfní engine *peCloakCapstone.py*. V pracovním adresáři bylo vytvořeno speciální Pythonové 2.7 virtuální prostředí venv[172], konkrétně `morph_venv`. PeCloak vyžadoval několik Python knihoven a jejich dodání vhodným způsobem: pefile (`pip install pefile`), pydasm[173] a SecDoubleP.py. Vytváření probíhalo ve zmíněném Python oddělení venv, aby neovlivňovalo ostatní enginy. Několik analytických nástrojů od Sysinternals bylo tímto engine upravováno až byla vytvořena jejich maskovaná (polymorfně zakódovaná) varianta, která byla nadále plně funkční.

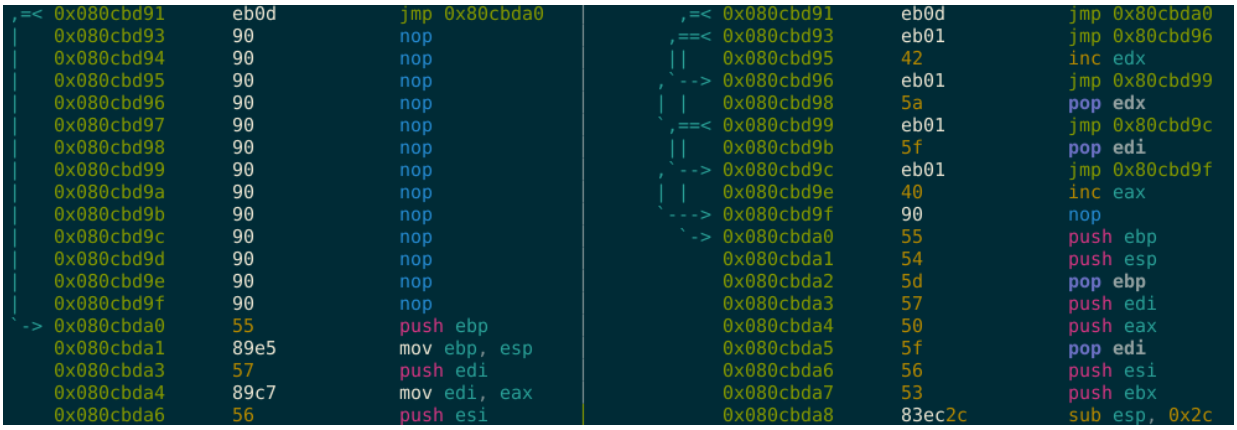
```
[*] Preserving the following entry instructions (at entry address 0x4755cb,
        [+] call 0x480584 (5)
[*] Generated Heuristic bypass of 3 iterations
[*] Generated Encoder with the following instructions:
        [+] ADD 0x1f
        [+] XOR 0x5
        [+] XOR 0x6f
        [+] ADD 0x6c
        [+] ADD 0x5
        [+] SUB 0xb6
        [+] ADD 0x3d
        [+] SUB 0x35
        [+] XOR 0x3e
[*] Encoding entire .text    section
[*] PE .text    section made writeable with attribute 0xE0000020
[*] Writing encoded data to file
[capstone] asm = u'call 0x480584'
[*] Overwriting first bytes at physical address 000749cb with jump to code
[*] Writing code cave to file
        [+] Heuristic Bypass
        [+] Decoder
        [+] Saved Entry Instructions
        [+] Jump to Restore Execution Flow
        [+] Final Code Cave (len=181):

            90909090909031f631ff9033c05131c959404851
            c9594090404483da047301875eb40485231d25a43
            90909033c0404840840434b3de4bf3f1575f242
            9090909033c090405131c9593d9420951875f353
            db5b909090b8001040005131c9595331db5b8030
            4048434b800035424a80283d424a5131c9598000
            5231d25a5331db5b802805414980286c41498030
            5331db5b8030059c9d404880281f403d87fc4800
            b59090e8d494e6ffe91be5e5ff

[*] Saving ..\procmon_1535728978_cloaked.exe
```

Obrázek 11:Ukázka maskování nástroje zakódováním instrukcí polymorfním engine

[172] https://docs.python.org/3/tutorial/venv.html
[173] https://github.com/axcheron/pydasm

metame – metamorfnímu engine *metame.py* bylo vytvořeno potřebné runtime prostředí a poté byl opět aplikován na vybrané analytické nástroje Sysinternals (Process Monitor `procmon.exe` a `procmon64.exe`, Process Explorer `procexp.exe`, Autoruns `autoruns.exe` a `autoruns64.exe`).

Ukázka nahrazení NOP instrukcí (levý sloupec) za ekvivalentní mutovaný kód s jinými instrukcemi (pravý sloupec) pomocí metamorfního engine metame. Zobrazeno v radare2.



Obrázek 12: Ukázka původního kódu programu vlevo a jeho metamorfní varianty vygenerované pomocí metame vpravo

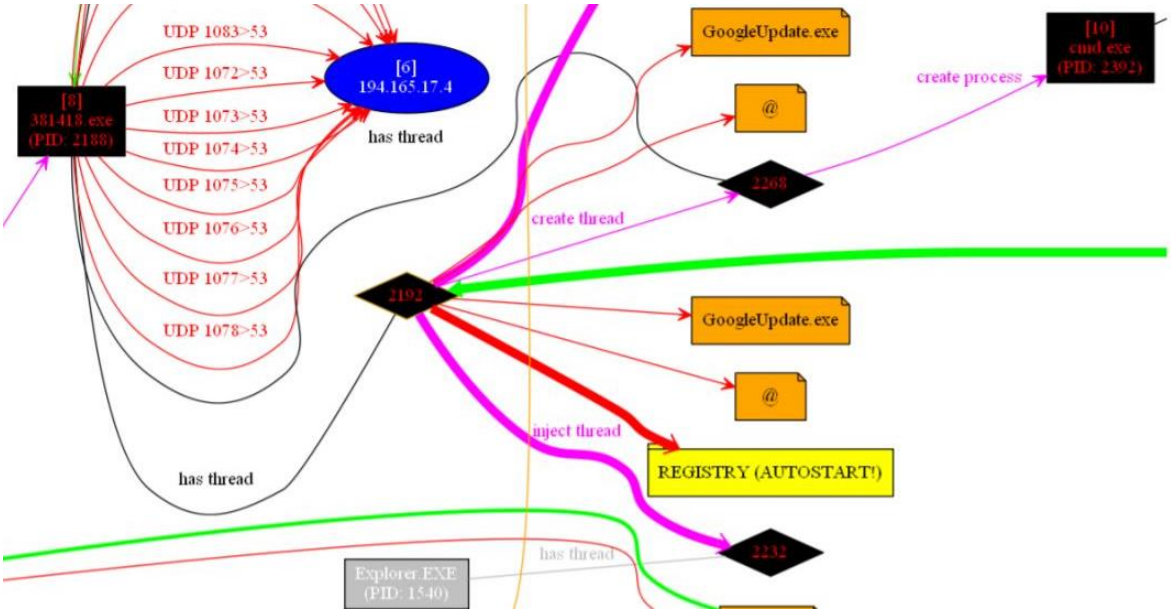
Vhodným doplněním pokročilých maskování analytických souborů proti případné detekci ze strany malware jsou také zcela základní opatření jako přejmenování názvu binárních souborů. Různé kombinace maskování byly provedeny na těchto analytických nástrojích od Sysinternals – Autoruns (`autoruns.exe`), Process Monitor (`procmon.exe`), Process Explorer (`procexp.exe`).

Vhodným doplňujícím maskovacím opatření je změna popisu běžícího procesu, tj. položky zobrazující se na anglických Windows pod položkou 'Descritpion' v Task Manageru, byla provedena pomocí editoru a de-/kompilátoru Resource Hacker[174]. Takto maskované programy byly přidány strojům Analytic Victim u variant pros#1 a #2. Způsobů, jak skrýt vybrané soubory/procesy je celá řada a často jsou použitelné postupy z penetračního testování, například použitím nástrojů MetaSploit[175] nebo MetaTwin[176].

[174] ResourceHacker. Dostupné z: http://www.angusj.com/resourcehacker/. Použitá verze byla 5.1.6 vydaná 07/2017.
[175] https://www.offensive-security.com/metasploit-unleashed/binary-payloads/
[176] http://threatexpress.com/2017/10/metatwin-borrowing-microsoft-metadata-and-digital-signaturesto-hide-binaries/

5.5.3 Vizualizace aktivity škodlivého kódu z logů maskovaných nástrojů

Vizualizaci lokálních a síťových aktivit škodlivého umožňuje nástroj ProcDOT[177]. ProcDOT zpracovává výstupní logy Sysinternals ProcessMonitoru (Procmon) a PCAP logy (Windump, Tcpdump) z kterých následně vygeneruje graf využitím sady GraphViz. Vzniklý graf propojí lokální aktivitu se síťovou, vizualizuje relevantní aktivity analyzovaného malware (rozsah detailu volitelný) a může být interaktivně procházen (např. volbou zkoumaného procesu). Na analytický virtuální stroj Analytic Victim byly proto vedle maskovaného nástroje Procmon přidán maskovaný WinDump. Autor sběr a přípravu logů pro ProcDOT automatizoval pomocí Batch skriptu[178]. Skript používá již maskované monitorovací nástroje Procmon a WinDump. Jakmile jsou uloženy logy z dynamické fáze, pak lze již rozbor zachycených událostí ve vizualizovaném grafu ProcDOTu provádět na jakémkoliv systému. Proto postačuje a je vhodné mít ProcDOT místo na analytických Windows [w7av] jen na analytickém Serveru [remn6x].



Obrázek 13: Vizualizovaný výstup nástroje ProcDOT

Jak lze vidět na obrázku (Obrázek 13), ProcDOT zobrazuje informace o souborech, ke kterým daný proces přistupoval, o klíčích registrů Windows, o síťové komunikaci a dalších důležitých parametrech. Takto lze rychle získat komplexní přehled o chování malware a zároveň rozpoznat klíčové části průběhu infekce. Jednotlivé části je možné interaktivně procházet. Navíc lze

[177] WOJNER, Christian. ProcDOT. Dostupné z: http://procdot.com/index.htm.
[178] Git repozitář této práce. https://gitlab.com/jarkovar/javeman/blob/master/tools/AnalyticVictimVM/Procmon-WinDump-ProcDOT/StartWinDump-Procmon.bat

visualizaci zaznamenané aktivity spustit v módu animace podle časové osy, což opět napomůže porozuměné průběhu infekce škodlivým kódem. V grafu lze také vyhledávat textové řetězce, což opět může pomoci nalézt například komunikaci s konkrétní doménou.[179] Tím vším je ProcDOT vhodným nástrojem při analýze škodlivého kódu v experimentech této práce.

5.5.4 Mimikry (obrana napodobením)

Fenomén zjevně využitelný, jak pro obranu, tak útok, jsou *mimikry*.[180] Termín označuje povrchní podobnost jednoho organismu – provádějícího mimikry (angl. *mimicking*) – s chováním a rysy jiného blíže nepříbuzného druhu.[181] V přírodě se mimikry vyskytují ve dvou variantách: *Obranné mimikry* skrývají přítomnost organismu před predátory a usilují o vyvolání výhodné změny v chování útočníka. *Útočné mimikry* používá predátor, který se buď vydává za jiný druh, aby přilákal kořist, anebo svou přítomnost zcela maskuje. Inspirace přírodními mimikrami pro analytický systém i zde spočívá v působení jako bezbranná kořist, která predátorský malware podněcuje k projevu útočné (škodlivé) aktivity.

MetaTwin[182] – umožňuje nahrát metadata z obyčejného programu (např. calc) na jiný program, který je tímto způsobem maskován a napodobuje z pohledu metadat původní obyčejný program.

5.5.5 Očkování klamnými známkami napadení škodlivým kódem

Analytický systém lze dále naočkovat vůči škodlivým kódům (*infekcím)* klamnými známkami (*atrapami*) přítomnosti nějakého škodlivého kódu a zmást tím skutečné infekce.[183] Dynamickou analýzu vzorku pak obohatí zjištění nakolik skutečný škodlivý kód usiluje o vyřazení jiného (konkurenčního) kódu, aniž by věděl, že vyřazuje atrapu, na kterou jsou navíc nastraženy detekční podmínky, zaznamenávající jeho snahu. Na přípravu atrap lze použít Mystique[184] – extrahuje ze vzorku seznam mutexů[185] využitelných pro „očkování“ (Python).

[179] https://m.systemonline.cz/it-security/procdot-a-density-scout.htm
[180] MALAVER, Sharron. Mimicry: The Evolution of Deceptive Defenses and Attacks.
[181] WICKLER, Wolfgang. Mimicry. Dostupné z: https://www.britannica.com/science/mimicry.
[182] http://threatexpress.com/2017/10/metatwin-borrowing-microsoft-metadata-and-digital-signaturesto-hide-binaries
[183] ZELTSER, Lenny. *Malware Vaccination for the Enterprise* [online].
[184] https://github.com/MinervaLabsResearch/Mystique.
[185] Vzájemné vyloučení: z více procesů má přístup ke sdílenému prostředku jen jeden (z angl. mutual exclusion, *mutex*).

6. Experimenty

V této kapitole je sbírkou na testování anti-evasivní úrovně ověřena schopnost prostředí skrývat svou virtuální podstatu. Následně je prostředí použito k detekci evasivího malware.

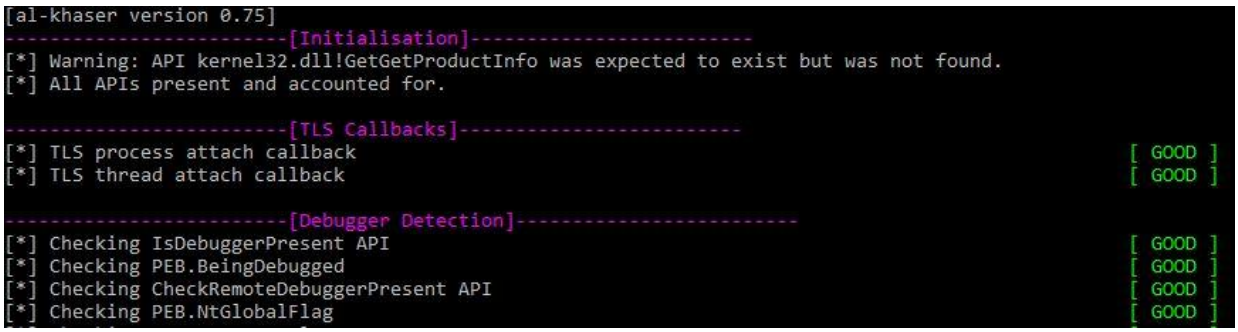
6.1 Detekce prostředí sbírkou na testování anti-evasion úrovně

Sbírka na testovnání anti-evasion úrovně, uvedená v kapitole 4.3.4, obsahuje pět programů[186]. Nejprve bude ověřena schopnost sbírky detekovat virtuální prostředí spuštěním v nevirtuálním prostředí i v nezamaskovaném virtuálním prostředí. Následným spuštěním v zamaskovaném virtuálním prostředí bude ověřena účinnost maskování. Obdobně ověříme funkčnost zamaskování analytických nástrojů.

6.1.1 Nevirtuální prostředí

Testovací programy byly nejprve spouštěny na normálním nevirtuálním prostředí Windows 10 64-bit pro ověření, co nahlásí v situaci, kdy by žádné virtuální prostředí detekovat neměly. Jednalo se o test na *false-positive*.

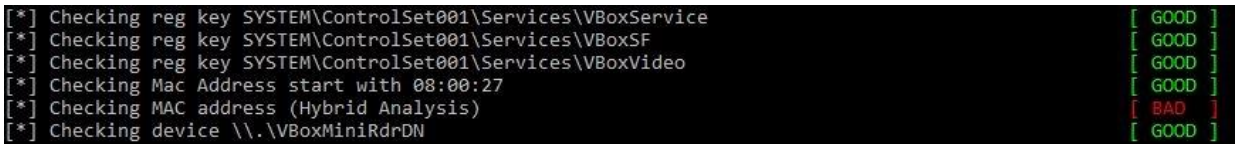
al-khaser_x64.exe – spustil celých 190 testovacích podmínek, přičemž chyboval pouze 4krát.



Obrázek 14: Výstup evasivního testovacího programu al-khaser_x64.exe

Podle MAC adresy se spletla jedna testovací podmínka na online analyzátor Hybrid Analysis (testovaný počítač měl běžnou síťovou kartu Intel).

[186] InviZzzible od firmy Check Point Software byl oficiálně dostupný jen ve formě zdrojového kódu, který bylo nutné zkompilovat ve Visual Studio 2015 C++ za přítomnosti Python 2.7. Build log obsahoval opakovaný warning u několika tříd, což by mohlo být příčinou případných nedostatků programu.



Obrázek 15: Testovací podmínka na detekci Hybrid Analysis

al-khaser_x86.exe – výsledek shodný s 64-bitovou verzí tohoto programu s jediným rozdílem, že nyní program testovací podmínku *Local Descriptor Table location* vyhodnotil jako v pořádku.

InviZzzibleSandboxEvasion.exe – testoval stopy přítomnosti populárního analytického nástroje Cuckoo. Program vygeneroval HTML report s tabulkou obsahující výsledek každé testovací podmínky, její stručný popis a krátké doporučení, jak se vyhnout její detekci anebo upozornění, že by to bylo velmi obtížné až nereálné. Náhled několika položek tabulky reportu:

Detection Name | Description	Detected | Countermeasures
UnbalancedStack Check if canaries at the top of the stack remain same after function call.	NO. Stack adjusting before function call. Kernel-mode hooking.
DelaysAccumulate Check if delays accumulation is valid using get time functions.	NO. Complex.

pafish.exe – testuje celkem 58 podmínkami přítomnost obvyklých analytických prostředí jako VMware, Bochs, Cuckoo, apod. Na nevirtuálním prostředí program spletl dvě podmínky:

```
[pafish] Sandbox traced by checking file path
[pafish] Sandbox traced by mouse inactivity
```

Na to, aby testovací podmínky přestaly hlásit domnělou detekci postačilo pouhé přesunutí programu z uživatelské složky /Downloads do jiné (zde postačilo české /Soubory) a letmý pohyb myší po dobu běhu programu. Jiné podobně jednoduché opatření představuje přejmenování souboru na náhodný název (např. jdfklbj.exe).

SEMS.exe – označil systém, na kterém byl spuštěn, za analytické prostředí a detekoval přítomnost skutečně nainstalovaného Wiresharku.

Pozn. Společnost Joe Security publikovala v roce 2017 testovací makro pro Microsoft Word určené též na detekci virtuálního prostředí.[187] Do sbírky této práce makro zahrnuto nebylo s ohledem na malou věrohodnost. Během zkoušky v normálním nevirtuálním prostředí 3 z 14 testovacích podmínek chybně určily, že byla detekována známka virtuálního prostředí (tj. *false-positive rate*[188] *FPR=21%*).

Testovací programy byly spuštěny opakovaně. Každý byl aspoň jednou spuštěn s Wiresharkem naslouchajícím na síťové kartě, aby se ověřilo, zda používají na detekci virtuálního (analytického) prostředí přístup k Internetu. K tomu bylo vypnuto několik síťových služeb Windows, aby byl omezen šum v zachyceném toku a lépe se všechno analyzovalo, např. služba SSDP (Simple Service Discovery Protocol). Výsledky ukázaly, že testovací programy lze považovat za přijatelné z hlediska false-positive a mohou být využity k ověřování úrovně anti-evasivní úrovně a zamaskování analytického virtuálního prostředí této práce.

6.1.2 Nezamaskované virtuální prostředí #0

Pro nezamaskovanou variantu, stroj [w7cv-0-guestadd], byly použity Windows 7 získané ze stránek Microsoftu pomocí akademické licence.[189] Předinstalovány byly také doplňky VirtualBox Guest Additions, které však ve virtuálním stroji zanechávají mnoho stop a evasivní kód pak snadno detekuje přítomnost virtualizace. Instalace doplňků VirtualBox je téměř doslovným opakem anti-evasivních opatření.

Snadnou detekovatelnost takového virtuálního stroje potvrdily následující výsledky experimentů:

`al-khaser_x86.exe` našel 55 příznaků virtualizace stroje, mnoho z nich se týká detekce doplňků Guest Additions. Detekované příznaky jsou uvedeny cele pro názornou ukázku práce doplňků:

```
[*] Checking NtYieldExecution
[*] Checking Number of processors in machine
[*] Checking Number of cores in machine using WMI
[*] Checking hard disk size using WMI
[*] Checking SetupDi_diskdrive
[*] Checking mouse movement
[*] Checking disk size using GetDiskFreeSpaceEx
[*] Checking if CPU hypervisor field is set using cpuid(0x1)
```

[187] https://github.com/joesecurity/pafishmacro
[188] *Study Design 101* [online]. Dostupné z: https://himmelfarb.gwu.edu/tutorials/studydesign101/formulas.html.
[189] Viz kap. §5.2.1.

```
[*] Checking hypervisor vendor using cpuid(0x40000000)
[*] Checking SerialNumber from BIOS using WMI
[*] Checking Model from ComputerSystem using WMI
[*] Checking Manufacturer from ComputerSystem using WMI
[*] Checking power capabilities
[*] Checking reg key HARDWARE\Description\System - SystemBiosVersion is set
    to VBOX
[*] Checking reg key HARDWARE\Description\System - SystemBiosDate is set to
    06/23/99
[*] Checking VirtualBox Guest Additions directory
[*] Checking file C:\Windows\system32\drivers\VBoxMouse.sys
[*] Checking file C:\Windows\system32\drivers\VBoxGuest.sys
[*] Checking file C:\Windows\system32\drivers\VBoxSF.sys
[*] Checking file C:\Windows\system32\drivers\VBoxVideo.sys
[*] Checking file C:\Windows\system32\vboxdisp.dll
[*] Checking file C:\Windows\system32\vboxhook.dll
[*] Checking file C:\Windows\system32\vboxmrxnp.dll
[*] Checking file C:\Windows\system32\vboxogl.dll
[*] Checking file C:\Windows\system32\vboxoglarrayspu.dll
[*] Checking file C:\Windows\system32\vboxoglcrutil.dll
[*] Checking file C:\Windows\system32\vboxoglerrorspu.dll
[*] Checking file C:\Windows\system32\vboxoglfeedbackspu.dll
[*] Checking file C:\Windows\system32\vboxoglpackspu.dll
[*] Checking file C:\Windows\system32\vboxoglpassthroughspu.dll
[*] Checking file C:\Windows\system32\vboxservice.exe
[*] Checking file C:\Windows\system32\vboxtray.exe
[*] Checking file C:\Windows\system32\VBoxControl.exe
[*] Checking reg key HARDWARE\ACPI\DSDT\VBOX__
[*] Checking reg key HARDWARE\ACPI\FADT\VBOX__
[*] Checking reg key HARDWARE\ACPI\RSDT\VBOX__
[*] Checking reg key SOFTWARE\Oracle\VirtualBox Guest Additions
[*] Checking reg key SYSTEM\ControlSet001\Services\VBoxGuest
[*] Checking reg key SYSTEM\ControlSet001\Services\VBoxMouse
[*] Checking reg key SYSTEM\ControlSet001\Services\VBoxService
[*] Checking reg key SYSTEM\ControlSet001\Services\VBoxSF
[*] Checking reg key SYSTEM\ControlSet001\Services\VBoxVideo
[*] Checking Mac Address start with 08:00:27
[*] Checking device \\.\VBoxMiniRdrDN
[*] Checking device \\.\VBoxGuest
[*] Checking VBoxTrayToolWndClass / VBoxTrayToolWnd
[*] Checking VirtualBox Shared Folders network provider
[*] Checking virtual box processe vboxservice.exe
```

```
[*] Checking virtual box processe vboxtray.exe
[*] Checking DeviceId from WMI
[*] Checking Mac address from WMI
[*] Checking NTEventLog from WMI
[*] Checking SMBIOS firmware
[*] Checking ACPI tables
[*] Checking reg key SOFTWARE\Microsoft\Virtual Machine\Guest\Parameters
```

al-khaser_x64.exe je 64-bitová verze předchozího programu a detekoval shodný počet příznaků.

pafish.exe detekoval 23 příznaků přítomnosti virtualizace či analýzy. Jedná se o podobné příznaky, jaké detekoval program al-khaser_x86.exe, ovšem jsou shrnuté do menšího počtu bodů:

```
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing
    VM exit
[*] Checking hypervisor bit in cpuid feature bits
[*] Checking cpuid hypervisor vendor for known VM vendors
[*] Using mouse activity
[*] Checking if disk size <= 60GB via GetDiskFreeSpaceExA()
[*] Checking if NumberOfProcessors is < 2 via raw access
[*] Checking if NumberOfProcessors is < 2 via GetSystemInfo()
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion")
[*] Reg key (HKLM\SOFTWARE\Oracle\VirtualBox Guest Additions)
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion")
[*] Reg key (HKLM\HARDWARE\ACPI\DSDT\VBOX__)
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\VBOX__)
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\VBOX__)
[*] Reg key (HKLM\SYSTEM\ControlSet001\Services\VBox*)
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate")
[*] Driver files in C:\WINDOWS\system32\drivers\VBox*
[*] Additional system files
[*] Looking for a MAC address starting with 08:00:27
[*] Looking for pseudo devices
[*] Looking for VBoxTray windows
[*] Looking for VBox network share
[*] Looking for VBox processes (vboxservice.exe, vboxtray.exe)
[*] Looking for VBox devices using WMI
```

InviZzzibleSandboxEvasion.exe detekující přítomnost Cuckoo Sandboxu nenašel žádný příznak, protože se o Cuckoo Sandbox nejedná.

SEMS.exe detekoval 42 příznaků přítomnosti virtualizace nebo analýzy. Příznaky byly podobné jako v předchozích případech, bylo detekováno 1 jádro CPU, velikost disku menší než 60 GB,

VboxBIOS, Windows registry klíče VitualBoxu tykající se HW i SW, MAC adresa VirtualBoxu a zejména mnoho souborů i procesů zanesených do stroje doplňky VirtulBox Guest Additions.

6.1.3 Mírně maskované virtuální prostředí #1

Virtuálnímu stroji Victim [w7cv-1-fresh] byl nainstalován systém Windows 7 64-bit. Před instalací operačního systému byly v nastavení VirtualBoxu upraveny parametry virtuálního stroje tak, aby lépe odpovídaly podobě reálného nevirtuálního počítače. Byly nastaveny následující HW parametry:

CPU: 2 jádra

HDD: 82 GB

RAM: 2048 MB

Video RAM: 128 MB

Síťové připojení: Internal, vnitřní virtuální síť javeman

USB port: vypnut

Operační systém Windows byl nainstalován standardním způsobem a bez VirtualBox doplňků. Operační systém byl upraven tak, aby byl přívětivější ke škodlivému kódu pomocí následujících změn: UAC (User Access Control): vypnuto

Windows Firewall: vypnuto

Windows Update: vypnuto

Windows Telemetry: vypnuto

Zároveň mělo vypnutí služeb výše příznivý efekt na přehlednější síťové logy při monitorování provozu na Analytic Serveru ve Wiresharku, INetSimu a dalších případných nástrojích (Burp). Simulace internetového připojení stroji Victim pomocí simulované virtuální sítě javenet byla zprovozněna nastavením statické IP adresy a nasměrováním provozu na Analytic Server:[190]

IP 192.168.0.2

Maska: 255.255.255.0

Výchozí brána a DNS: 192.168.0.1

Také bylo provedeno opotřebování systému Windows běžným užíváním. Byl přidán nový uživatel systému Windows, doinstalovány programy (Chrome, PSPad), zástupce 'This Computer' byl přidán na plochu, byly vytvořeny nové soubory, některé z nich byly následně smazány do

[190] Dle schéma virtuální sítě javenet v kap. §5.2.

koše a byly přidány záložky do prohlížeče Internet Explorer. Upravený snímek virtuálního stroje byl uložen jako výchozí a následně byly spuštěny evasivní testovací programy.

6.1.3.1 Mírně maskované virtuální prostředí #1 bez analytických nástrojů

al-khaser_x86.exe našel již jen 19 příznaků virtualizace stroje. Jde o podmnožinu příznaků detekovaných v nezamaskovaném virtuálním prostředí #0. Detekované příznaky jsou:

```
[*] Checking hard disk size using WMI
[*] Checking SetupDi_diskdrive
[*] Checking mouse movement
[*] Checking if CPU hypervisor field is set using cpuid(0x1)
[*] Checking hypervisor vendor using cpuid(0x40000000)
[*] Checking SerialNumber from BIOS using WMI
[*] Checking Model from ComputerSystem using WMI
[*] Checking Manufacturer from ComputerSystem using WMI
[*] Checking power capabilities
[*] Checking reg key HARDWARE\Description\System - SystemBiosVersion is set
    to VBOX
[*] Checking reg key HARDWARE\Description\System - SystemBiosDate is set to
    06/23/99
[*] Checking reg key HARDWARE\ACPI\DSDT\VBOX__
[*] Checking reg key HARDWARE\ACPI\FADT\VBOX__
[*] Checking reg key HARDWARE\ACPI\RSDT\VBOX__
[*] Checking Mac Address start with 08:00:27
[*] Checking DeviceId from WMI
[*] Checking Mac address from WMI
[*] Checking SMBIOS firmware
[*] Checking ACPI tables
```

al-khaser_x64.exe detekoval shodných 19 příznaky virtualizace jako výše uvedená 32-bitová verze programu.

pafish.exe detekoval 11 příznaků přítomnosti virtuálního stroje.

```
[*] Checking the difference between CPU timestamp counters (rdtsc) forcing
    VM exit
[*] Checking hypervisor bit in cpuid feature bits
[*] Checking cpuid hypervisor vendor for known VM vendors
[*] Reg key (HKLM\HARDWARE\Description\System "SystemBiosVersion")
[*] Reg key (HKLM\HARDWARE\Description\System "VideoBiosVersion")
[*] Reg key (HKLM\HARDWARE\ACPI\DSDT\VBOX__)
[*] Reg key (HKLM\HARDWARE\ACPI\FADT\VBOX__)
[*] Reg key (HKLM\HARDWARE\ACPI\RSDT\VBOX__)
[*] Reg key (HKLM\HARDWARE\DESCRIPTION\System "SystemBiosDate")
```

```
[*] Looking for a MAC address starting with 08:00:27
[*] Looking for VBox devices using WMI
```

InviZzzibleSandboxEvasion.exe detekující přítomnosti Cuckoo Sandboxu opět nenašel žádný příznak, protože se o Cuckoo Sandbox nejedná.

SEMS.exe detekoval 8 příznaků VirtualBoxu podle virtualizovaných hardwarových registračních klíčů, BIOSu a MAC adresy.

```
[*] VirtualBox Detected (HKLM\HARDWARE\ACPI\DSDT\VBOX__)
[*] VirtualBox Detected (HKLM\HARDWARE\ACPI\FADT\VBOX__)
[*] VirtualBox Detected (HKLM\HARDWARE\ACPI\RSDT\VBOX__)
[*] VirtualBox Detected (vboxBios1)
[*] VirtualBox Detected (vboxBios2)
[*] VirtualBox Detected (vboxBios4)
[*] VirtualBox Detected (Enum)
[*] VirtualBox Detected (MAC)
```

6.1.3.2 Mírně maskované virtuální prostředí #1 s nezamaskovanými nástroji

Do obnoveného výchozího snímku virtuálního stroje [w7cv-1-fresh] byly dodány následující programy:

```
autoruns.exe
procmon.exe
procexp.exe
```

Tyto programy, sloužící jako analytické nástroje, byly spuštěny. Následně byly provedeny testy sbírkou na testování anti-evasion úrovně. Záměrem bylo ověřit, zda budou nově přidané analytické nástroje detekovány.

al-khaser_x86.exe a al-khaser_x64.exe detekovaly kromě 19 příznaků odhalených již v předchozí variantě bez analytických nástrojů také následující 3 příznaky navíc:

```
[*] Checking RDTSC which force a VM Exit (cpuid)
[*] Checking process of malware analysis tool: autoruns.exe
[*] Checking process of malware analysis tool: procmon.exe
```

Je vidět, že nově přidané analytické nástroje byly zaznamenány.

pafish.exe detekoval shodných 11 příznaků přítomnosti virtuálního stroje jako v případě bez analytických nástrojů, analytické nástroje nezaznamenal.

InviZzzibleSandboxEvasion.exe rovněž nedetekoval nové známky přítomnosti analýzy.

SEMS.exe detekoval 9 příznaků přítomnosti virtuálního stroje, z nichž 8 příznaků detekce VirtualBoxu bylo shodných s předchozí variantou bez analytických nástrojů. Navíc byl detekován Proces Explorer – jeden ze spuštěných Sysinternals nástrojů.

```
[*] Process Explorer detected
```

6.1.4 Zamaskované virtuální prostředí #2 se zamaskovanými analytickými nástroji

Zamaskování varianty #2 bylo provedeno opět vytvořením stroje Victim jako v předchozí variantě #1, avšak před prvním spuštěním stroje byly provedeny úpravy dle kapitoly 5.4 (Rozšíření analytického prostředí o anti-evasion opatření).

Do obnoveného výchozího snímku virtuálního stroje [w7cv-2-antievasion] byly dodány stejné analytické nástroje jako v předchozí variantě, ovšem po maskovacích úpravách.[191] Nejprve byl editací sekce ve spustitelném souboru nástroje změněn popisek ‚Description', který by po spuštění odhaloval běžící proces. Poté byl zamaskován metamorfním engine metame.py a následně přejmenované na nepředvídatelné názvy:

```
metame_masked_snurotuA.exe
metame_masked_mncorP.exe
metame_masked_xpcorP.exe
```

Tyto programy byly před zahájením testování anti-evasivní úrovně spuštěny. Cílem bylo ověřit, zda zamaskované analytické nástroje již nebudou detekovány.

al-khaser_x86.exe a al-khaser_x64.exe detekovaly 8 příznaků, z nichž 7 příznaků bylo detekovaných ve verzi bez analytických nástrojů, navíc byl detekován následující příznak:

```
[*] Checking RDTSC which force a VM Exit
```

[191] Viz kap. §5.6.4.

Oproti verzi s nezamaskovanými analytickými nástroji nebyly detekovány programy autoruns.exe a procmon.exe. Je vidět, že maskování analytických nástrojů testovací programy zmátlo a zamaskované analytické nástroje nebyly zaznamenány.

pafish.exe a InviZzzibleSandboxEvasion.exe nezaznamenaly zamaskované analytické nástroje. To je očekávaný výsledek, jelikož nezaznamenaly ani v přechozí variantě nezamaskované analytické nástroje.

SEMS.exe detekoval shodných 5 příznaků, z nichž 4 příznaky byly detekce virtualizace.

6.1.5 Vyhodnocení detekce prostředí sbírkou na testování anti-evasion úrovně

Nejprve byla otestována *false-positive* věrohodnost sbírky na testování anti-evasion úrovně spuštěním na nevirtuálním stroji. Nedošlo k téměř žádným detekcím příznaků virtualizace nebo analytického prostředí tohoto nevirtuálního stroje, což je správný výsledek.

Poté byla otestována *true-positive* věrohodnost sbírky testovacích programů spuštěním v nezamaskovaném virtuálním prostředí [w7cv-0-guestadd] bez analytických nástrojů. Testovací program InviZzzibleSandboxEvasion.exe nedetekoval žádný příznak, protože je zaměřen pouze na detekci jiného řešení. Jinak u všech ostatních testovacích programů došlo k výraznému nárůstu počtu detekce příznaků virtualizace nebo analýzy, což je opět správný výsledek potvrzující schopnost programů detekovat virtuální stroj.

Detekci nižšího počtu příznaků virtualizace a analytického prostředí lze interpretovat jako úspěšné zamaskování odpovídajícího virtuálního stroje.

Zamaskování virtuálního stroje [w7cv-1-fresh] spočívalo v použití vlastnoručně vytvořeného stroje Victim instalací Windows bez doplňků VirtualBox Guest Additions. To s několika upravenými parametry virtuálního stroje (např. větší pevný disk) vedlo k výraznému snížení počtu detekcí virtuálního stroje u všech relevantních testovacích programů.

Přidáním nezamaskovaných analytických nástrojů v případě virtuálního stroje [w7av-1-fresh] se počet detekovaných příznaků zvýšil. Tím bylo ověřeno, že sbírka programů na testování anti-evasion úrovně umí rozpoznat analytické nástroje. Testovací sbírku tedy lze použít i k ověření funkčnosti maskování analytických nástrojů.

Záměnou nezamaskovaných analytických nástrojů za zamaskované analytické nástroje ve virtuálním stroji [w7av-1-fresh] opět došlo k snížení počtu detekovaných příznaků. Nejméně detekovaných příznaků pak připadá nejvíce zamaskovaným variantám virtuálních strojů [w7cv-2-antievasion][w7av-2-antievasion].

Testy anti-evasion úrovně postupně maskovaných variant prostředí této práce	**Testovací evasivní programy**				
	Al-khaser_x86	Al-khaser_x64	Pafish	InviZzzible	Sems
Celkový počet testovacích podmínek na detekci sandboxu	190	190	58	20	–
Počet detekcí sandboxu na **nevirtuálním prostřed**í	4	3	2	0	2
Varianty virtuálních strojů **Clean Victim VM** bez analytických nástrojů uvnitř					
Počet detekcí na velmi **nezamaskovaném virtuálním** stroji [w7cv-0-guestadd]	55	55	23	0	42
Počet detekcí na **mírně maskovaném virtuálním** stroji [w7cv-1-fresh]	19	19	11	0	8
Počet detekcí na **zamaskovaném virtuálním** stroji [w7cv-2-antievasion]	8	8	4	0	5
Varianty virtuálních strojů **Analytic Victim VM** s analytickými nástroji uvnitř					
Počet detekcí na **mírně maskovaném virtuálním** stroji [w7av-1-fresh] s **nezamaskovanými** analytickými nástroji	22	22	11	0	8

Počet detekcí na **mírně maskovaném virtuálním** stroji [w7av-1-fresh] se **zamaskovanými** analytickými nástroji	20	20	11	0	8
Počet detekcí na **zamaskovaném virtuálním** stroji [w7av-2-antievasion] se **zamaskovanými** analytickými nástroji	8	8	4	0	5

Tabulka 6: Výsledky detekovatelnosti variant prostředí evasivními testovacími programy

Výše uvedená tabulka testů anti-evasion úrovně potvrzuje, že se zamaskované varianty prostředí pomocí implementovaných opatření z hlediska detekovatelnosti blíží nevirtuálnímu prostředí.

6.2 Transport vzorků v prostředí a automatizace řízení analýzy

Postup manuální i automatizované analýzy začíná přesunem vzorků do analytického prostředí na stroj Clean nebo Analytic Victim.

6.2.1 Manuální postup přesunu vzorku

1. Nejprve je třeba připravit si vzorek škodlivého kódu, který bude analyzován, například stažením z (internetových) sbírek popsaných v kapitole §4.3. Vzorek škodlivého kódu ponechat v obvyklé zazipované a zaheslované podobě. Pokud je záměrem na infikovatelný stroj nahrát bezpečné soubory (např. evasion testovací program nebo nový analytický nástroj) je možné s nimi pracovat rovnou ve spustitelné formě .exe souboru.
2. Vzorek přesunout do složky sdílené s VM Collection. Stačí k tomu pouze oprávnění ke čtení, proto lze vybrat složku `/root/Javeman/VirtualBoxMachines/shared_ro/`.
3. Ve VirtualBoxu spustit VM Collection. Vzorek připravený ve sdílené složce na adrese /mnt/shared_ro přesunout a přejmenovat pomocí příkazu `cp /mnt/shared_ro/nazev-vzorku.exe sample.exe`.

4. Vzorek přesunout z VM Collection na VM Analytic Server pomocí `scp` příkazu. Nejprve ve VirtualBoxu spustit VM Analytic Server a poté se vrátit se do stroje VM Collection. Pro přesunutí vzorku zadat příkaz `scp sample.exe remnux@192.168.0.1:~/Samples`. A následně při dotazu na heslo vyplnit: `Javeman2018`
5. Nyní je vzorek přesunut do vniřní části virtuálního prostředí a lze vypnout VM Collection. Tím je virtuální analytické prostředí strojů Victim a Analytic Server dočasně zcela izolováno.
6. Vzorek přesunout pomocí simulované internetové sítě ze stroje VM Analytic Server na zvolenou verzi VM Victim. Na VM Analytic Server je simulovaná internetová síť pomocí nástroje `inetsim` již spuštěna. Připravený vzorek podstrčit do sítě pomocí příkazu: `sudo mv ~/Samples/sample.exe /var/lib/inetsim/http/fakefiles/sample.exe`.
7. Ve VirtualBoxu spustit vybraný stroj Victim Clean nebo Victim Analytic. Spustit libovolný internetový prohlížeč a z adresy `http://service.org/malware/sample.exe`, která je simulována virtuálním strojům uvnitř virtuální sítě, stáhnout již připravený vzorek. V prohlížeči Internet Explorer je odkaz připravený v záložce. Při ukládání soubor přejmenovat na původní název a změnit příponu na původní hodnotu. Archiv se vzorkem rozbalit, přičemž obvyklé heslo zazipovaných vzorků je '`infected`' nebo '`malware`'.
8. Nyní je vzorek škodlivého kódu připraven na VM Victim analytického prostředí. V něm je možné vzorek spustit a prozkoumat metodami analýzy škodlivého kódu.

6.2.2 Automatizace přesunu vzorku a řízení analýzy z vně prostředí

Výše uvedený postup přesunu vzorku z Kali Host na Victim VM je samozřejmě automatizovatelný, čehož je v této práci dosaženo zejména naprogramováním Bash skriptů. Velmi se osvědčil obsáhlý příkaz `vboxmanage`[192] sloužící k ovládání VirtualBoxu a virtuálních strojů z příkazové řádky. Spuštění VM lze tudíž řídit z příkazové řádky na Kali Host pomocí `vboxmanage startvm`, například Collection VM se spustí zadáním `vboxmanage startvm "collect"`. K vypnutí, pozastavení a probuzení VM pak slouží příkazy `vboxmanage controlvm`.

Standardním postupem, jak vkládat příkazy do Collect VM zvenčí z Kali Host by bylo pomocí `vboxmanage guestcontrol`. Konkrétně by přesun vzorku mezi stroji provedlo: `vboxmanage guestcontrol collect exec --image /bin/sh --username root --wait-exit --wait-stdout --wait-stderr -- -c 'scp sample.exe remnux@192.168.0.1:~/Samples'.`

[192] VirtualBox User Manual – Chapter 8. VBoxManage. Dostupné z: https://www.virtualbox.org/manual/ch08.html.

Pro účely této práce není tento postup vhodný, protože pro `guestcontrol` jsou potřeba doplňky Guest Additions. Pouze u nezamaskované klasické varianty prostředí Javeman #0 "vboxadd" by tak bylo možné řídit tímto způsobem stroje Victim VM, které mají jako jediné zmíněné doplňky demonstrativně nainstalované. Právě výskyt doplňků Guest Additions zásadně znesnadňuje až znemožňuje zamaskovat existenci virtuálního prostředí před detekcí škodlivým kódem. *Zamaskované varianty analytického prostředí vyžadují postup bez jakýchkoliv Guest Additions.*

V práci byl proto použit postup, který se obejde bez Guest Additions a totiž řízení sadou příkazů:

```
vboxmanage controlvm keyboardputscancode | keyboardputstring | keyboardputfile
```

Nejméně již VirtualBox 3.x z roku 2009[193] obsahoval funkci keyboardputscancode simulující stisky kláves dovnitř Guest VM pomocí hodnot řídících znaků kláves (angl. scancodes) zadaných v hexadecimální soustavě. Postupně vzniklo několik usnadnění pro vkládání znaků a řetězců založených na této funkci, ovšem jednalo se o návody a projekty mimo standardní VirtualBox.[194] Až teprve od verze VirtualBox 5.2.0 byla tato funkce simulující stisky kláves z vně virtuálního prostředí dovnitř do virtuálních strojů doplněna o dvě další funkce, keyboardputstring a keyboardputfile, které umožňují do virtuálních strojů vkládat ASCII řetězce. Relevantní změna (changeset 64906[195]) byla provedena v prosinci 2016 a commit v SVN repozitáři VirtualBox obsahuje tabulku mapování hex řídících kódů rozložení standardní US klávesnice na ASCII znaky.

Klávesa	**Hex**	Poznámka
<Enter>	**1c**	
<Enter><released>	**9c**	0x1c + 0x80

Tabulka 7: Mapování řídících znaků klávesy Enter pro funkci keyboardputscancode

Ukázka implementace automatizovaného vložení vzorku a řízení virtuálních strojů Collection a Analytic Server naprogramovaným Bash skriptem z fyzického Kali: `submit-sample.sh`[196]

```
#!/bin/bash
### This program moves samples from COL VM /shared_ro to REM VM
### over simulated network services. Launch on Kali. ./submit-sample.sh <filename>
vboxmanage startvm remn6x
```

[193] VirtualBox User Manual 3.0.8. Dostupné z: ftp://priede.bf.lu.lv/pub/Utilities/VirtualBox/UserManual-3.0.8.pdf
[194] Například návod z roku 2011: Dostupné z: https://www.halfdog.net/Misc/TipsAndTricks/VirtualBox.html.
[195] VBoxManage: limit keyboardputscancode 'typing' rate. Simple commands to 'type' ASCII chars. Dostupné z: https://www.virtualbox.org/changeset/64906/vbox/trunk/src/VBox/Frontends/VBoxManage/VBoxManageControlVM.cpp
[196] Git repozitář této práce. https://gitlab.com/jarkovar/javeman/blob/master/control/submit-sample.sh

```
vboxmanage startvm collect; sleep $WAIT;
### Start a script inside COL VM that sends file to REM
vboxmanage controlvm collect keyboardputstring ./move.sh $1 #Move sample to REM
vboxmanage controlvm collect keyboardputscancode 1c 9c #Input <Enter> to COL VM
sleep $WAIT; #Time to simulate above keystrokes and execute inside script
### Sample moved. Now poweroff COL VM until needed to receive report from REM.
vboxmanage controlvm collect poweroff
... ### Initial Static Analysis on REM
vboxmanage controlvm remn6x keyboardputstring viper open -f /home/remnux/Samples/$1
vboxmanage controlvm collect keyboardputscancode 1c 9c #Input <Enter> to REM VM
...
```

Jak lze vidět z výše uvedeného kódu je simulací kláves z vně stroje Collection volán uvnitř stroje v Ash shellu[197] skript `move.sh`, který transportuje vzorek na Analytic Server. Jde o zefektivnění, aby stačilo simulovat vložení jen několika znaků. Ukázka části programu `move.sh`:

```
#!/bin/ash
### Note: Alpine Linux has no BASH, only ASH shell. Default $1: sample.zip
/etc/init.d/sshd restart
FILENAME=""; if [ -z "$1" ]; then FILENAME=sample.zip ; else FILENAME=$1; fi
scp -i ~/.ssh/col_hss_asr_di /mnt/shared_ro/$FILENAME remnux@192.168.0.1:~/Samples
```

Obdobným způsobem jsou automatizované opakované kroky analýzy na strojích Server a Victim.

6.3 Analýza vzorků škodlivého kódu

6.3.1 Škodlivý kód těžící kryptoměnu „CryptoMining Malware“

Tento typ škodlivého kódu je vhodným zástupcem evasivního malware na experimenty, protože má jako cíl zůstat dlouho nedetekován a mezitím využívat infikovaný stroj k těžbě kryptoměny.

6.3.1.1 WaterMiner

Například evasivní miner kryptoměny Monero, WaterMiner[198] sleduje, jestli není v systému, na kterém běží, otevřeno okno nástroje zobrazujícího spuštěné procesy. Jmenovitě sleduje přítomnost aplikací, kterými by uživatel mohl zpozorovat jeho škodlivou aktivitu:

[197] Alpine Linux obsahuje tzv. Almquist Shell (ash). Manuálové stránky dostupné z: https://linux.die.net/man/1/ash
[198] Minerva Labs Research Team. WaterMiner – a New Evasive Crypto-Miner. Dostupné z: https://blog.minerva-labs.com/waterminer-a-new-evasive-crypto-miner.

Windows Task-Manager (pouze jeho anglické a ruské verze), TaskManager, AnVir[199] a ProcessHacker. Vzorek obsahuje evasivní techniky, ovšem dle firmy Minerva je autorem začátečník, jenž se sám prozradil na sociální síti VK.com.[200] a škodlivý kód byl rychle eliminován.

6.3.2 Automatizovaná analýza „acf6aade8ed9e7d1aea8c0c9f377a243.zip“

MD5: acf6aade8ed9e7d1aea8c0c9f377a243

Originální zdroj vzorku:
https://mcfp.felk.cvut.cz/publicDatasets/CTU-Malware-Capture-Botnet-347-1/

Soubor: acf6aade8ed9e7d1aea8c0c9f377a243.zip

(Heslo zip archivu: infected)

Zdroj vzorku (HA): https://www.hybrid-analysis.com/download-sample/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e

HA Report: https://www.hybrid-analysis.com/sample/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e?environmentId=2

VT Report: https://www.virustotal.com/en/file/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e/analysis/

Vzorek škodlivého kódu nazvaného CoinMiner byl zachycen dne 4. dubna 2018 v honeypotu projektu Stratosphere IPS[201]. CoinMiner je podle příspěvku Microsoftu s Threat Intelligence přehledem hrozeb výrazně rozšířený zejména v Evropě.[202]

V analytickém prostředí této práce proběhla automatizovaná analýza pomocí autorem napasaných Bash programů, které analýzu na vytvořených virtuálních strojích řídí z vně tohoto prostředí pomocí simulovaných stisků kláves[203]: `./store-sample.sh` a `./submit-sample.sh`. Automatizovaná analýza ukazála, že vzorek je vhodné podrobně prozkoumat manuální analýzou.

6.3.3 Manuální analýza „acf6aade8ed9e7d1aea8c0c9f377a243.zip“

MD5: acf6aade8ed9e7d1aea8c0c9f377a243

Zdroj vzorku (HA): https://www.hybrid-analysis.com/download-sample/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e

[199] Ruskojazyčná obdoba Task Manageru.
[200] CIMPANU, Catalin. WaterMiner Malware Author Can't Keep His Mouth Shut on Social Media. Dostupné z: https://www.bleepingcomputer.com/news/security/waterminer-malware-author-cant-keep-his-mouth-shut-on-social-media/.
[201] GARCIA, Sebastian. Malware Capture Facility Project. Dostupné z: https://stratosphereips.org.
[202] Virus, malware, and threat descriptions - Windows Defender Security Intelligence. Dostupné z: https://www.microsoft.com/en-us/wdsi/threats?id=1.
[203] Podrobný popis postupu v kap. §6.2.2.

```
Soubor: acf6aade8ed9e7d1aea8c0c9f377a243.zip
```

```
HA Report: https://www.hybrid-analysis.com/sample/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e?environmentId=100
Původní VT Report (46/67 detekcí malware): https://www.virustotal.com/#/file/7489a1b748cba275888ada88ae1cb9d21cd00d3bbdc4eec8dd11a7e8e7345f6e/detection
```

Analýza proběhla na stroji Clean Victim [w7cv-2]. Vzorek po svém spuštění, vytvoří běžící spustitelný program spyxx_amd64.exe, který je viditelný ve Windows Task Manageru. Podle VirusTotal reportu s výsledky skenu antiviry lze nabýt dojmu, že jedná o Bitcoin Miner. Slovo „bitcoin" uvádí celkem 9 AV v názvu škodlivého kódu (např. F-Secure: *Trojan.BitCoinMiner.DQ*).

Analýza byla zahájena obdobně jako v předchozím případě z fyzického Kali příkazem: `./submit-sample.sh acf6aade8ed9e7d1aea8c0c9f377a243.zip` Příkaz ovšem sloužil pouze k usnadnění přesunu vzorku do analytického prostředí a dále byla analýza již řízena manuálně.

Záměrnou nepřítomnost analytických nástrojů u analýz na stroji Clean Victim nahrazuje možnost provést z vně virtuálního prostředí export jeho operační paměti (*memory dump*)[204] do souboru ELF[205]. VirtualBox používá formát EFL na soubory virtuálního stroje vytvořené `vboxmanage debugvm`.[206] Soubor s exportovanou pamětí lze dále přesunout zpět do prostředí na stroj Analytic Server a tam paměť prozkoumat nástroji forenzní analýzy jako *Volatility* nebo *Rekall*. Zadaním následujícího příkazu na fyzickém Kali Host byl z vně virtuálního stroje Victim proveden jeho memory dump:

```
vboxmanage debugvm "w7cv-2-antievasion" dumpvmcore --filename w7cv2.elfdump
# gdb --core w7cv2.elfdump # dump memory [phys-mem-file] 0x0 [size vm-memory]
```

Nyní bylo potřeba vyextrahovat ze souboru .elfdump první LOAD sekci, kde je uložena reference na hlavní paměť. Získání správného offsetu je možné provést použitím `objdump`: [207]

```
$ objdump -h w7cv2.elfdump | egrep -w "(Idx|load1)"

Idx Name       Size      VMA               LMA               File off  Algn
  1 load1      40000000  0000000000000000  0000000000000000  00000720  2**0
```

[204] FORTUNA, Andrea. How to extract a RAM dump from a running VirtualBox machine. Dostupné z: https://www.andreafortuna.org/dfir/forensics/how-to-extract-a-ram-dump-from-a-running-virtualbox-machine/.
[205] ELF-64 Object File Format. Dostupné z: http://downloads.openwatcom.org/ftp/devel/docs/elf-64-gen.pdf.
[206] VirtualBox VM core format. Dostupné z: https://www.virtualbox.org/manual/ch12.html#ts_guest-core-format.
[207] FORTUNA, Andrea. How to extract a RAM dump from a running VirtualBox machine. Dostupné z: https://www.andreafortuna.org/dfir/forensics/how-to-extract-a-ram-dump-from-a-running-virtualbox-machine/.

V souboru .elfdump začíná memory dump na offsetu **0x720** a čítá **0x40000000** bytů (1024MB). Zbývá vyextrahovat oblast RAM. To je proveditelné „ořezáním“ prvních bajtů souboru paměti:

```
size=0x40000000;off=0x720; head -c $(($size+$off)) w7cv2.elfdump | \
tail -c +$(($off+1)) > w7cv2.raw
```

Vzniklý soubor w7cv2.raw již obsahuje obraz RAM paměti, který lze analyzovat `Volatility`:

```
$ volatility -f w7cv2.raw imageinfo
Volatility Foundation Volatility Framework 2.6
```

Nyní byl tedy soubor w7cv2.raw s obsahem operační paměti běžícího infikovaného Victim stroje manuálně poslán postupem z kapitoly 5.2. (*Transport vzorků*) dovnitř analytického prostředí na stroj Analytic Server [remn6x] a prozkoumán nástrojem Volatility. Vzorek škodlivého kódu okamžitě po spuštění smazal svůj původní spustitelný soubor a místo toho vytvořil jiný ve složce `\ProgramData\WindowsPerformanceRecorder\` a dalších pomocné DLL, jak ukázala analýza obsahu operační paměti stroje Victim pomocí Volatility. Hledáním podezřelých procesů příkazy `pslist, psxview, malfind, dllist, connscan`. pluginů Volatility byl odhalen `PID 2788: spyxx_amd64.exe9f377a243.exe`, který představuje hlavní skrytý proces škodlivého kódu:

```
remnux@remnux:~/Samples$ vol.py -f w7cv2.raw --profile=Win7SP1 ldrmodules -p 2788
Volatility Foundation Volatility Framework 2.5
Pid  Process        Base      InLoad InInit InMem MappedPath
-------- -------------------- ---------- ------ ------ ----- ----------
2788 acf6aade8ed9e7 0x73bd0000 True  True   True  Windows\System32\uxtheme.dll
2788 acf6aade8ed9e7 0x00400000 True  False  True \ProgramData\WindowsPerfor
manceRecorder\spyxx_amd64.exe9f377a243.exe
2788 acf6aade8ed9e7 0x76f40000 True   True  True \Windows\System32\usp10.dll
```

Soubor EXE a podezřelé DLL byly vyextrahovány (`procdump, dlldump`) a popsány hash otisky.

```
remnux@remnux:~/Samples$ vol.py -f w7cv2.raw --profile=Win7SP1x86 dlldump -
p 2788 -b 0x003e0000 -D .
Volatility Foundation Volatility Framework 2.5
Process(V) Name                Module Base Module Name          Result
---------- -------------------- ----------- -------------------- ------
0x84fb4030 acf6aade8ed9e7       0x0003e0000 UNKNOWN              OK: module
.2788.7f9b4030.3e0000.dll
```

Zatímco u podezřelého DLL se přítomnost škodlivého kódu sdílením do VirusTotalu nepotvrdila,

```
module.2788.7f9b4030.3e0000.dll
MD5: 2c93171059620a4b4e51c879e8ebe384
SHA256: e1e439ad7b2cf08fc68f1ce0944ebc4b5910a3692b67e2004e951f841375d9d8
VT Report (0 / 57 označení za malware): https://www.virustotal.com/#/file/
e1e439ad7b2cf08fc68f1ce0944ebc4b5910a3692b67e2004e951f841375d9d8/detection
```

u objeveného EXE skrytého procesu škodlivého kódu se naopak potvrdila velmi jednoznačně. K původnímu analyzovaném souboru vzorku škodlivého kódu, který se po spuštění okamžitě smaže a nahradí skrytým, tak byl zjištěn dodatečný údaj sloužící k rozpoznání škodlivého kódu. Analýza vzorku a sdílení vyextrahovaného skrytého EXE včetně hash otisků s komunitami analytiků skrze VirusTotal upřesnila, že malware ve skutečnosti těží kryptoměnu Monero a obsahuje kód těžebního programu XMRig[208]. Oproti původnímu vzorku je tento skrytý EXE detekován výrazně méně AV. Zjištěný artefakt vzorku z analýzy autorem této práce, indikátory kompromitace IOC a jejich sdílení:

```
spyxx_amd64.exe9f377a243.exe (vyextrahováno jako executable.2788.exe)
MD5: 844de22842b7ffa31f8857ef43f540e7
SHA256: c89900af226681ad08d3d174b1881bd25e84bd4a161c921884058691d7e735c7
VT Report (26 / 67 označení za malware): https://www.virustotal.com/#/file/
c89900af226681ad08d3d174b1881bd25e84bd4a161c921884058691d7e735c7/detection
```

[208] KESSEM, Limor. XMRig: Father Zeus of Cryptocurrency Mining Malware?. Dostupné z: https://securityintelligence.com/xmrig-father-zeus-of-cryptocurrency-mining-malware/.

7. Závěr

Výsledné provedené analýzy škodlivého kódu potvrzují, že prostředí umožňuje analýzu škodlivého kódu, který uplatňuje techniky vyhýbání se analýze ve virtuálním prostředí. Významnou výhodou analýz této práce je, že proběhly v nově vzniklém analytickém prostředí, na které se z principu škodlivý kód nemohl při detekcích a evasivních technikách zaměřit cíleně jako tomu činí u populárních online malware analyzátorů, stejně jako nemohl provést průzkum prostředí zevnitř. Škodlivému kódu zbývají při vyhýbání se analýzám v prostředí Javeman pouze obecné metody a proti nejčastějším evasivním technikám byla implementována anti-evasivní opatření. Úspěšnost použitých opatření byla ověřena a potvrzena speciálními evasivními testovacími programy. Celkem bylo při vývoji variant prostředí, testování detekce a analýze škodlivého kódu vyvinuto 7 různých virtuálních strojů s Windows nebo Linux.

Experimenty s analýzami vzorků škodlivého kódu tzv. „live malware“ demonstrovaly praktické použití této práce. Nadále se však autor domnívá, že uplatnění najde zejména u zájemců o malware analýzu a junior malware analytiků. Použití v bezpečnostní komunitě populární distribuce Kali Linux jako základu pod virtualizovaným analytickým prostředím může vhodně zvýšit atraktivitu vzniklého řešení.

Inspirace biologickými viry pro návrh a zlepšení malware analýzy se ukázala jako přínosná. Vedla k implementaci opatření proti teprve očekávaným evasivním technikám. V otázce trendu vývoje evasivních technik může vhodně napovědět právě biologie. V tamní časté ko-evoluci vztahů parazit–hostitel, resp. predátor–kořist se neustále vzájemně jedni přizpůsobují druhým. Převedením na vztah autor škodlivého kódu (malware) – analytik (analytické prostředí) bylo autorem práce předvedeno, jak evasivní malware při odhalování své analýzy používá základní detekční techniky (charakteristiky systému včetně názvů procesů) jakoby raného anti-antiviru a jak lze na obranu analytického prostředí naopak využít techniky převzaté od raných škodlivých kódů (změnou typických charakteristik). V očekávání, že budoucí malware bude své anti-antivirové kapacity dále vyvíjet a uplatní statické detekce pomocí hash otisků či signatur byly v práci již dopředu implementovány poly-/metamorfní mutace kódu několika analytických nástrojů

Návrhy pro budoucí zlepšení zahrnují zejména více automatizace analýzy škodlivého kódu s případným nasazením strojového učení. Jistě bude zajímavé sledovat trend kdy, zda a nakolik malware začne používat statickou detekci (obdobnou té používané antiviry) pomocí signatur na detekování analytických nástrojů a jak se osvědčí již nyní v práci implementované maskování poly-/metamorfními generátory. Nabízí se také doplnit analýzu vzorků analýzami v nevirtuálním prostředí (bare-metal). Zajímavým anti-evasivním opatřením by bylo ověřit proveditelnost, výhody a nevýhody analýz virtuálního prostředí založeného na speciální implementaci virtualizačního nástroje spočívající například v samo-modifikujícím se kódu virtualizačního nástroje.[209] Prioritou jsou zlepšení analýzy škodlivého kódu a způsoby, jak čelit novým hrozbám.

[209] Alexandre Dulaunoy (vývojář MISP, člen CERT@Luxembourg) se zmínil v rozhovoru o nepublikované implementaci QEMU pomocí samo-modifikujícího se kódu „self-modifying QEMU“ v prosinci 2017 a jako autora uvedl bezpečnostního analytika Paul Junga (Thanat0s – thanatos@trollprod.org. – https://github.com/Th4nat0s).

8. Přílohy

8.1 Slovníky pojmů a použitých zkratek

Akronymy používané vládními organizacemi USA
https://niccs.us-cert.gov/acronyms/

Anglický slovník běžných pojmů kybernetické bezpečnosti
https://niccs.us-cert.gov/glossary

Anglické slovníky IT a kybernetické bezpečnosti – ISACA
http://www.isaca.org/Knowledge-Center/Documents/Glossary/

Anglický slovník pojmů kybernetické bezpečnosti z oblasti Threat Intelligence
https://www.threatconnect.com/cyber-security-glossary/

Česko-anglický výkladový slovník kybernetické bezpečnosti – Třetí vydání[210]
https://nukib.cz/cs/informacni-servis/slovnik/[211]

Online vyhledávač akronymů a zkratek – Acronym Finder
https://www.acronymfinder.com/

8.2 Seznam odkazů na varianty vytvořeného prostředí

Odkazy na úložiště obsahují plnohodnotné prostředí a některé varianty virtuálních strojů. Odkazy na soubory s prostředím Javeman ke stažení, které jsou aktuálně umístěné na:

- Cloudovém úložišti MEGA.nz s end-to-end šifrováním (kapacita do 50 GB zdarma)
- GitLab repozitáři projektu (dočasně zveřejněném) – https://gitlab.com/jarkovar/javeman

Soubory .ova s virtuálními stroji a další prvky prostředí řídící analýzu lze stáhnout a nainstalovat skriptem `setup-javeman-on-kali.sh`[212] umístěném v GitLab repozitáři.

[210] JIRÁSEK, Petr, Luděk NOVÁK a Josef POŽÁR. *Výkladový slovník kybernetické bezpečnosti* [online].
[211] Obdobně na https://www.govcert.cz/.
[212] https://gitlab.com/jarkovar/javeman/blob/master/control/setup-initial-javeman-on-kali.sh

8.3 Seznam vložených tabulek

8.4 Seznam vložených obrázků

8.5 Seznam citované literatury

ADAMS, Matthew. Windows 10 hits 35% user base, Windows 7 takes the crown with 43%. In: *WindowsReport* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://windowsreport.com/windows-10-windows-7-user-base/

Alpine Linux 3.8.1 Released. *Alpine Linux* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://alpinelinux.org/posts/Alpine-3.8.1-released.html

Alpine Linux: About. *Alpine Linux* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://alpinelinux.org/about/

Alpine Linux: Downloads (Virtual). In: *Alpine Linux* [online]. b.r. [cit. 2018-11-05]. Dostupné z: http://dl-cdn.alpinelinux.org/alpine/v3.8/releases/x86_64/alpine-virt-3.8.1-x86_64.iso

ANDERSON, Hyrum. Bot vs. Bot: Evading Machine Learning Malware Detection: Endgame's presentation at Black Hat USA 2017. In: *Blackhat.com* [online]. 2017 [cit. 2018-04-10]. Dostupné z: https://www.blackhat.com/docs/us-17/thursday/us-17-Anderson-Bot-Vs-Bot-Evading-Machine-Learning-Malware-Detection.pdf

AV-TEST: Security Report 2015/2016. *AV-TEST* [online]. Magdeburg (Germany), 2016 [cit. 2018-04-15]. Dostupné z: https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2015-2016.pdf

AV-TEST: Security Report 2017/2016. *AV-TEST* [online]. Magdeburg (Germany), 2017 [cit. 2018-04-15]. Dostupné z: https://www.av-test.org/fileadmin/pdf/security_report/AV-TEST_Security_Report_2016-2017.pdf

BACURIO, Floser, Wayne LOW. Prevalent Threats Targeting Cuckoo Sandbox Detection and Our Mitigation. *Fortinet* [online]. 2018 [cit. 2018-11-26]. Dostupné z: https://www.fortinet.com/blog/threat-research/prevalent-threats-targeting-cuckoo-sandbox-detection-and-our-mitigation.html

BRANCO, Rodrigo a Gabriel BARBOSA. Scientific but Not Academical Overview of Malware Anti-Debugging, Anti-Disassembly and Anti- VM Technologies. In: *Black Hat* [online]. Las Vegas: Black Hat USA, 2012 [cit. 2018-03-28]. Dostupné z:

https://media.blackhat.com/bh-us-12/Briefings/Branco/BH_US_12_Branco_Scientific_Academic_WP.pdf

CESNET a CZ.NIC vybudují kybernetický bezpečnostní systém CTI. In: *CESNET* [online]. Praha, ČR: CESNET, z. s. p. o., 2017 [cit. 2018-11-30]. Dostupné z: https://www.cesnet.cz/sdruzeni/zpravy/tiskove-zpravy/cesnet-a-cz-nic-vybuduji-kyberneticky-bezpecnostni-system-cti/

CIMPANU, Catalin. WaterMiner Malware Author Can't Keep His Mouth Shut on Social Media. *BleepingComputer* [online]. 2017 [cit. 2018-08-24]. Dostupné z: https://www.bleepingcomputer.com/news/security/waterminer-malware-author-cant-keep-his-mouth-shut-on-social-media/

CISALPINO, Daniel a Eric BORTZ. Quora: Do viruses attack all cells but are only noticed in the cell type which has the dna affected or do they only attack specific cell types?. In: *Quora* [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://www.quora.com/Do-viruses-attack-all-cells-but-are-only-noticed-in-the-cell-type-which-has-the-dna-affected-or-do-they-only-attack-specific-cell-types

COHEN, Fred. Computer viruses. *Computers & Security* [online]. 1987, **6**(1), 22-35 [cit. 2018-03-18]. DOI: 10.1016/0167-4048(87)90122-2. ISSN 01674048. Dostupné z: http://linkinghub.elsevier.com/retrieve/pii/0167404887901222

CZUMAK, Mike. *PeCloak.py – An Experiment in AV Evasion* [online]. b.r. [cit. 2018-04-24]. Dostupné z: https://www.securitysift.com/pecloak-py-an-experiment-in-av-evasion/

Desktop Windows Version Market Share Worldwide (March 2017 - March 2018). *Statcounter: Globalstats* [online]. b.r. [cit. 2018-04-15]. Dostupné z: http://gs.statcounter.com/windows-version-market-share/desktop/worldwide

Exploring the opportunities and limitations of current Threat Intelligence Platforms — ENISA. In: *ENISA* [online]. Attiki, Heraklion (Greece), 2018 [cit. 2018-11-30]. Dostupné z: https://www.enisa.europa.eu/publications/exploring-the-opportunities-and-limitations-of-current-threat-intelligence-platforms

FLEGR, Jaroslav. *Evoluční biologie*. Praha: Academia, 2005. ISBN 80-200-1270-2.

FORTUNA, Andrea. How to extract a RAM dump from a running VirtualBox machine. In: *Personal Blog* [online]. 2017 [cit. 2018-12-01]. Dostupné z: https://www.andreafortuna.org/dfir/forensics/how-to-extract-a-ram-dump-from-a-running-virtualbox-machine/

GAJDOŠOVÁ, Markéta. Reverzní inženýrství malwaru pomůže v aktivní ochraně. In: *Computerworld: SecurityWorld* [online]. Praha: IDG, 2013 [cit. 2018-04-12]. Dostupné z: https://computerworld.cz/securityworld/reverzni-inzenyrstvi-malwaru-pomuze-v-aktivni-ochrane-49491

Gartner Forecasts Worldwide Security Spending Will Reach $96 Billion in 2018, Up 8 Percent from 2017. In: *Gartner*. Egham (UK), 2017. [cit. 2018-07-17]. Dostupné také z: https://www.gartner.com/newsroom/id/3836563

Gartner Security & Risk Management Summit 2015 [online]. [cit. 2018-07-12]. Dostupné z:

https://web.archive.org/web/20161028053430/https://www.gartner.com/binaries/content/assets/events/keywords/security/sec16i/sec15_gartner_predicts_1.pdf

GILLET, Stephen. Chronicle Blog: Give Good The Advantage. *Medium* [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://medium.com/chronicle-blog/give-good-the-advantage-75ab2c242e45

GRANOFF, Allan. a Robert WEBSTER. *Encyclopedia of virology*. 2nd ed. San Diego: Academic Press, 1999. ISBN 978-0-122270307.

IDA Pro – at the cornerstone of IT security. In: *Hex-Rays* [online]. Belgium: Hex-Rays SA, 2009 [cit. 2018-12-08]. Dostupné z: https://www.hex-rays.com/products/ida/ida-executive.pdf

JIRÁSEK, Petr, Luděk NOVÁK a Josef POŽÁR. *Výkladový slovník kybernetické bezpečnosti: Cyber security glossary* [online]. Třetí aktualizované vydání. Praha: Policejní akademie ČR v Praze, 2015 [cit. 2018-04-09]. ISBN 978-80-7251-436-6. Dostupné z: https://nukib.cz/download/aktuality/container-nodeid-665/slovnikkb-cz-en-1505.pdf

Joe Security - About Company [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://www.joesecurity.org/company-joe-security

KANKOWSKI, Peter. Performance measurements with RDTSC. In: *Strchr.com* [online]. 2008 [cit. 2018-12-07]. Dostupné z: https://www.strchr.com/performance_measurements_with_rdtsc

KASPERSKY. Kaspersky Lab. Duqu 2.0: Technical Details: Technical Details. In: *Securelist.com*. Kaspersky Lab, 2015 [cit. 2018-07-04]. Dostupné také z: https://web.archive.org/web/20171123190036/https://securelist.com/files/2015/06/The_Mystery_of_Duqu_2_0_a_sophisticated_cyberespionage_actor_returns.pdf

KASPERSKY, Eugene. Kaspersky Lab investigates hacker attack on its own network. *Kaspersky Lab* [online]. 2015 [cit. 2018-07-03]. Dostupné z: https://blog.kaspersky.com/kaspersky-statement-duqu-attack/

KOVÁŘ, Jaroslav. Virtuální prostředí pro behaviorální analýzu škodlivého kódu. Bakalářská práce (Bc.). Jihočeská univerzita v Českých Budějovicích. Přírodovědecká fakulta. 2013.

LARABEL, Michael. Debian Making Progress On UEFI SecureBoot Support In 2018. In: *Phoronix* [online]. USA: Phoronix Media, 2018 [cit. 2018-12-01]. Dostupné z: https://www.phoronix.com/scan.php?page=news_item&px=Debian-UEFI-SecureBoot-2018

Lastline Enterprise. Evasive Malware Defeats Advanced Detection Tools. *Lastline* [online]. 2017 [cit. 2018-04-15]. Dostupné z: https://www.infosecurityeurope.com/__novadocuments/357217?v=636295319723800000

LEE LI YING, By. Cybersecurity chief on attacks in Singapore in 2017. In: *CNA* [online]. Mediacorp, b.r. [cit. 2018-04-10]. Dostupné z: https://www.channelnewsasia.com/news/singapore/we-were-just-lucky-cybersecurity-chief-on-attacks-in-singapore-9504336

Linux Cuckoo. In: *Cuckoo Sandbox* [online]. 2017 [cit. 2018-11-28]. Dostupné z: https://web.archive.org/web/20170809220228/https://linux.huntingmalware.com/

LOMBARDI, Mike. Rolling Your Own Incident Response with PowerShell - SANS Blue Team Summit 2018. *SANS* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://www.sans.org/cyber-security-summit/archives/file/summit-archive-1524594407.pdf

LUSTIG, A a A LEVINE. One hundred years of virology. *Journal of Virology*. 1992, **66**(8), 4629-4631. ISSN 0022-538X. Dostupné také z: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC241285/

MAEC 5.0 Cuckoo Report Module. In: *GitHub* [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://github.com/MAECProject/cuckoo/blob/maec5.0-cuckoo2.0/cuckoo/reporting/maecreport.py

MAEC 5.0. *MAEC* [online]. MITRE Corporation, 2017 [cit. 2018-04-10]. Dostupné z: http://maecproject.github.io/releases/5.0/

MALAVER, Sharron. Mimicry: The Evolution of Deceptive Defenses and Attacks. *Minerva* [online]. 2018 [cit. 2018-08-24]. Dostupné z: https://blog.minerva-labs.com/mimicry-the-evolution-of-deceptive-defenses-and-attacks

Malware Information Sharing Platform - NATO. *NCI Agency (NATO Communications and Information Agency)* [online]. Brussels (Belgium), b.r. [cit. 2018-11-30]. Dostupné z: https://www.ncia.nato.int/Documents/Agency%20publications/Malware%20Information%20Sharing%20Platform%20(MISP).pdf#page=2

Malware Statistics & Trends Report | AV-TEST. *AV-TEST* [online]. Magdeburg, Germany, 2018 [cit. 2018-11-28]. Dostupné z: https://www.av-test.org/en/statistics/malware/

MANKY, Derek. Cybersecurity 2018: Year in Preview. In: *Fortinet* [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://www.fortinet.com/content/dam/fortinet/assets/analyst-reports/Derek-Manky-2018-Year-in-Preview.pdf

MARAK, Victor. *Command Line for Windows Malware and Forensics* [online]. In: . 2013 [cit. 2018-04-24]. Dostupné z: http://resources.infosecinstitute.com/commandline-malware-and-forensics/

MARAK, Victor. Leveraging the command line for windows: malware analysis and forensics. *InfoSec Institute* [online]. 2013 [cit. 2018-11-05]. Dostupné z: https://resources.infosecinstitute.com/command-line-for-windows-malware-analysis-forensics-part-i/

MAREŠ, Milan. *Základy teorie informace: zdroje informace a její měření*. 1. vyd. České Budějovice: Jihočeská univerzita, 2011. ISBN 978-80-7394-291-5.

MARTINEZ, Emiliano. Malware analysis sandbox aggregation: Welcome Tencent HABO!. In: *VirusTotal* [online]. 2017 [cit. 2018-11-28]. Dostupné z: https://blog.virustotal.com/2017/11/malware-analysis-sandbox-aggregation.html

Minerva Labs Research Report: 2017 Year in Review. *Minerva* [online]. 2017 [cit. 2018-04-14]. Dostupné z: https://l.minerva-labs.com/hubfs/Minerva 2017 Yearly Report_FINAL.pdf

Minerva Labs Research Team. WaterMiner – a New Evasive Crypto-Miner. In: *Minerva Labs: Blog* [online]. 2017 [cit. 2018-04-13]. Dostupné z: https://blog.minerva-labs.com/waterminer-a-new-evasive-crypto-miner

MUELLER, John Paul. *Příkazový řádek Windows pro Windows Vista, 2003, XP a 2000*. Vyd. 1. Přeložil Milan ZELENKA, přeložil Pavel PALONCÝ, přeložil Tomáš MOSLER. Brno: Computer Press, 2008. ISBN 978-80-251-1961-7.

NopSec State of Vulnerability Risk Management Report. *NopSec* [online]. 2015 [cit. 2018-04-23]. Dostupné z: http://info.nopsec.com/rs/736-UGK-525/images/NopSec_StateofVulnRisk_WhitePaper_2015.pdf

PAGANINI, Pierluigi. Duqu 2.0: The Most Sophisticated Malware Ever Seen. *InfoSec Institute* [online]. 2015 [cit. 2018-04-09]. Dostupné z: http://resources.infosecinstitute.com/duqu-2-0-the-most-sophisticated-malware-ever-seen/

PALKMETS, Lauri, Cosmin CIOBANU, Yonas LEGUESSE, Christos SIDIROPOULOS. ENISA – Building artifact handling and analysis environment. In: *ENISA* [online]. Heraklion (Greece), 2014 [cit. 2018-12-09]. Dostupné z: https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/documents/building-artifact-handling-and-analysis-environment-toolset

ResourceHacker: freeware resource compiler & decompiler for Windows applications. *Agnus Johnson* [online]. b.r. [cit. 2018-11-06]. Dostupné z: http://www.angusj.com/resourcehacker/

ROYAL, Paul. Entrapment: Tricking Malware with Transparent, Scalable Malware Analysis. Black Hat 2012. In: *Black Hat* [online]. 2012 [cit. 2018-12-08]. Dostupné z: http://media.blackhat.com/bh-eu-12/Royal/bh-eu-12-Royal-Entrapment-WP.pdf

SAJEEV, Nair. Live Response Using PowerShell: White Paper. *SANS* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://www.sans.org/reading-room/whitepapers/forensics/live-response-powershell-34302

Share of Steam users in March 2018, by operating system. *Statista* [online]. b.r. [cit. 2018-04-15]. Dostupné z: https://www.statista.com/statistics/265033/proportion-of-operating-systems-used-on-the-online-gaming-platform-steam/

SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis: the hands-on guide to dissecting malicious software* [online]. San Francisco: No Starch Press, 2012 [cit. 2018-03-18]. ISBN 978-1-59327-290-6.

SIKORSKI, Michael. a Andrew. HONIG. *Practical malware analysis: the hands-on guide to dissecting malicious software* [online]. San Francisco: No Starch Press, 2012 [cit. 2018-07-03]. ISBN 978-1-59327-290-6.

STEFNISSON, Siggi. Evasive Malware Now a Commodity. *SecurityWeek* [online]. Wired Business Media, 2018 [cit. 2018-09-25]. Dostupné z: https://www.securityweek.com/evasive-malware-now-commodity

STEMP, Greg, Dean TSALTAS, Bob WELLS a Ethan WILANSKY. WMI Scripting Primer: Part 1. *Microsoft Docs* [online]. 2006 [cit. 2018-12-08]. Dostupné z: https://docs.microsoft.com/en-us/previous-versions/windows/internet-explorer/ie-developer/scripting-articles/ms974579(v=msdn.10)

Study Design 101: The Himmelfarb Health Sciences Library [online]. 2011 [cit. 2018-08-26]. Dostupné z: https://himmelfarb.gwu.edu/tutorials/studydesign101/formulas.html

Symantec: Internet Security Threat Report (ISTR) 2018. In: *Symantec* [online]. 2018 [cit. 2018-04-15]. Dostupné z: https://www.symantec.com/content/dam/symantec/docs/reports/istr-23-2018-en.pdf

SZOR, Peter. *The art of computer virus research and defense*. Upper Saddle River, NJ: Addison-Wesley, 2005. ISBN 03-213-0454-3.

TIMMER, John. Researchers encode malware in DNA, compromise DNA sequencing software. In: *ArsTechnica.com* [online]. New York: Condé Nast Publications, 2017 [cit. 2018-03-28]. Dostupné z: https://search.proquest.com/docview/1928007073?accountid=9646

TOXEN, Bob. *Bezpečnost v Linuxu: prevence a odvracení napadení systému*. Brno: Computer Press, 2003. Security (CP Books). ISBN 80-722-6716-7.

Use 7-Zip to explore & extract EXE file contents. In: *Blog* [online]. b.r. [cit. 2018-11-05]. Dostupné z: https://qtechbabble.wordpress.com/2016/11/07/use-7-zip-to-explore-exe-file-contents/

Version 6 Release of the REMnux Linux. In: *Zeltser* [online]. 2016 [cit. 2018-04-15]. Dostupné z: https://zeltser.com/remnux-v6-release-for-malware-analysis/

Virus, malware, and threat descriptions - Windows Defender Security Intelligence. In: *Microsoft* [online]. 2018 [cit. 2018-12-06]. Dostupné z: https://www.microsoft.com/en-us/wdsi/threats?id=1

WICKLER, Wolfgang. Mimicry: Biology. *Encyclopedia Britannica* [online]. b.r. [cit. 2018-08-24]. Dostupné z: https://www.britannica.com/science/mimicry

WILLIAMS, Alice. Why your next malware infection might be AI-controlled. WILLIAMS, Alice. *VentureBeat* [online]. b.r. [cit. 2018-04-10]. Dostupné z: https://venturebeat.com/2017/05/11/why-your-next-malware-infection-might-be-ai-controlled/

Windows Sysinternals. *Microsoft* [online]. 2017 [cit. 2018-12-02]. Dostupné z: https://docs.microsoft.com/en-us/sysinternals/

WOJNER, Christian. ProcDOT. *ProcDOT - Visual Malware Analysis* [online]. b.r. [cit. 2018-09-27]. Dostupné z: http://procdot.com/index.htm

XU CHEN, , Jon ANDERSEN, Z. Morley MAO, Michael BAILEY a Jose NAZARIO. Towards an understanding of anti-virtualization and anti-debugging behavior in modern malware. *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)* [online]. IEEE, 2008, , 177-186 [cit. 2018-11-26]. DOI:

10.1109/DSN.2008.4630086. ISBN 978-1-4244-2397-2. Dostupné z: http://ieeexplore.ieee.org/document/4630086/

ZELTSER, Lenny. *Malware Vaccination for the Enterprise: Brought to You by Minerva* [online]. 2017 [cit. 2018-07-03]. Dostupné z: https://blog.minerva-labs.com/malware-vaccination-for-the-enterprise-brought-to-you-by-minerva

A. Sbírka na testování anti-evasion úrovně virtuálních prostředí

Název souboru	**Hash** souboru (md5, sha256) a reporty (srovnání s HybridAnalysis)	**Přidáno** do sbírky
al-khaser_x86.exe	**MD5:** 717f0ef3b7bb89027b149da1780fde5c **SHA256:** 6b6de107433cf579748c3fe669d79d3415548cf55b494824bb7eae3a846bde15 **HA Report** (*2018-12-07*): https://www.hybrid-analysis.com/sample/6b6de107433cf579748c3fe669d79d3415548cf55b494824bb7eae3a846bde15 **Zdroj:** https://github.com/LordNoteworthy/al-khaser/blob/master/al-khaser_ x86.exe	2018-08-12
al-khaser_x64.exe	**MD5:** 516d30e45b631f2a06cd31a8fc0ccefd **SHA256:** 33a7ea4d9ed0bf7ca2022d2fa6f667acf1bd0096ad61f46c7fa6288deace313e **HA Report** (*2018-08-24*): https://www.hybrid-analysis.com/sample/33a7ea4d9ed0bf7ca2022d2fa6f667acf1bd0096ad61f46c7fa6288deace313e/5b606f6f7ca3e16f446e6115 **Zdroj:** https://github.com/LordNoteworthy/al-khaser/blob/master/al-khaser_x64.exe	2018-08-12
InviZzzibleSandboxEvasion.exe	**MD5:** 26461532baba77c06ea3b07ace8c5fc3 **SHA256:** 93674c7982014a61c8d8e22df7f30c497db4e761b43368fdb331657d0cc41da3 93674c7982014a61c8d8e22df7f30c497db4e761b43368fdb331657d0cc41da3 **HA Report** (*2018-05-12*): https://www.hybrid-analysis.com/sample/93674c7982014a61c8d8e22df7f30c497db4e761b43368fdb331657d0cc41da3?environmentId=100 **Zdroj:** https://github.com/CheckPointSW/InviZzzible (nutné zkompilovat)	2018-04-15
pafish.exe	**MD5:** 9159edb64c4a21d8888d088bf2db23f3 **SHA256:** 2180f4a13add5e346e8cf6994876a9d2f5eac3fcb695db8569537010d24cd6d5 **HA Report** (2018-06-02): https://www.hybrid-analysis.com/sample/2180f4a13add5e346e8cf6994876a9d2f5eac3fcb695db8569537010d24cd6d5/57c76240aac2eddc62ac912d **Zdroj:** https://github.com/a0rtega/pafish/blob/master/pafish.exe	2018-05-27
SEMS.exe	**MD5:** b16eefa3049bb3cb0fcdd35bbdac5439 **SHA256:** 59bb9d0838882ec12d28eae3ef9b02ad58cd83ddffeecec4b938f7a58bed50ee **HA Report** (*2018-03-23*): https://www.hybrid-analysis.com/sample/59bb9d0838882ec12d28eae3ef9b02ad58cd83ddffeecec4b938f7a58bed50ee?environmentId=120 Zdroj: https://github.com/AlicanAkyol/sems/blob/master/Debug/sems.exe	2018-03-03