Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

Virtuální prostředí pro behaviorální analýzu škodlivého kódu

Bakalářská práce

Jaroslav Kovář

Školitel: Ing. Petr Břehovský

České Budějovice 2013

Bibliografické údaje

Kovář J., 2013: Virtuální prostředí pro behaviorální analýzu škodlivého kódu.

[Virtual environment for behavioral analysis of malware. Bc. Thesis, in Czech.] 45 p., Faculty of Science, The University of South Bohemia, České Budějovice, Czech Republic.

Annotation:

The goal of this bachelor thesis was to propose and implement a virtual environment for behavioral analysis of malware. Specifically with simulated network services. Subsequently was this environment applied to the analysis of several selected malware samples.

Anotace:

Cílem této bakalářské práce bylo navrhnout a implementovat virtuální prostředí pro behaviorální analýzu škodlivého kódu. Specificky se simulovanými síťovými službami. Následně bylo prostředí aplikováno na analýzu několika vybraných vzorků škodlivého kódu.

Prohlašuji, že svoji bakalářskou 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 bakalářské 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 26. dubna 2013

_____

Jaroslav Kovář

Poděkování

Děkuji svému školiteli Ing. Petru Břehovskému za odborné vedení práce, cenné rady a připomínky. Dále děkuji rodičům za podporu.

Obsah

Seznam tabulek

Seznam obrázků

1. Úvod

V dnešním světě počítačů, především mám na mysli prostředí Internetu, jsme doslova obklopeni škodlivým kódem, ač to nemusí být na první pohled zřejmé. Pojem škodlivý kód zahrnuje viry, trojské koně, červy, boty a libovolné další digitální škůdce. Souběžně se zrychlujícím vývojem celého odvětví informatiky se zdokonalují i škodlivé kódy. Uvážíme-li pak jejich zničující účinek, stává se detekce, analýza a zneškodnění škodlivého kódu důležitým až rozhodujícím úkolem informatiky.

Jednou z forem obrany je právě virtuální prostředí, v kterém každý pokročilejší uživatel může sám analyzovat podezřelé soubory na projevy škodlivého kódu.

První část práce je teoretická, obsahuje obecný popis škodlivého kódu a přehled nástrojů používaných k jeho analýze. Další části popisují mé hlavní přínosy k této problematice. Je zde podrobně popsán návrh virtuálního prostředí pro analýzu škodlivého kódu, způsob implementace navrženého prostředí. Následně jsem ve vytvořeném prostředí spustil a otestoval několik vzorků škodlivého kódu.

Kapitola 1 obsahuje úvod a cíle práce. Kapitola 2 popisuje škodlivý kód, jeho projevy a stávající postupy jeho analýzy. V kapitole 3 je uveden přehled virtualizačních a analytických nástrojů použitelných pro účely této práce.

V kapitole 4 se zabývám návrhem samotného virtuálního prostředí pro analýzu škodlivého kódu. Tato kapitola obsahuje výběr vhodného operačního systému k infikování, návrh virtuálního prostředí pro analýzu škodlivého kódu, návrh simulace síťových služeb a návrh postupu analýzy škodlivého kódu. V kapitole 5 popisuji, jak jsem postupoval při instalaci virtuálních stojů, jak jsem vytvořil simulaci síťových služeb a jakými analytickými nástroji jsem vybavil implementované prostředí.

V kapitole 6 jsem navrhl postup pro testování škodlivého kódu a popsal několik experimentů, kdy jsem pomocí vytvořeného prostředí otestoval vzorky škodlivého kódu. Kapitola 7 obsahuje návrhy budoucích řešení a v kapitole 8 je závěr práce.

1.1. Cíle

- Návrh a implementace virtuálního prostředí pro behaviorální analýzu škodlivého kódu
- Praktické vyzkoušení a aplikace vytvořeného virtuálního prostředí pro behaviorální analýzu škodlivého kódu
- Analýza výsledků práce

2. Škodlivý kód a jeho analýza

Nejprve uvádím přehled základních pojmů a zkratek dále používaných v práci:

Analytik

Tímto pojmem je označován bezpečnostní odborník analyzující škodlivý kód. V případě této práce též libovolný jiný uživatel používající realizované prostředí.

Klient / Oběť

V práci je pojmy označován virtuální stroj určený k inifkování škodlivým kódem.

Virtualizace

Postup, který umožňuje k dostupným zdrojům přistupovat jiným způsobem, než jak fyzicky existují. Software běžící ve virtualizovaném prostředí je přesvědčen, že běží na skutečném hardware, ovšem fyzický hardware nemůže řídit. [1 Hoopes, 2009]

Virtualizační nástroj

Program řídící přístup virtuálních strojů k hardwarovým prostředkům fyzického stroje.

Virtuální stroj – Virtual machine (VM)

Virtuální stroj představuje v této práci operační systém běžící ve virtuálním prostředí.

Hostující operační systém – Host OS

Operační systém fyzického počítače, na němž je spouštěn virtualizační nástroj.

Hostovaný operační systém - Guest OS

Operační systém virtuálního stroje běžícího v rámci virtualizačního nástroje.

Snímek virtuálního stroje (Snapshot)

Stav virtuálního stroje v libovolném okamžiku, který byl uložený a je možno ho následně obnovit.

Monitor virtuálních strojů - Virtual machine monitor (VMM)

Při virtualizaci vsuneme přímo pod kód systému další úroveň, programovou vrstvu zvanou Monitor virtuálních strojů. Ten se stará o přidělování zdrojů a provedení privilegovaného kódu systému, aplikační kód pak běží na nižší úrovni privilegovanosti a je vykonáván nativně.

Introspekce virtuálních strojů - Virtual machine introspection (VMI)

Technika umožňující z vně virtuálního prostředí analyzovat procesy běžící uvnitř tohoto prostředí.

2.1. Škodlivý kód

Škodlivý kód je souhrnný pojem zahrnující viry, trojské koně, červy, boty a libovolné další digitální škůdce, zpravidla záměrně vytvořené. Jedná se o spustitelný program schopný připojovat se k jiným programům, vykonávat škodlivou aktivitu a šířit se bez vědomí uživatele. Obecně obsahuje škodlivý kód spouštěcí, vlastní funkční a reprodukční část. Systém uživatele napadá z infikovaného záznamového média nebo síťového spojení. Škodlivý kód je tvořen na nejnižší úrovni ve strojovém kódu až po vysokoúrovňové programovací jazyky. Motivací většiny tvůrců škodlivého kódu je dnes finanční zisk, ovšem nebylo tomu vždy tak.

Historie škodlivého kódu sahá na přelom 60. a 70. let minulého století, kdy se na sálových počítačích objevují první sebereplikující se programy obsazující systémové prostředky a snižující výkonnost systému. Pojem počítačový virus definoval v roce 1983 Frederick Cohen [19] jako „počítačový program, který může infikovat jiný počítačový program takovým způsobem, že do něj nakopíruje své tělo, čímž se infikovaný program stává prostředkem pro další aktivaci viru."

Na příkladech sedmi škodlivých kódů shrnuji jejich vývoj.

Tabulka 1: Přehled historie škodlivého kódu

1971 Creeper	Jeden z **prvních** sebereplikujících se virů.
1986 Brain	Často pokládán za první počítačový vir pro MS-DOS (IBM PC virus). Napadal **boot** sektory disket a tím znemožnil jejich rozpoznání počítačem.
1987 Cascade	První vir se za**šifrovaným** kódem.
2000 Iloveyou	Vir šíříce se **poštou**, vracel jméno a heslo z napadeného počítače. Infikoval více než milion počítačů.
2004 Sasser	Červ **skenující zranitelné IP adresy**, na kterých ukládal své kopie.
2008 Conficker	Šířil se na Windows (servery), používal šifrování a **zneviditelnění** „stealth code".
2011 Zeus	Sada pro **generování škodlivého kódu** (botů spojujících se do botnetů) prodávaná v rámci stínové ekonomiky kyberzločinu sloužící k zisku například z obchodu s ukradenými daty.

Jak se šíří?

Škodlivý kód potřebuje ke svému šíření vhodné prostředí, počítač s vhodným operačním systémem a objekty, které dokáže napadnout. Nejčastější cesty šíření škodlivého kódu jsou přes Internet, kde škodlivý kód často využívá chyb v prohlížeči.

Škodlivý kód se nejčastěji šíří síťovými službami, elektronickou poštou nebo infikovanými USB zařízeními. Zpravidla se těmito kanály šíří infikovanými programy, skripty nebo přes dokumenty obsahující makra.

Při spuštění infikovaného programu se škodlivý kód nahraje do operační paměti počítače, odkud vykonává svůj zákeřný kód. V případě přenosu přes paměťová média je cílem viru boot sektor nebo partition tabulka disku. Z napadeného počítače se může škodlivý kód šířit dál.

Stínová ekonomika obchodující s ukradenými daty (osobními údaji) stále roste. Například skupina bezpečnostních analytiků z Vídeňské technické univerzity v období od dubna do října 2008 zaznamenala 173,000 obětí kybernetického útoku s tím, že bylo odcizeno 33 GB dat.

2.2. Statická analýza

Statická analýza škodlivého kódu poskytuje signatury pro jeho detekci [4]. Kód je analyzován velmi detailně na instrukční úrovni a tím o něm získají analytici přesné znalosti.

Tento typ analýzy je ovšem velmi pracný. Obvykle se provádí manuální analýzou zdrojového kódu. Analytici použijí metod reverzního inženýrství a ručního dissasemblování k získání charakteristických bytových sekvencí škodlivého kódu. Vytvořené signatury škodlivého kódu jsou následně zahrnuty do databází antivirových programů, které během skenování binárních dat hledají bytové sekvence charakteristické pro již identifikovaný škodlivý kód. Autoři škodlivého kódu se rychle přizpůsobili a začali vytvářet polymorfické kódy, které mění své charakteristiky a tudíž jsou antivirovými programy obtížně detekovatelné [4].

Vzhledem ke stále většímu objemu škodlivého kódu zasílanému antivirovým společnostem každý den je pro časovou náročnost a komplexnost úlohy stále obtížnější analyzovat škodlivý kód k získání charakteristických sekvencí v přiměřené době.

Tradiční metody statické analýzy a detekce škodlivého kódu přestávají být schopné držet krok se zrychlující se evolucí škodlivého kódu, která v poslední době přinesla například techniky maskování, zmíněný polymorfismus, packování a šifrování [4].

Odpovědí na tento problém je přechod k dynamické, případně následné behaviorální analýze, neboť vedle statické analýzy škodlivého kódu je možno zkoumat též jeho projevy. Vlastní chování určitého kódu lze definovat jako pozorovatelné projevy způsobené tímto kódem na okolní prostředí [22 Wombat].

2.3. Dynamická a behaviorální analýza

Proces dynamické analýzy škodlivého kódu poskytuje náhled na činnost prováděnou daným škodlivým kódem. K dosažení cílů dynamické analýzy škodlivého kódu jsou potřeba nástroje, které poskytují přehled o jeho chování. Takové nástroje by měly být schopny zaznamenat závažné bezpečnostní chování škodlivého kódu.

Kombinováním řady známých technik je možné chování daného škodlivého kódu sledovat. Škodlivé aktivity zkoumaného kódu se mohou skládat z otevření síťového spojení a přenosu paketů, vytvoření, změny nebo smazání souboru, modifikace registrů, přístupu k virtuální paměti, vytvoření procesů, a podobně.

Při dynamické analýze je škodlivý kód spuštěn v simulovaném prostředí a následně je analyzováno jeho chování. Jedna metoda spočívá v analýze rozdílů mezi dvěma stavy systému (jeden stav před spuštěním, druhý stav po spuštění škodlivého kódu), zatímco druhá metoda monitoruje akce provedené škodlivým kódem v reálném čase, přičemž obě mají své výhody a nevýhody [42 Weber 2010].

Analýza škodlivého kódu porovnáním stavu systému před a po provedení škodlivého kódu je zdánlivě jednodušší k realizaci a tudíž méně náchylná k implementačním chybám, ovšem výsledky takové analýzy jsou jen hrubé. Například sotva zjistí, zda v čase mezi dvěma stavy systému nebyl vytvořen a zase smazán soubor škodlivého kódu [42 Weber 2010].

Druhý postup dynamické analýzy škodlivého kódu má zase omezení v tom, že ač jsou monitorováním sledovány projevy škodlivého kódu zpravidla detailněji, zaměřuje se jen na určité oblasti systému. Též neposkytuje informace o tom, jak je škodlivý kód programován a zkoumá pouze prováděné akce škodlivého kódu v prostředí, v němž je spouštěn. Obecně platí, že informace získané tímto způsoben stačí k ohodnocení, jak je škodlivý kód nebezpečný a základnímu přehledu jeho funkčnosti.

Behaviorální analýza je nadstavbou nad dynamickou analýzou. Po získání dat z dynamické analýzy může behaviorální analýza určit pomocí metod strojového učení, například klasifikací, kategorii či míru nebezpečnosti vzorku škodlivého kódu. V práci jsem se zaměřil na tvorbu prostředí pro dynamickou analýzu, která je nutným předpokladem prostředí pro behaviorální analýzu škodlivého kódu.

2.4. Existující řešení dynamické analýzy

Existující prostředky pro řešení dynamické analýzy jsou přehledně zobrazeny v tabulce 2 převzaté z článku [4 Egele, 2010]. Z této tabulky plyne, že problematice simulovaných síťových služeb je věnována jen malá pozornost, přestože právě tato metoda umožňuje analyzovat neznámý škodlivý kód aniž by se během analýzy „ohlásil domů“ a tím například kontaktoval systém útočníka. V existujících řešeních je převážně škodlivému kódu jen filtrován přístup na Internet. Naproti tomu simulované síťové služby žádné informace na internetovou síť nedostanou.

V práci se proto zaměřím na tvorbu virtualizovaného prostředí obsahujícího simulované síťové služby, které umožní sledovat chování škodlivého kódu v izolovaném prostředí.

Tabulka 2: Porovnání existujících řešení dynamické analýzy

	Anubis (5.1)	Multipath exp. (5.2)	Clustering (6.1)	CWSandbox (5.3)	Learning & Clas. (6.3)	Norman Sandbox (5.4)	Joebox (5.5)	Ether (5.6)	WiLDCat (5.7)	Hookfinder (5.8)	Justin (5.9.1)	Renovo (5.9.2)	PolyUnpack (5.9.3)	OmniUnpack (5.9.4)	Behav. Spyw. (5.10.1)	TQana (5.10.2)	Panorama (5.10.3)	Behav. Clas. (6.2)
Analysis implementation																		
User-mode component	○	○	○	●	●	○	●	○	●	○	●	○	●	●	●	○	○	●
Kernel-mode component	○	○	○	●	●	○	●	○	●	●	●	●	○	●	●	○	●	●
Virtual machine monitor	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○
Full system emulation	●	●	●	○	○	○	○	○	○	●	○	●	○	○	○	●	●	○
Full system simulation	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○
Analysis targets																		
Single process	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	●	○	●
Spawned processes	●	○	●	●	●	○	●	○	○	○	○	○	○	○	○	○	○	○
All processes on a system	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	●	●	●
Complete operating system	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	●	○
Analysis support for																		
API calls	●	●	●	●	●	●	●	○	○	●	○	○	○	○	○	●	●	●
System calls	●	●	●	●	●	●	●	●	○	●	○	○	○	●	●	●	●	●
Function parameters	●	●	●	●	●	●	●	○	○	●	○	○	○	○	○	●	●	●
File system operations	●	●	●	●	●	●	●	○	○	●	○	○	○	●	●	●	●	●
Process/thread creation	●	●	●	●	●	●	●	○	○	●	○	○	○	○	●	●	●	●
Registry operations	●	●	●	●	●	●	●	○	○	●	○	○	○	●	●	●	●	●
COM components	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○	●	○	○
Detecting dyn. generated code	○	○	○	○	○	○	○	○	○	○	●	●	●	●	○	○	○	○
Restoring packed binaries	○	○	○	○	○	○	○	○	○	○	●	●	●	●	○	○	○	○
W ⊕ X page protection	○	○	○	○	○	○	○	○	○	○	●	●	○	●	○	○	○	○
Multiple layers of packers	○	○	○	○	○	○	○	○	○	○	○	●	○	○	○	○	○	○
Signature matching after unpacking	○	○	○	○	○	○	○	○	○	○	●	○	○	●	○	○	○	○
Instruction trace	○	●	●	○	○	○	○	●	●	●	○	●	●	○	○	○	●	○
Information flow tracking	○	●	●	○	○	○	○	○	○	●	○	○	○	○	○	●	●	○
Multiple path exploration	○	●	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
ASEP monitoring	○	○	○	○	○	●	○	○	○	●	○	○	○	○	○	○	○	○
Networking support																		
Simulated network services	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○	○	○	○
Internet access (filtered)	●	●	●	●	●	●	●	●	●	●	●	●	●	○	●	●	●	●
Performs cloaking																		
Shadow EFLAGS register	○	○	○	○	○	○	○	●	●	○	○	○	○	○	○	○	○	○
Process hiding	○	○	○	●	●	○	●	○	●	○	○	○	○	○	○	○	○	○
Dynamic module hiding	○	○	○	●	●	○	●	○	○	○	○	○	○	○	○	○	○	○
Disguise modified memory pages	○	○	○	○	○	○	●	○	○	○	●	○	○	○	○	○	○	○
Disguise elapsed time	○	○	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	○
Clustering Support	○	○	●	○	●	○	○	○	○	○	○	○	○	○	○	○	○	●
Simulated User Interaction	○	○	○	○	○	○	●	○	○	○	○	○	○	○	○	●	●	○

Table I. Comparison of General Malware Analysis Tools

ACM Computing Surveys, Vol. V, No. N, 20YY.

3. Přehled použitelných nástrojů

3.1. Přehled virtualizačních nástrojů

Oracle VM VirtualBox

Jedná se o multiplatformní virtualizační nástroj umožňující virtualizaci širokého spektra operačních systémů, podporovaný pro hostitelské operační systémy Windows, Linux, Macintosh a Solaris. Mimo jiné umožňuje vytváření snímků virtuálního stroje, klonování snímků virtuálního stroje, ovládání v grafickém režimu i přes příkazovou řádku, připojení USB zařízení a podporuje hardwarovou virtualizaci instrukčních sad[1].

Jedná se o profesionální řešení, které je dostupné pod licencí GNU General Public Licence (GPL) verze 2. Tento produkt byl původně vytvořen německou firmou Innotek GmbH, kterou v roce 2008 koupila firma Sun Microsystems a od roku 2009 je vyvíjen firmou Oracle.

VMWare Workstation

Jedná se o multiplatformní virtualizační nástroj umožňující virtualizaci širokého spektra operačních systémů, podporovaný pro hostitelské operační systémy Windows a Linux. Mimo jiné umožňuje vytváření snímků virtuálního stroje, 3D akceleraci, připojení USB zařízení a podporuje hardwarovou virtualizaci instrukčních sad.

Jedná s o komerční řešení firmy VMWare, Inc., která umožňuje zdarma vyzkoušet produkt v 30 denní zkušební době.

VMWare Player

Tento software umožňuje spuštění virtuálních stojů vytvořených jinými nástroji firmy VMWare, Inc., neumožňuje však už jejich vytváření ani další upravování. Možnost získat tento software zdarma je vyvážena jeho omezenými schopnostmi.

Microsoft Windows Virtual PC

Virtualizační program určený pro virtualizaci operačního systému Windows. Nejnovější verze programu, Microsoft Windows Virtual PC 7, umožňuje virtualizovat pouze operační systém Windows XP SP3 Professional, navíc podporuje jen hostitelské operační systémy Microsoft Windows 7. Pro podporu virtualizace starších variant operačního systému Windows je třeba použít starší verzi programu Microsoft Windows Virtual PC.

1# Zavedení instrukčních sad Intel VT-x a AMD-x v roce 2005 umožnilo procesorům architektury x86 splnit podmínky virtualizace dle Popeka a Goldberga.

3.2. Přehled analytických nástrojů

ESET SysInspector

Tento program je použitelný k všestrannému monitoringu systému. Program prohledává počítač a shromažďuje informace o nainstalovaných ovladačích, aplikacích, síťových připojeních a položkách v registrech. Tím lze zjistit náznaky podezřelého chování systému z důvodu nekompatibility softwaru/hardwaru nebo právě napadení škodlivým kódem. Program dále informuje o běžících procesech a službách, přítomných podezřelých souborech, kompatibilitě hardwaru i softwaru, ovladačích, poškozených registrech nebo podezřelých síťových připojeních.

INetSim

INetSim představuje sadu nástrojů simulujících běžné internetové služby v umělém (například virtuálním) prostředí. Simulací nahrazuje připojení k Internetu. Zpravidla se používá při analýze síťového chování neznámých vzorků škodlivého kódu.

Wireshark

Velmi populární nástroj pro sledování a analyzování síťové komunikace. Umožňuje zachytávat komunikaci procházející skrze síťová rozhraní počítače (Ethernet, IEEE 802.11, PPP, Bluetooth, Token Ring, a další). Mezi jeho důležité vlastnosti patří velké množství podporovaných protokolů, pokročilé filtrování zachycených dat, podpora dešifrování protokolů (IPSec, WPA, WEP, aj.), analyzování VoIP komunikací či možnosti exportu dat (plaintext, XML, PostScript, CSV a řada dalších). [6 Dostálek, 2005]

ProcessMonitor

Tento program monitoruje procesy systému v reálném čase. Vypisuje operace se soubory (zápis, čtení, přesun) a operace prováděné se systémovým registrem (čtení, úprava, zápis). Umožňuje pokročilé filtrování zajímavých informací a následný export výsledků do souboru.

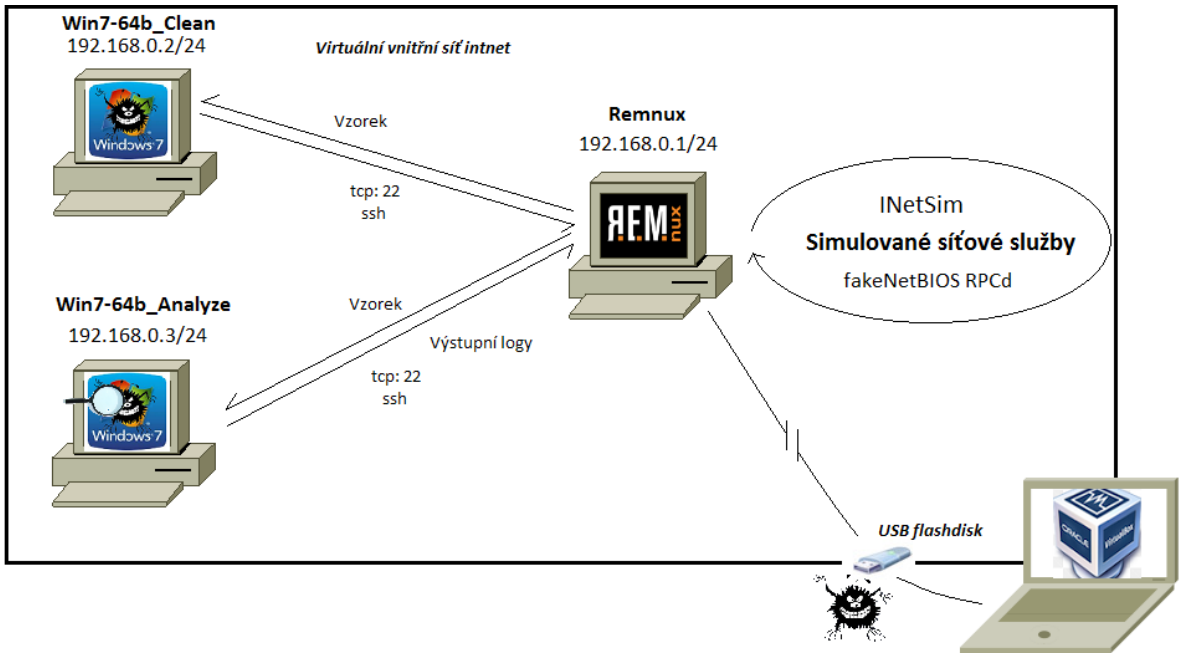
CaptureBAT

Je nástrojem pro analýzu programů pro operační systémy Windows (Win32). CaptureBAT umí monitorovat stav systému během spuštění programu a zpracování souborů, čímž umožňuje analytikovi zkoumat projevy programu bez potřeby znát jeho zdrojový kód. Nástroj monitoruje změny stavu na úrovni jádra a může být snadno použit napříč různými verzemi a konfiguracemi Windows. CaptureBAT je vyvíjen Christianem Seifertem v rámci The Honeynet Projectu [17], [18].

4. Návrh řešení

4.1. Obecný návrh prostředí

Hlavními požadavky na prostředí jsou izolace od okolí, spolehlivost a jednoduchost. Na úvod přikládám schéma návrhu.



Obrázek 1: Návrh virtuálních strojů v prostředí

4.2. Výběr operačního systému vhodného k infikování

Závislost mezi kořistí a predátory vyskytující se v přírodě, platí obdobně také ve světě počítačů. Čím více přibývá uživatelů určitého softwaru, tím více s určitým zpožděním přibývá tvůrců škodlivého kódu napadajícího tento software. Proto jsem se rozhodl zjistit jaký operační systém bude představovat nejlepší oběť, aby se analyzovaný škodlivý kód nejspíše projevil ve své pravé podstatě. Nejlepší obětí rozumím systém, který má z hledisek rozšířenosti a infikovatelnosti nejvyšší šanci, že bude napaden škodlivým kódem. Rozhodl jsem se ověřit vhodnost systému Windows 7 64-bit.

Existují viry pro 64-bitové Windows 7 a poběží na nich Win/32 viry?

Ano, například Zeus [28]. Škodlivý kód staršího data samozřejmě bude mít na 64-bitovém operačním systému potíže, avšak takových ubývá, zatímco rychle

přibývá těch, co s 64-bitovým prostředím počítají. Buď běží v režimu kompatibility a zaměřují se jen na 32-bitové procesy nebo jsou přímo napsány pro 64-bitové prostředí. Pro škodlivý kód navíc není problém nejdříve stáhnout bootstrapper, provést kontrolu zda je verze 32-bitová nebo 64-bitová a podle toho stáhnout svůj další kód.

Jsou 64-bitové Windows verze bezpečnější než 32-bitové?

Microsoft tvrdí, že ano, nicméně Alfred Huger, bývalý zaměstnanec společnosti Symantec [24 Keizer, 2009], přesvědčivě tvrzení vyvrací, když vysvětluje, že Microsoft při statistice vycházel z dat nasbíraných z nástroje *Microsoft malware removal tool*, který rozhodně nezpracovává všechny hrozby. Zaměřuje se na hlavní hrozby a tím pádem je statistika zkreslená. Trendem je růst hrozeb pro Windows 7 64-bit [29 Keiz].

Infikovanost Windows XP a Windows 7

Podle následující tabulky, převzaté ze zprávy Microsoft Security Intelligence Report Volume 13 [7, strana 57], je v průměru infikováno 9,5 z 1000 systémů s Windows XP, zatímco u hodnocených verzí Windows 7 je míra infikovanosti nižší, v rozmezí od 5,3 do 3,1.

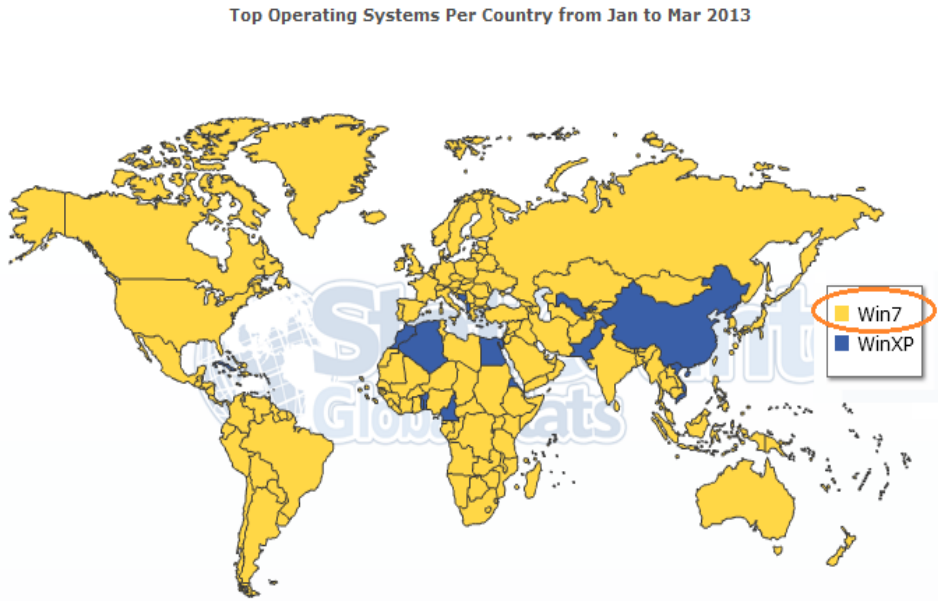
Tabulka 3: Míra infikovanosti operačních systémů Windows podle [7]

Verze operačního systému	Počet infikovaných systémů z 1000 systémů
Windows XP	9,5
Win7 32bit RTM	5,3
Win7 64bit RTM	4,3
Win7 32bit SP1	4,9
Win7 64bit	3,1

Zatím jsou více infikovány 32-bitové systémy Windows 7, nicméně podle Microsoftu se jedná o dočasný trend. Ve zprávě [7] se předpokládá, že tento trend bude postupně nahrazen vyšším počtem infikovaných 64-bitových verzí Windows 7. Předpokládá se, že 64-bitové systémy byly donedávna sférou odborníků, přičemž teprve v poslední době začínají být mainstreamovou záležitostí. Odborník byl a bude schopen lépe zabezpečit svůj operační systém než běžný uživatel, což dočasně snižuje podíl nakažených 64-bitových Windows 7.

Rozšířenost Windows XP a Windows 7

Mapa rozšíření operačních systémů Windows XP a Windows 7, kterou jsem převzal z [26], ukazuje větší rozšířenost Windows 7. Podle GlobalStats pouze v Číně významně převažuje Windows XP nad Windows 7, překvapivě až dvojnásobně. Tato statistika byla vytvořena podle verze operačního systému návštěvníků zhruba 3 miliónů sledovaných webů [26].



Obrázek 2: Mapa rozšíření operačních systémů[26]

V následujících tabulkách je uvedeno rozšíření Windows 7 a Windows XP ve světě podle různých průzkumů. Vyšší procentuální rozšíření vychází jednoznačně pro Windows 7.

Tabulka 4: Průzkum W3Schools [25] podílu Windows XP a Windows 7 ve světě

Období	Win7	WinXP
Říjen 2011	44.7%	33.4%
Říjen 2012	56.8%	22.1%

Tabulka 5: Průzkum GlobalStats [26] podílu Windows XP a Windows 7 ve světě

Listopad 2012	Win7	WinXP
Svět (průměr)	53.1%	26.1%
USA	50%	15%

Listopad 2012	Win7	WinXP
ČR	54%	30%
Čína	32%	63%

Tabulka 6: Souhrný přehled podílu Windows XP a Windows 7 ve světě

Říjen 2012	Win7	WinXP
W3Schools [25]	56.8%	22.1%
GlobalStats [26]	52.9%	27.0%
NetMarket [27]	40.4%	36.0%
Průměr	**50%**	**28%**

Jaká verze Windows bude nejlepší obětí škodlivého kódu z hlediska rozšíření a infikovanosti?

Mohu provést hrubé odhady šance na infekci pro jednotlivé verze operačního systému, kde šanci na infekci operačního systému spočítám jako součin procentuálního rozšíření operačního systému a jeho míry infikovanosti.

*procentuální rozšíření * míra infikovanosti = šance na infekci*

Tabulka 7: Odhad šance na infekci pro Windows XP a Windows 7

Systém	Procentuální rozšíření	Míra infikovanosti	Šance na infekci
WinXP	0,28	9,5	2,66
Win7 32b RTM	0,5	5,3	2,65
Win7 všechny verze (nevážený) průměr	0,5	4,4	2,25

Podíl zastoupení Win7 ve světě roste (trend z výše uvedených statistik) a též tvůrci škodlivého kódu se postupně více zaměřují na Windows 7 [29]. Výrazný pokles rozšířenosti Windows XP lze očekávat s blížícím se ukončením podpory tohoto systému Microsoftem od dubna 2014 [40] a můžu předpokládat, že část bývalých uživatelů Windows XP bude migrovat na Windows 7.

Co se týče podílu 32-bitových a 64-bitových verzí Windows 7, není o tom obecně mnoho dostupných dat, nicméně už v roce 2010 byla zhruba polovina všech verzí Windows 7 ve verzi 64-bit [41] a její podíl roste. Ve prospěch Windows 7 64-bit hovoří průzkum Garnteru [30] i počítačoví hráči, kteří mu jasně dávají přednost podle průzkumu Steam [31]. Lze předpokládat, že potenciál mít více než 4GB operační paměti bude významnou motivací k přechodu na 64-bitovou verzi operačního systému.

Výhledově tudíž pokládám za nejlepší oběť z hlediska rozšířenosti a infikovanosti systém Windows 7 64-bit a rozhodl jsem se tento systém použít v navrhovaném virtuálním prostředí.

4.3. Návrh virtuálního prostředí pro analýzu škodlivého kódu

Výběr virtualizačního nástroje

Vybíral jsem mezi virtualizačními nástroji *Oracle VM VirtualBox* a *VMWare Workstation*. Oba tyto nástroje nabízejí obdobné možnosti [32].

Vybral jsem si *Oracle VM VirtualBox 4.2.x* (dále jen *VirtualBox*) vzhledem k tomu, že je navíc dostupný pod licencí GNU General Public Licence. V budoucnu by se mi mohl hodit přístup ke zdrojovým kódům tohoto virtualizačního prostředí. Využiji možnost nastavit jednotlivým virtuálním strojům míru vytíženosti CPU (a dalšího hardware), čímž umožním spustit prostředí i na méně výkonných počítačích. S ohledem na případnou automatizaci je výhodou možnost propojení *VirtualBoxu* na programovací prostředí Python [2 Hoopes, 2007].

Návrh virtuálních strojů

Virtualizací pomocí nástroje VirtualBox jsem vytvořil izolované prostředí se třemi virtuálními stroji ve (virtuální) síti. Na prvním je stoji je Windows 7 64bit, na druhém je Windows 7 64bit s analytickými nástroji a na třetím je Linux server, konkrétně varianta Remnux [35]. Tyto virtuální stroje jsem pojmenoval popořadě Win7-64b_Clean, Win7-64b_Analyze a Remnux.

Systém Remnux vytvořil pro výukové účely Lenny Zeltser, který vyučuje kurzy malware v rámci organizace SANS Institute[2] a předpokládám, že jeho systém bude rozsahem vhodný i pro účely mé studentské práce.

Virtuálnímu stroji Win7-64b_Clean je vyhrazeno 20 GB disku, 512 MB operační paměti. Virtuálnímu stroji Win7-64b_Analyze je vyhrazeno stejně prostoru jako stroji Win7-64b_Clean. Virtuálnímu stoji Remnux, určenému pro Remnux server, je vyhrazeno 25 GB disku a opět 512 MB operační paměti.

Stroj Win7-64bit_Clean je navržen pro testování škodlivého kódu z jiného systému než na kterém kód běží, tedy pomocí síťové aktivity, která je v mém případě zachytávána na serveru Remnux. Při testování tímto způsobem nemá škodlivý kód možnost najít žádné testovací nástroje a pojmout podezření. Stroj Win7-64bit_Analyze je pak navržen jako oběť, na které je současně s během škodlivého kódu prováděna analýza na stejném systému.

Návrh propojení virtuálních strojů je vyobrazen v kapitole 4.1

Zabezpečení prostředí proti úniku škodlivého kódu

Celé prostředí je kompletně odpojeno od Internetu. Virtuální stroje jsou úplně izolovány od skutečného OS a tím minimalizuji riziko úniku škodlivého kódu.

Zakázal jsem sdílené adresáře, zrušil síťové spojení na fyzický stroj a zakázal jsem promiskutiní mód síťovým kartám virtuálních strojů, na fyzickém stroji zablokoval firewallem veškerou komunikaci (neprostupnost jsem ověřil nmap -PN skenem) a kontroloval jsem aktualizace Windows i VirtualBoxu. Samozřejmostí byl antivirový program na fyzickém stroji, konkrétně se jednalo o ESET SmartSecurity 6.

Simulování uživatelské aktivity

Prostředí jsem vybavil běžnými aplikacemi (webový prohlížeč, poštovní klient...). Dále je možné prostředí dle libosti doplnit daty uživatele (uložená hesla, emailové kontakty, konta, ...), aby měl škodlivý kód úspěšný lov.

V návrhu počítám s manuálním řízením uživatelské aktivity.

Pasivně založený škodlivý kód a boti očekávající příkazy od jejich bot-mastera se vůbec nemusí v mém prostředí projevit. Takovýmto kódem se zabývají například v projektu BotSwindler [33], kde na odhalení škodlivého kódu vkládají do virtuálního prostředí věrohodné návnady. Přístup do virtuálního prostředí v jejich projektu probíhá přes grafickou mezipaměť, přes kterou zasahují do běhu škodlivého kódu, podvrhují mu návnady a čekají jak škodlivý kód zareaguje.

2# Soukromá společnost založena 1989 v USA specializující se na školení v oblasti internetové bezpečnosti.

4.4. Návrh simulace síťových služeb

Celé prostředí je úplně odpojeno od Internetu a obsahuje simulované síťové služby na způsob klient-server modelu, kde na serveru běží služby DNS, IRC server, webový server, dále pošta SMTP a FTP server. Infikován bude klient a veškerá spojení budou přesměrována na server, který naslouchá na všech portech. Pro požadavky na porty, pro které není simulována služba, vrací server klientovi chybovou hlášku

Celkový síťový provoz budu sledovat Wiresharkem.

Alternativou je například tcpdump**.** Pro program Wireshark jsem se rozhodl vzhledem k propracovanosti filtrů, a grafickému uživatelskému rozhraní.

Služby na serveru simuluji nástrojem INetSim. Spustí služby, odpovídá na požadavky klienta a provoz loguje.

Alternativou k InetSimu by byla sada nástrojů fakeDNS nebo ApateDNS, inspireIRCd, fakeSMTP s tím, že každý nástroj je třeba zvlášť konfigurovat, spouštět a jeho log poskládat dohromady s ostatními. Tyto nástroje by dohromady poskytovaly obdobnou škálu síťových služeb jako InetSim.

Tabulka 8: Přehled síťových služeb simulovaných ve virtuálním prostředí

Typ služby	Proč je služba potřeba?	Jak jsem službu zprovoznil?
DNS	Škodlivý kód (viry, trojské koně, boti) má obvykle své „domovské“ adresy uložené DNS názvem, protože tím je adresa na rozdíl od IP nezávislá na fyzickém stroji. Správci škodlivého kódu tak stačí při výpadku jednoho serveru jen k DNS názvu přiřadit IP adresu jiného. Jedním z prvních kroků škodlivého kódu bude překlad uloženého DNS a proto potřebuji mít spuštěnou DNS službu, která bude na tyto dotazy odpovídat podvrženou IP adresu mého serveru.	**Pomocí nástroje INetSim.** Standardní odpověď nastavím na IP adresu simulovaného serveru – v mém případě *192.168.0.1*. Dále můžu nastavit hostname, doménu, verzi DNS a statické mapování (to přijde vhod zjistím-li předem, co zkoumaný škodlivý kód očekává a podvrhnu mu to).
HTTP HTTPS	Škodlivý kód může chtít stahovat nebo nahrávat data na webový server. Službu HTTP též využívá škodlivý kód operující s webovým prohlížečem (např. spyware). Služba je třeba k aktivaci http botů (echo/command-based) a v ideálním případě k umožnění a zachycení následné komunikace (BotSwindler) [33].	**Pomocí nástroje INetSim,** nastavitelná verze serveru. Umožňuje připravit podvrhnuté soubory pro GET požadavky na různé typy souborů (.txt,.html,.exe,.com...)

Typ služby	Proč je služba potřeba?	Jak jsem službu zprovoznil?
HTTP **HTTPS**	Například boti vytvoření ZeuS sadou nástrojů pro vývoj webového command-control botnetu zašlou během své instalace na jim nastavený server HTTP GET požadavek na stažení konfigurace (co sbírat) [34]. Nasbíraná data odesílají přes HTTP POST. Navíc pokud HTTP "200 OK" odpověď na POST obsahuje skrytý skript s příkazy, provedou jej a návrat. hodnotu odešlou s dalšími daty.	
IRC	Služba IRC je využívána množstvím škodlivého kódu. Především boty, kteří přes IRC přijímají příkazy od bot-masterů a komunikují s nimi. Bot často zná jen název IRC command-control serveru, který si nechá od DNS přeložit na IP adresu. Službu IRC používá například červ Hamweq [35]. Též DoS a DDoS útoky jsou často koordinované přes IRC kanál.	**Pomocí nástroje INetSim,** nastavitelný parametr verze.
SMTP **SMTPS**	Zjistím, jestli škodlivý kód rozesílá spam, šíří se emailem nebo odesílá nasbíraná data. Prohledám-li spam adresář pošty regulárním výrazem http:/.*{.cmd|.exe|.scr} nejspíš najdu škodlivý kód. Škodlivý kód používající SMTP má tradici, například VBS/Loveletter odesílající hesla přes smtp.super.net.ph, červy MyDoom, Sober, NetSky a další.	**Pomocí nástroje INetSim,** nastavitelné parametry hostname, banner, autentizace.
POP3 **POP3S**	Opět pro škodlivý kód šířící se elektronickou poštou.	**Pomocí nástroje INetSim,** nastavitelné parametry hostname, banner, autentizace.
FTP **FTPS**	Pro škodlivý kód stahující nebo nahrávající data na FTP server. Aby škodlivý kód mohl provést útok na FTP server, případně zadat uživatelské údaje získané z infikovaného klienta.	**Pomocí nástroje INetSim,** nastavitelný parametr verze.

Interakce se škodlivým kódem

Nastavením toku síťových dat na portech 80 a 443 mezi Windows klientem a Remnux serverem přes HTTP proxy na portu 8080 bylo umožněno tento tok pozměňovat. Modifikace, propuštění nebo zahození příchozích požadavků a simulovaných odpovědí je umožněna implementací nástroje BurpSuite Proxy na Remnux serveru. Implementaci podrobněji popisuji v Kapitole 5.3

4.5. Návrh analýzy škodlivého kódu

Dynamická a behaviorální analýza

Přehled o chování zkoumaného programu získám sledováním, jak zachází se soubory, registry, procesy a sítí. To zjistím především pomocí Process Monitoru a Wiresharku. Oba mají propracované filtrování a monitorují celou svou oblast. To se uplatní při rozlišování běžné aktivity systému od té specifické pro škodlivý kód. Rozhodující tudíž bude nástrojům vhodně navrhnout filtry vstupu.

Fáze analýzy

V první fázi budu testovat škodlivý kód na systému Win7-64bit_Clean připojenému k serveru Remnux. Na systému Win7-64bit_Clean určenému k infikování pouze nechám běžet po určitou dobu škodlivý kód a analýzu budu provádět vně tohoto virtuálního systému. Budu sledovat síťovou aktivitu zachycenou na serveru Remnux.

V další fázi budu testovat škodlivý kód na systému Win7-64bit_Analyze připojenému k serveru Remnux. Analýzu budu provádět přímo na napadeném systému Win7-64bit_Analyze. Předpokládám, že nejvíce stop zanechává škodlivý kód na napadeném systému. Komunikaci přes síť budu sledovat na serveru Remnux. Jedna z nevýhod je, že škodlivý kód může rozpoznat, že je sledovaný nejen detekcí virtualizace.

Oblasti analýzy

Během analýzy škodlivého kódu se zaměřím na zjišťování poznatků o následujících oblastech jeho činnosti:

- Síťová aktivita
- Analýza procesů
- Analýza souborů
- Změny registrů Windows

Cyklus analýzy

1. Vytvoření snímku virtuálního stroje Win7 určeného k infikování v „čistém“ stavu před napadením
2. Přípravné kroky před spuštěním škodlivého kódu (spuštění simulace služeb)
3. Nahrání škodlivého kódu na virtuální stroj určený k infikování
4. Spuštění škodlivého kódu (analýza síťové komunikace a procesů)
5. Po ukončení běhu škodlivého kódu přerušení/pozastavení virtuálního stroje Win7 určeného k infikování (analýza paměti a souborů)
6. Obnova virtuálního stroje určeného k infikování ze snímku „čistého“ stavu před napadením

5. Implementace

Během implementace virtuálního prostředí na Windows jsem stále více oceňoval výhody konfigurace Oracle VirtualBoxu z příkazové řádky oproti grafickému rozhraní. Výhodami míním rozsáhlejší možnosti konfigurace, rychlejší odezvu nebo skriptovatelnost, souběžně s tím jsem však přicházel na omezení rozhraní příkazové řádky *cmd.exe*. Vzhledem k tomu jsem občas používal *Windows PowerShell*[3] s rozšířenou podporou skriptování a integrací .Net prostředí. Uvažoval jsem dokonce o použití sbírky linuxových nástrojů *Cygwin*[4] či migraci pracovního prostředí fyzického stroje z Windows na Linux. Nakonec jsem jako východisko rozšířil Windows fyzického stroje o platformu jazyka *Python*, kterou VirtualBox speciálně podporuje a postupně se učil jazyk ovládat a aplikovat jej na řízení, případně analýzu, virtuálních strojů.

Pro správnou práci prostředí je třeba zapnout v BIOSu hostitelského počítače podporu technologie virtualizace.

5.1. Instalace virtuálních strojů

Ve VirtualBoxu jsem založil tři virtuální stroje podle návrhu v Kapitole 4. Klienty určené k infikování jsem označil *Win7-64b_Clean* a *Win7-64b_Analyze*, řídící server pak *Remnux*.

Podle návrhu v Kapitole 4 má být virtuální prostředí izolováno od fyzického hosta. Proto jsem Windows strojům určeným k infikování v '*Nastavení*' grafického konfigurační rozhraní *'Oracle VM VirtualBox Správce'* úplně zakázal vzdálený server obrazovky (*RDP*[5]), podporu USB zařízení a sdílené složky. Toho lze též dosáhnout nástrojem příkazového řádku *VboxManage.exe*, který umožňuje plnou kontrolu nad VirtualBoxem a virtuálními stroji z příkazového řádku hostitelského OS, obvykle fyzického počítače. Standardně se nachází v adresáři VirtualBoxu a zmíněnou izolaci prostředí lze nastavit příkazy v *CMD* programem

cd %PROGRAMFILES%\Oracle\VirtualBox\vboxmanage

Nastavení virtuálních strojů Windows 7 i Remnux jsem zamkl heslem příkazem

vb*oxmanage --settingspw V!rtua1B0x-2013-Ma1war8*

3# O PowerShellu například na http://en.wikipedia.org/wiki/Windows_PowerShell nebo přímo příkazem SystemRoot%\system32\WindowsPowerShell\v1.0\powershell.exe help about_Windows_PowerShell_2.0

4# http://www.cygwin.com/

5# Remote Desktop Protocol; Je-li virtuálnímu počítači povolen RDP server, umožní vzdáleným klientům připojení a práci s běžícím virtuálním počítačem.

Heslo přednastavené pro přihlášení na virtuální stroje Windows 7 i Remnux je

malware

Windows Win7-64-bit_Clean

Ve VirtualBoxu jsem vytvořil nový virtuální stroj s parametry podle návrhu v Kapitole 4. Na tento virtuální stroj jsem nainstaloval Windows 7 64-bit ze staženého .iso obrazu s licenčním kódem z MSDNAA[37]. Abych mohl čas virtuálního stroje měnit relativně vůči skutečnému, vypnul jsem synchronizaci času s fyzickým strojem příkazem CMD

VBoxManage setextradata Win7-64b_Clean "VboxInternal/Devices/VMMDev/0/Config/GetHostTimeDisabled" 1

Linux Remnux

Zprovoznění USB flashdisku připojitelného z fyzického systému hosta k virtuálnímu Linux serveru

Tímto krokem jsem umožnil prostřednictvím USB flashdisku vkládání vzorků škodlivého kódu do zatím nepřístupného izolovaného virtuálního prostředí přes řídící Linux server a případné exportování reportů opět z tohoto stroje. Pro účely této práce byl vybrán 1 GB USB flashdisk iTE Tech[6] s nastavitelným hardwarovým chráněním proti zápisu.

Ve VirtualBoxu bylo třeba stroji Remnux povolit virtuální řadič a přidat podporu USB zařízení buď pomocí grafického konfiguračního programu *'Oracle VM VirtualBox Správce'* v *'Nastavení'* stroje Remnux pod položkou *'USB'* povolit USB ovladač a přidat prázdný filtr USB zařízení (například klávesou *Ins*)

anebo nástrojem příkazového řádku VboxManage.exe

vboxmanage modifyvm Remnux --usb on

Dále jsem využil příkaz *vboxmanage usbfilter*, jenž spravuje jak USB filtry konkrétních virtuálních strojů, tak globální filtry všech strojů virtuálního prostředí, přičemž globální filtry mají přednost. Zatímco na připojení flashdisku bude použit filtr konkrétní, právě vlastnost globálních filtrů poslouží k zaručenému odpojení flashdisku od virtuálního prostředí po infekci škodlivým kódem a následné analýzy. Samozřejmě jistotou je poté flashdisk odpojit fyzicky. Jeho předchozí softwarové odpojení má smysl nejen z hlediska

6# Například na Linuxu lze výrobce flashdisku vypsat příkazem *lsusb*

zabezpečení, ale především předchází poškození flashdisku[7].

Globální filtry připojených USB zařízení dostupných VirtualBoxu se vypíší příkazem

vboxmanage list usbhost

Pozor na rozdílnou funkčnost verzí VirtualBox *4.2.10 r84104* a *r84105*[8] vydaných 15. března 2013, kdy *r84104* nekorektně pracuje s USB, viz chybný výstup

C:\Program Files\Oracle\VirtualBox>vboxmanage list usbhost
Host USB Devices:
<none>

a chybová hláška zobrazující se v grafického konfiguračního programu *'Oracle VM VirtualBox Správce'* při přístupu k *'Nastavení'* virtuálního stroje pracujícího s USB zařízeními

Nepodařil se přístup na subsystém USB. Nepodařil se přístup na službu USB Proxy (VERR_FILE_NOT_FOUND) ... E_FAIL (0x00004005).

Příkaz *vboxmanage usb filter add* pro vytvoření USB filtru vyžaduje alespoň tři parametry. Parametr index určuje pozici na seznamu, kam se filtr přidává. Dále se buď parametrem *target* vybere konkrétní virtuální stroj, anebo se parametrem *global* přidá filtr na všechny virtuální stroje. Obecně doporučuji USB flashdisk vkládat jen do USB 2.0 portů, protože chyby se mi objevovaly jen při zapojení flashdisku do USB 3.0 portů. Na fórech VirtualBox se lze dočíst, že funkční podpora USB ve virtuálním prostředí je relativně novou záležitostí. Nejnovější verze VirtualBox *4.2.12 r84980* vydaná 12.dubna 2013 podle historie změn [11 OracleChangelog 4.2] podporu USB 3.0 explicitně nezmiňuje.

Windows Win7-64b_Analyze

Virtuální stroj Win7-64b_Analyze byl vytvořen jako klon virtuálního stroje Win7-64bit_Clean. Využil jsem možnosti nástroje VirtualBox vytvořit kopii již existujícího virtuálního stroje včetně jeho disků pomocí procesu klonování.

Na takto vzniklý virtuální stroj byly nainstalovány analytické nástroje ESET SysInspector, ProcessMonitor a CaptureBAT.

7# Flashdisk připojen virtuálními ovladači jsem jednou bez řádného odpojení vyjmul z fyzického počítače a pak už jen složitě obnovoval poškozené sektory.

8# http://download.virtualbox.org/virtualbox/4.2.10/

5.2. Simulace síťových služeb

Konfigurace virtuální sítě

Mezi Windows7 a Linux (Remnux) jsem vytvořil simulované síťové služby pomocí nástroje INetSim.

Linux Remnux

/etc/network/interfaces

auto eth0	
iface eth0 inet static	
address	*192.168.0.1*
netmask	*255.255.255.0*
network	*192.168.0.0*
broadcast	*192.168.0.255*

/etc/resolv.conf

nameserver	*192.168.0.1*

- Editorem *nano* v řežimu *sudo*
- V příkazovém řádku nastavit pomocí *ifconfig*

Windows Win7-64-bit_Clean

Konfigurace TCP/IP (protokol IPv4)

Static IP:	*192.168.0.2*
Subnet mask:	*255.255.255.0*
Gateway:	*192.168.0.1*
DNS:	*192.168.0.1*

- Grafickým rozhraním pro změnu vlastností adaptéru *'Připojení k místní síti'*
- V příkazovém řádku lze nastavit příkazem *route*

Windows Win7-64-bit_Analyze

Konfigurace TCP/IP (protokol IPv4)

Static IP:	*192.168.0.3*
Subnet mask:	*255.255.255.0*
Gateway:	*192.168.0.1*
DNS:	*192.168.0.1*

INetSim

Obecně jsou u všech parametrů konfiguračního souboru programu INetSim nastavené výchozí hodnoty, nicméně řadu parametrů jsem nastavil explicitně:

nano /etc/inetsim/insetsim.conf

/etc/inetsim/insetsim.conf

dns_default_domainname service.org

dns_version "9.2.4"

http_version "Microsoft-IIS/8.0"

https_version "Microsoft-IIS/8.0"

smtp_fqdn_hostname mail.service.org

smtp_banner "SMTP Mail Service ready."

smtps_fqdn_hostname mail.service.org

smtps_banner "SMTPS Mail Service ready."

pop3_banner "POP3 Server ready"

pop3_capability IMPLEMENTATION "POP3 Server"

pop3s_banner "POP3 Server ready"

pop3s_capability IMPLEMENTATION "POP3s Server"

ftp_version "vsFTP 2.0.4 - secure,fast,stable"

ftp_banner "FTP server"

ftps_version "vsFTP 2.0.4 - secure, fast, stable"

ftps_banner "FTP Server ready"

irc_fqdn_hostname irc.service.org

irc_version "Unreal3.2.7"

#redirect_external_address 10.10.10.1

redirect_exclude_port tcp:22

redirect_ignore_netbios yes

Nástroj rovnou loguje příchozí komunikaci včetně odeslaných podvržených odpovědí. Standardně do souboru */var/log/inetsim/report/report.$PID.txt .*

Ze simulace je vynechán port 22 *(SSH)*, aby bylo možné skutečné a současně zabezpečené síťové spojení se serverem, jednak pro přenos škodlivého kódu na stroj Windows určený k infikování a jednak pro případné nahrání dílčích logů z Windows klienta na Remnux server. Po přenosu škodlivého kódu je možné port 22 *(SSH)* opět přidat do simulace.

INetSim nesimuluje Windows služby a tudíž jsem nastavil, aby nereagoval na služby NetBIOS a RPC, které simuluji skripty fakeNetbios.py a RPCd.

Do *~/.bash_aliases* jsem přidal zkratku pro spuštění INetSimu ve vlastním terminálu

alias inetsim-launch='lxterminal --geometry=70x30 --title=INETSIM -e '"'"'bash -c "sudo inetsim; read" '"'"'

Výpis všech otevřených TCP/UDP portů Remnux serveru je možné provést příkazem

alias ports='netstat -tulanp'

Detekce připojení Windows

Pozoroval jsem ve Wiresharku, že neinfikovaný Windows klient hned po startu sám od sebe zasílá nejprve DNS dotaz na *www.msftncsi.com* a následně HTTP požadavek */ncsi.txt*. Podle [20] tímto způsobem Windows ověřují svou konektivitu na Internet, jedná se o službu Microsoft NCSI[9].

První krokem ověřování připojení je zmíněný DNS dotaz na *www.msftncsi.com* a očekávání *IP 131.207.255.255* jako odpovědi. Druhým krokem je pak stažení textového souboru *ncsi.txt*, který obsahuje pouze "Microsoft NCSI", z této domény.

Rozhodl jsem se pro lepší věrohodnost prostředí simulovat uvedené reálné odpovědi. Nepomůže nastavit INetSimu na tento DNS dotaz odpovídat očekávanou IP adresou místo standardní IP Remnux serveru. Použil jsem postup naopak na Windows klientovi změnit v Regeditu pod klíčem

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\NlaSvc\Parameters\Internet

parametr *ActiveProbeHost* z očekávané IP na IP Remnux serveru. Ověřil jsem, že ostatní parametry týkající se ověřování dostupnosti domény *msftncsi.com* je již INetSim schopen nasimulovat.

V [16] se též uvádí, jak pomocí stejného klíče Windows ověřování konektivity NCSI zcela zakázat, což však nechci, neboť mám zájem zdánlivou dostupnost síťových služeb prostředí zvyšovat. Následně jsem INetSimu do složky s podvrženými soubory vytvořil soubor *ncsi.txt* a nastavil jeho odesílání přes *http_static_fakefile.*

9#NCSI „Network Conectivity Status Indicator " je součást 'Připojení k síti' zavedená od Windows Vista. http://technet.microsoft.com/en-us/library/cc766017(v=ws.10).aspx

5.3. Vybavení prostředí analytickými nástroji

Wireshark

Na server Remnux jsem nainstaloval program Wireshark. Odchytávání příchozí a simulované odchozí komunikace na mé virtuální síti spustím příkazem

wireshark -i eth0 -k

BurpSuite Proxy

Na řídícím Remnux serveru je nastaveno proxy na portu 8080. Použitím nástroje Burp jsem umožnil modifikaci, propuštění nebo zahození příchozích požadavků a odchozích odpovědí.

Nejprve jsem změnil přes *Proxy > Options > Edit proxy listener* rozhraní na kterém naslouchá Burp z *loopback only* na IP Remnux serveru a výchozí port ponechal stejný (*192.168.0.1:8080*). Pro zprovození proxy bylo třeba změnit nastavení síťových spojení řídícího Remnux serveru. Vytvořil jsem v *iptables* následující pravidla pro přesměrování libovolné HTTP (tcp:80) a HTTPS (tcp:443) komunikace na port 8080 s naslouchajícím Burpem

sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-ports 8080

sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-ports 8080

(sudo iptables -t nat -A PREROUTING -p tcp -m multiport --dports 80,443 -j REDIRECT --to-ports 8080)

Eventuálně lze pravidla omezit na konkrétní rozhraní přidáním *-i eth0*, ovšem protože používám právě jen *eth0* není omezení nutné.

Po provedení těchto nastavení příchozí HTTP/HTTPS požadavek nejprve přijme Burp, avšak protože je v režimu *invisible-proxy* a *non-proxy aware* klienti se připojují přímo na něj, bez nastavení přesměrování na IP Remnux serveru vytvoří Burp smyčku a místo přeposlání přijatých paketů INetSimu odpoví Windows klientu chybovou hláškou. Například pokud škodlivý kód z Windows klienta odešle HTTP požadavek na adresu *virus.org*, bez přesměrování na IP Remnux serveru nebude Burp znát kam po volbě *Forward* požadavek *Request to http://virus.org:80 [unkown host]* poslat a místo přeposlání požadavku INetSimu rovnou odešle Windows klientovi chybovou hlášku *HTTP1.0/502 Bad Gateway* obsahující *Burp proxy server error: Unkown host: virus.org*.

Je proto ještě třeba nastavit Burpu přes *Edit proxy listener* pod záložkou *Request handling* přesměrování na IP Remnux serveru *Redirect to host: 192.168.0.1* a Burp je nyní pro příchozí požadavky vložen mezi Windows klienta a INetSim. V *Proxy > Options*

> Intercept Client Requests jsem z pravidla, které požadavky nezachytávat, odebral javascript i grafické soubory a tím je zachytáváno všechno.

Otázkou je jaký může mít smysl zachytávat Burpem příchozí požadavky, když je INetSim pak stejně „zahodí“ a odpoví podle přednastavených vzorů *(inetsim.conf)*. Podle mne právě proto, aby analytik například lépe věděl co změnit na odpovědích INetSimu podle zachycených požadavků, ačkoliv jsou samozřejmě další možnosti, jak toho dosáhnout třeba provést *replay-attack* vybraného požadavku škodlivého kódu. Též pokud se rozhodne, že na vybraný požadavek škodlivého kódu není žádoucí odpovídat, může jej nyní rovnou zahodit bez přeposílání INetSimu a následného zahození simulované odpovědi.

Vzhledem k tomu, že chci také umět zachytit a upravit nebo zahodit odpovědi INetSimu, předpokládal jsem po zprovoznění dosud popsaných kroků mylně, že bude potřeba doplnit *iptables* o pravidla pro odchozí komunikaci. Ve skutečnosti stačilo Burpu v *Proxy > Options > Intercept Server Responses* aktivovat volbu *Intercept responses based on following rules* včetně nastavení vhodného pravidla například stejného jako u příchozích požadavků a Burp začal fungovat obousměrně.

Během „ladění“ potřebných *iptables* pravidel jsem doplnil prostředí o jednoduchý bash skript pro resetování pravidel *iptables-flush, /usr/bin/iptables-flush*

```
#!/bin/bash
    # Reset default policies
    iptables -P INPUT ACCEPT
    iptables -P FORWARD ACCEPT
    iptables -P OUTPUT ACCEPT
    iptables -t nat -P PREROUTING ACCEPT
    iptables -t nat -P POSTROUTING ACCEPT
    iptables -t nat -P OUTPUT ACCEPT
# Flush all rules, delete non-default (user)  chains
    iptables -F; iptables -X; iptables -t nat -F ; iptables -t nat -X
```

Do *~/.bash_aliases* jsem přidal následující zkraty pro výpisy *iptables*

```
    alias iptables-ls='sudo iptables -L -n -v'
    alias iptables-lsn='sudo iptables -t nat -L -n -v'
```

Navíc bylo třeba pravidla přidaná do *iptables* uložit a automaticky obnovovat při startu *iptables-save > /root/sim.fw (*po *sudo -s)* a přidat pro obnovu buď do */etc/rc.local* řádek *iptables-restore < /root/sim.fw* nebo do */etc/network/interfaces* u sekce *eth0* řádek *post-up iptables-restore*. O *iptables* více přímo z linuxových manuálových stránek a [15].

6. Testování

Překvapivě první zaznamenanou aktivitou vůbec byl pokus Windows o připojení do centrály Microsoftu, pravděpodobně šlo o aktivaci produktu, nicméně zajímavé je, že toto chování vlastně vykazuje znaky shodné s aktivitou škodlivého kódu, kdy také obvykle první, o co se pokouší je ohlásit se tvůrcům a například oznámit napadení dalšího počítače.

6.1. Navržení experimentů

V této části jsou uvedeny jednotlivé kroky v navrženém průběhu experimentů.

i. Výběr vzorku škodlivého kódu

Vyberu vzorek, který chci testovat ve vytvořeném prostředí. Vybraný vzorek škodlivého kódu nahraji na USB flashdisk

Virtuální stroj Remnux

ii. Spuštění aktuálního virtuálního stroje

Stroj lze spustit buď z grafického konfiguračního rozhraní nebo příkazem

vboxmanage startvm Remnux

Pokud bude žádoucí spustit snímek virtuálního stroje, pak lze opět buď z grafického rozhraní pod položkou *'Snímky (n)'* nebo se příkazu *vboxmanage startvm* dá jako parametr hash požadovaného snímku, který lze vypsat (a po kliknutí pravým tlačítkem zkopírovat nabídkou označit) například následujícím CMD příkazem

*vboxmanage list -l vms | findstr /R "^.*Name.*Win.*$ ^.*Name.*Rem. *$ ^UUID.*$"*

iii. Nahrání vybraného vzorku škodlivého kódu

Nahrávat je možné z flashdisku připojeného ke stroji Remnux přes vnitřní virtuální síť rovnou na virtuální stroj Windows určený k infikování pomocí SSH spojení.

Další možností je zkopírovat vzorek škodlivého kódu z flashdisku připojeného ke stroji Remnux do adresáře podvrhovaných souborů programu INetSim

sudo cp /media/FlashUSB/malware.zip /var/lib/inetsim/http/fakefiles/sample.exe

Po spuštění systému Windows bude možné soubor stáhnout pomocí webového prohlížeče přes vnitřní virtuální síť z adresy

http://service.org/malware/sample.exe

iv. Spustit simulaci síťových služeb a monitorovací nástroje

Simulaci síťových služeb a monitorovací nástroje spustím pomocí připraveného skriptu *start.sh*, který se nachází na ploše systému Remnux. Tím spustím INetSim, Wireshark a Burp se všemi potřebnými parametry. Příkazy skriptu *start.sh*

```
#!/bin/bash
#Launch simulation and monitoring
#inetsim-launch without alias :
lxterminal --geometry=70x30 --title=INETSIM -e 'bash -c "sudo inetsim; read"' &
#Real-time log viewing :
lxterminal --geometry=70x30 --title=LOG-INETSIM -e 'bash -c "tail -f
  /var/log/inetsim/service.log; read"' &
sudo wireshark -i eth0 -k &
burp &
```

Virtuální stroj Win7-64b_Clean

v. Spustit vzorek škodlivého kódu

Spustím vzorek škodlivého kódu po dobu přibližně 10 minut, pozoruji jeho chování a sleduji síťovou komunikaci.

Virtuální stroj Win7-64b_Analyze

vi. Spustit monitorující a analytické nástroje

Spustím ESET SysInspektor, ProcesMonitor a CaptureBat.

vii. Spustit vzorek škodlivého kódu

Spustím vzorek škodlivého kódu po dobu přibližně 10 minut a pozoruji jeho chování a sleduji síťovou komunikaci.

6.2. Výběr vzorků škodlivého kódu

První vzorek škodlivého kódu jsem si obstaral z webu věnovaného výuce analýzy škodlivého kódu http://zeltser.com[10]. Ostatní jsou z databáze Open Malware[11].

Open Malware je databáze společnosti pro výzkum a analýzu škodlivého kódu. Výrazně varuje, že obsahuje „živý" škodlivý kód, což přesně chci použít ve virtuálním prostředí pro analýzu. Jedná se rozsáhlou veřejnou databázi vzorků škodlivého kódu, které je možné vyhledávat podle názvu, MD5, SHA1, SHA256 hashů nebo obvyklé signatury. Umožňuje stahování škodlivého kódu v .zip archivu s heslem 'infected'. Před zahájením stahování je po uživateli požadována identifikace přihlášením na Google účet, aby prokázal, že není třeba automatický nástroj na stahování virů. Databáze Open Malware je hostována na serverech Georgia Tech Information Security Center, Georgia Institute of Technology, Atlanta, USA.

http://www.offensivecomputing.net/

http://oc.gtisc.gatech.edu:8080/

Obrázek 3: Open Malware stránka pro stahování virů

10# zeltser.com/reverse-malware/live-messenger-malware.zip

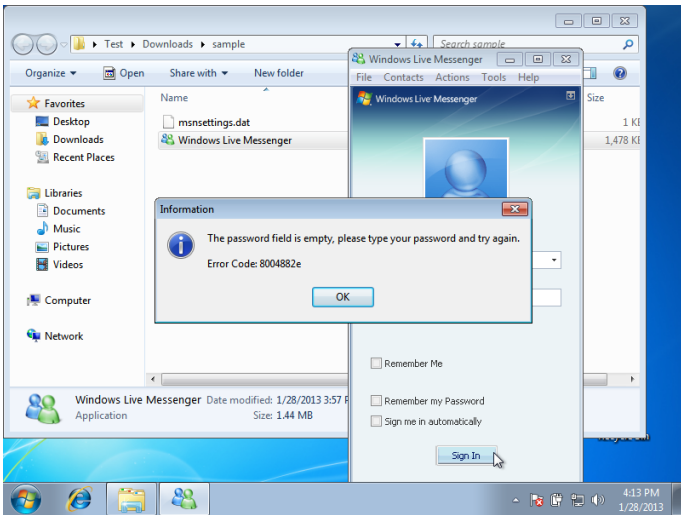
11# http://oc.gtisc.gatech.edu:8080/

6.3. Popisy provedených experimentů

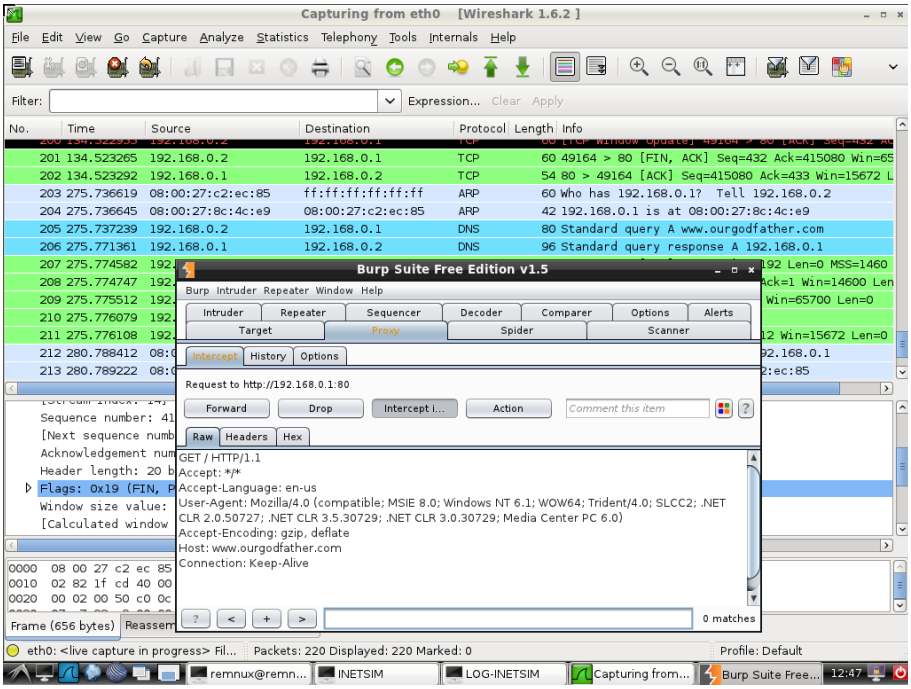
Experiment 1

Vzorek: LiveMessangerMalware
MD5: 78cf2489ad068d8eeae176590bd65fdb
Systém: Win7-64bit_Clean

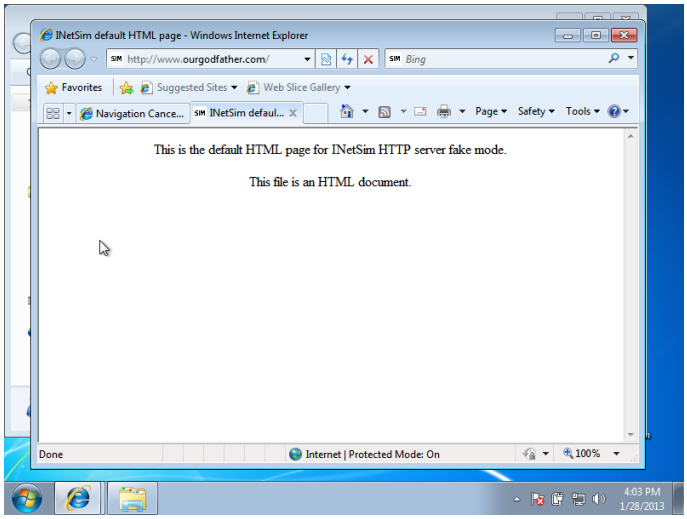
Po spuštění vzorku se spustil program Live Messanger, ovšem poněkud omezená verze. Pokud se pokusím přihlásit bez zadání e-mailové adresy a hesla, objeví se chybová hláška požadující zadání hesla. Pokud se pokusím přihlásit se zadanou e-mailovou adresou i heslem, objeví se hláška oznamující problémy s přihlášením. Doporučované tlačítko *Troubleshoot* nijak nepomůže, ovšem funguje tlačítko *Cancel* a můžu se opět vrátit k hlavnímu oknu. Lišta nabídek sice obsahuje nějaké záložky, ale zřejmě k nim nejsou naprogramovány žádné aktivity. Až doposud se vzorek nesnaží inicializovat žádnou síťovou aktivitu. Pokud se rozhodnu vypnout tento poněkud nefunkční program, objeví se náhle se projev síťové aktivity, konkrétně se vzorek snaží načíst internetovou stránky na adrese *http://www.ourgodfather.com*.



Obrázek 4: Screenshot Win7 LiveMessenger



Obrázek 5: Síťová komunikace vzorku LiveMessenger



Obrázek 6: LiveMessanger

Pomocí vytvořeného prostředí jsem ověřil, že se nejedná o pravý Live Messanger program. Například se po pokusu o přihlášení nesnaží vytvářet žádnou síťovou aktivitu, ovšem zjistil jsem, že po zavření okna programu se najednou snaží načíst v internetovém prohlížeči stránku *http://www.ourgodfather.com.* Podle *whois* záznamu tato adresa patří web-hostingové společnosti *ENOM, Inc.,* jako kontakt na registranta je uveden *E. Viva (ogfathersupport@gmail.com)* a v současné době je k pronajmutí.

Experiment 2

Vzorek: Worm.IRC.Seteada
MD5: 3d75da19901650ffa614d013677633ba
Systém: Win7-64bit_Clean

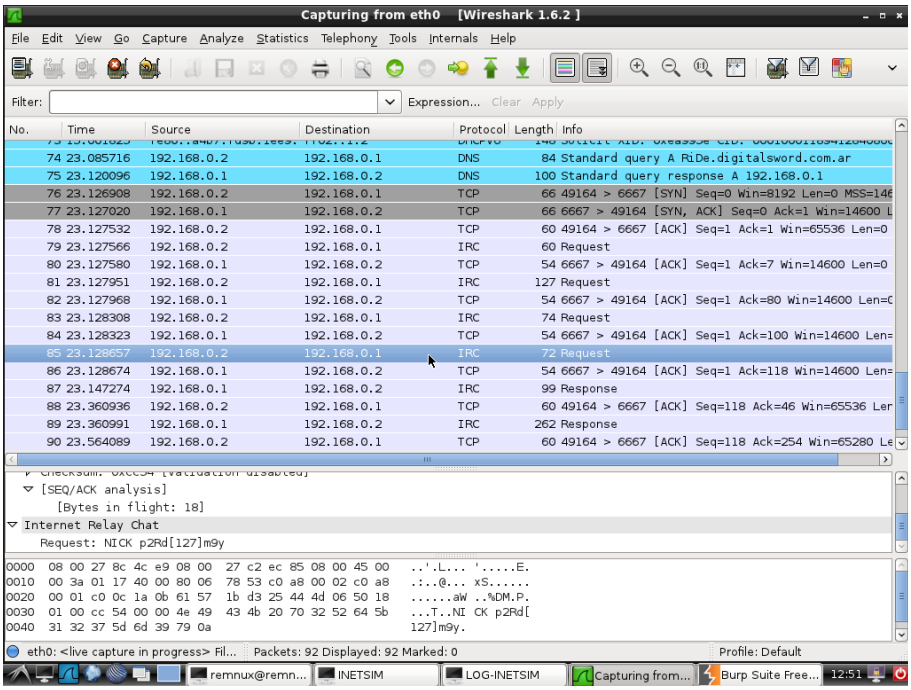
Po spuštění škodlivého kódu bylo jeho první síťovou aktivitou zaslaní DNS požadavku na URL adresu *RiDe.nightrun.com.ar*. Na to Remnux server odpověděl svou IP a škodlivý kód s ním začal komunikovat přes protokol IRC. Vzorek poslal na IRC port 6667 tři nešifrované dotazy očividně v domnění, že komunikuje se svým IRC serverem, konkrétně to byly dotazy:

Request: PASS

Request: NICK ipjaa
Request: USER ipjaa ipjaa ipjaa :ProtoTypr RiDe v2.3.0 (build 500)

Request: IRCX
Request: MODE ipjaa +i
Request: NICK p2Rd[127]7lb

Server Remnux odpověděl dvě odpovědi, nejprve server přivítal účastníka komunikace, následně oznámil chybějící parametr u hesla a tím komunikace skončila.



Obrázek 7: Síťová komunikace vzorku Worm.IRC.Seteada

Pomocí vytvořeného prostředí jsem rozpoznal, že se vzorek pokouší komunikovat přes IRC protokol. Podle *nic.ar* jsou argentinské domény *nightrun.com.ar* a *digitalsword.com.ar* momentálně k dispozici.

El dominio nightrun.com.ar se encuentra disponible.

El dominio digitalsword.com.ar se encuentra disponible.

Tento vzorek škodlivého kódu jsem spouštěl opakovaně a další adresa, na kterou se pokoušel přihlásit, byla *RiDe.digitalsword.com.ar*. První požadavek byl vždy PASS bez uvedení hesla, druhý a třetí požadavek již byly vygenerovány s přidáním prvku náhody, IRCX variantu požadoval vždy. Odhaduji, že po kvalitnější simulaci IRC serveru, buď vhodným nastavením INetSimu nebo doplněním o nástroj InspireIRCd dokáži déle udržovat komunikaci se vzorkem a dostanu více informací.

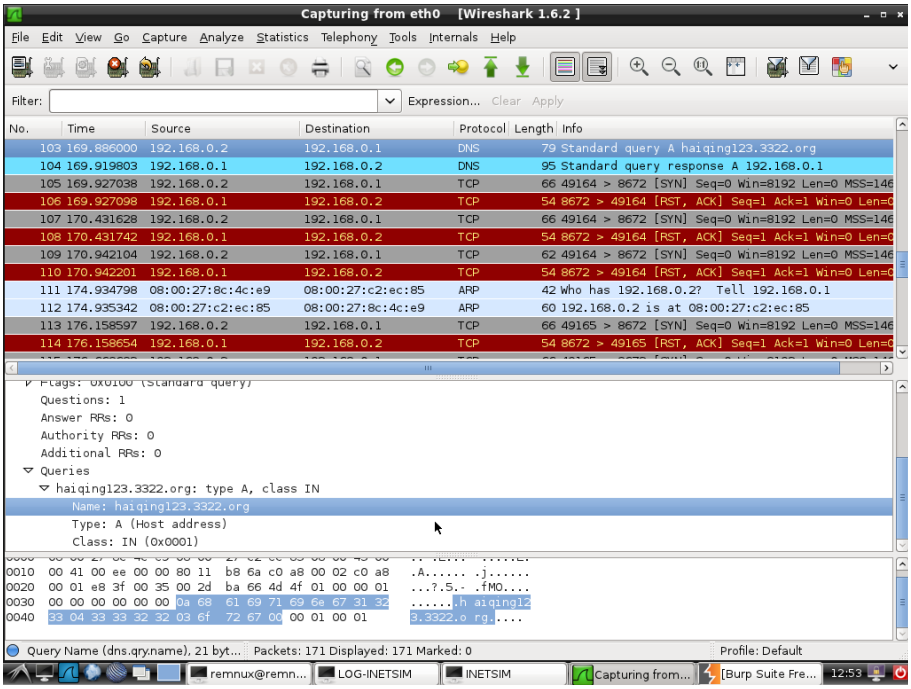
Podle zprávy Symantecu[12] se jedná o červa snažícího se šířit v lokální síti, který navíc může mít vlastnost backdoor. Při experimentech jsem nepozoroval snahu o šíření v lokální sítí, avšak zjistil jsem pokusy o komunikaci přes IRC protokol, které zpráva ani neuvádí.

Experiment 3

Vzorek: Trojan.Zlob.32625
MD5: 640a406b14557a7b0caaed7905984b30
Systém: Win7-64bit_Clean

Při spuštění vzorku s uživatelskými právy jsem nepozoroval žádnou viditelnou aktivitou ani síťovou komunikací. Po změně práv na administrátorská práva se situace hned zlepší. Vzorek se okamžitě snaží kontaktovat doménu *haiqing123.3322.org*. Nejprve se přes DNS požadavek snaží zjistit IP adresu, načež Remnux server odpoví posláním své IP adresy. Poté se vzorek snaží komunikovat z portu 49134 na port 8672 na cílovém serveru. Remnux server nesimuluje komunikaci na tomto portu a odpovídá chybovou hláškou.

12# http://www.symantec.com/security_response/writeup.jsp?docid=2004-011618-0828-99



Obrázek 8: Síťová komunikace vzorku Trojan.Zlob.32625

Pomocí vytvořeného prostředí jsem rozpoznal, že se vzorek pokouší o síťovou komunikaci na port 8672, který je podle seznamu registrovaných portů IANA (Internet Assigned Numbers Authority)[13] momentálně neobsazený.

Ve zprávě Symantec[14] jsem nalezl, že se vzorek pokouší vytvářet HTTP připojení na následující domény s použitím různých URL adres.

* vnp7s.net

* zxserv0.com

* dumpserv.com

Tím je podle zprávy umožněno vzorku posílat ping a hlásit svůj stav provozovateli škodlivého kódu a provádět instrukce uložené ve vzdálených souborech.

Domény *vnp7s.net*, *zxserv0.com* jsou momentálně na prodej, doména *dumpserv.com* je registrovaná ve Vietnamu u společnosti *April Sea Information Technology Corporation*.

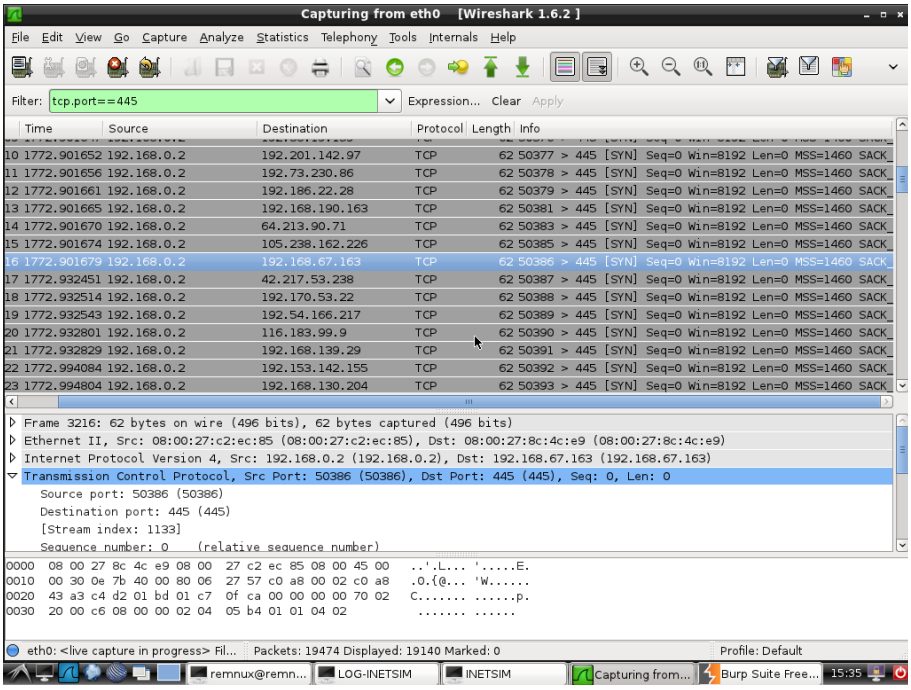
Doména *3322.org*, kterou jsem nalezl při mém zkoumání vzorku, je registrovaná u čínské společnosti *Bitcomm Ltd.* zabývající se poskytováním technologie cloudových výpočtů na profesionální úrovni, podle jejich tvrzení v originále na *whois.domaintools.com,* 作为云计算技术的专业服务提供商，中国公云(3322)为数百万用户提供云主机、智能域名、动态域名等基础信息服务。

13# http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml
14# http://www.symantec.com/security_response/attacksignatures/detail.jsp?asid=22910

Experiment 4

Vzorek: Net-Worm.Win32.Sasser.a
MD5: 1a2c0e6130850f8fd9b9b5309413cd00
Systém: Win7-64bit_Clean

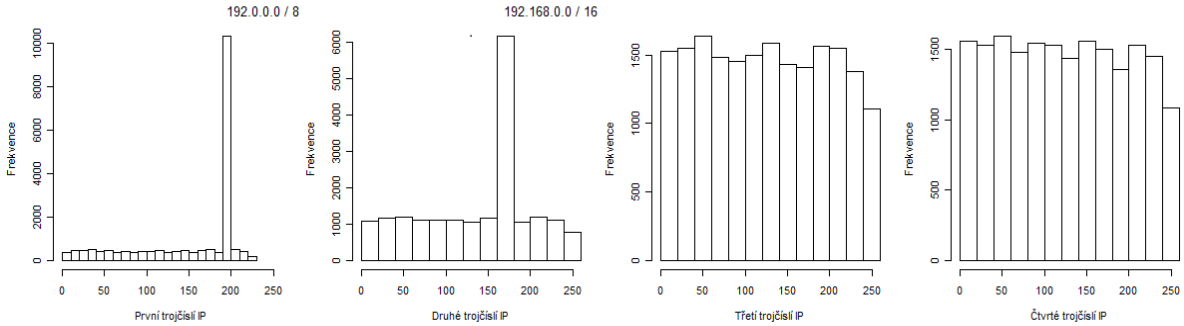
Vzorek se ihned po spuštění snaží kontaktovat velké množství IP adres pomocí protokolu TCP na portu 445. Tento port, poskytující službu Windows DS, je jedním z nejčastěji atakovaných portů. Vzorek se snažil atakovat náhodně vygenerované IP adresy. Vzhledem k tomu, že v mé simulované síti jsou jen dvě IP adresy (server Remnux a sytém Windows) nepodařilo se vzorku uhodnout žádnou funkční IP adresu.



Obrázek 9: Síťová komunikace vzorku Net-Worm.Win32.Sasser.a

Pomocí vytvořeného prostředí jsem rozpoznal, že se vzorek pokouší velmi aktivně inicializovat komunikaci na mnoha náhodně generovaných IP adresách na portu 445, který je registrovaný službou Microsoft-DS. Toto rozhodně není legitimní chování programu.

Zkoumal jsem rozložení IP adres, které zkoumaný vzorek generuje. Ukázalo se, že jsou preferovány útoky na IP adresy z rozsahu mé (vnitřní) sítě. Rozložení ostatních IP adres je poměrně rovnoměrné.



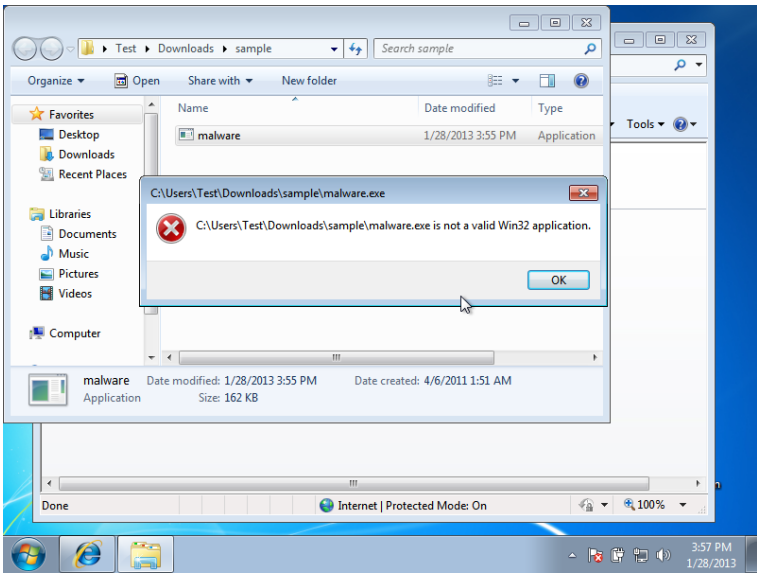
Obrázek 10: Histogramy IP adres generovaných vzorkem rozdělené podle trojčíslí IP adresy

Podle zprávy Symantecu[15] se jedná o variantu W32.Sasser.Worm snažící se využít zranitelnost LSASS popsanou ve zprávě Microsoft Security Buletin MS04-011[16]. Dále uvádí, že se vzorek snaží skenovat náhodně vybrané IP adresy při hledání zranitelných systémů, což potvrzuje mé pozorování.

Experiment 5

Vzorek: W32/Conficker!Generic
MD5: 0921282d4ed6008aa7c04e268d8367ae
Systém: Win7-64bit_Clean

Tento vzorek škodlivého kódu se nedaří v prostředí spustit. Během spouštění se objeví hláška o neplatné Win32 aplikaci.



Obrázek 11: Chybová hláška při snaze o spuštění vzorku

15# http://www.symantec.com/security_response/writeup.jsp?docid=2004-050114-1001-99
16# http://technet.microsoft.com/en-us/security/bulletin/ms04-011

Ve vytvořeném prostředí se mi nepodařilo vzorek spustit. Ne všechny programy musí být spustitelné v 64-bitovém prostředí Windows 7. Ověřil jsem spuštění i v režimu kompatibility a výsledek byl stejný. Navíc jsem zkoušel spustit i několik dalších variant škodlivého kódu Conficker, všechny reagovaly obdobně.

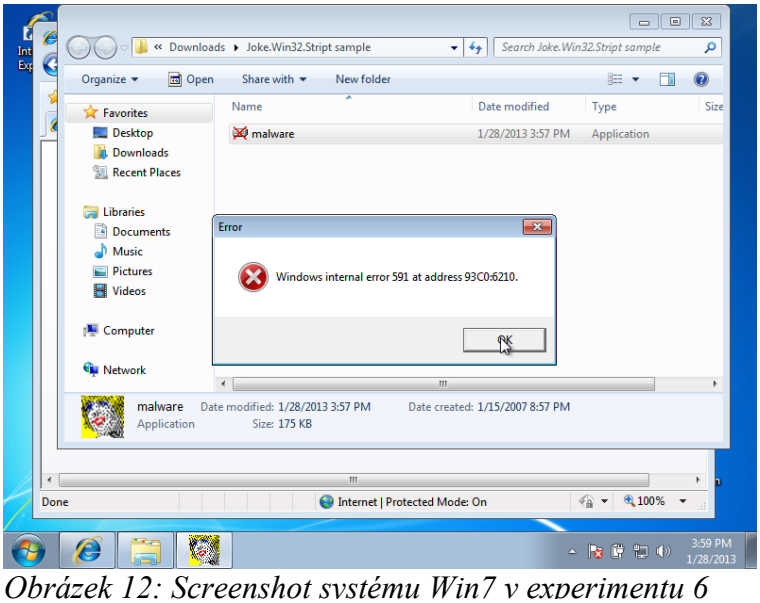
Očekávám více projevů, pokud se spustí vzorek v prostředí s jinými verzemi operačního systému Windows.

Experiment 6

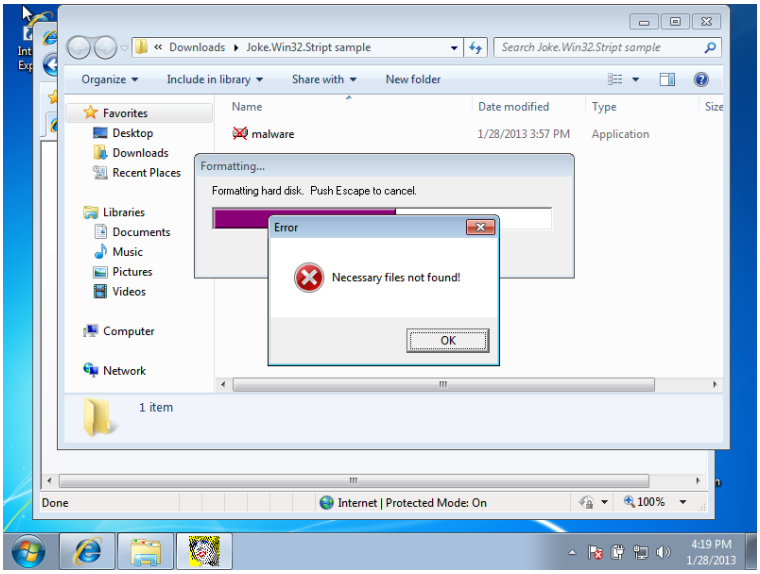
Vzorek: Joke.Win32.Stript
MD5: 09b1a069b765651243b6c84a3ad0b516
Systém: Win7-64bit_Clean

Hned po spuštění zobrazuje vzorek chybové hlášky, několik interních chyb a jednu fatální. Příčinou může být opět nekompatibilita s 64-bitovým prostředím. Poté zdánlivě nastoupí destrukční fáze, zobrazí se formátování disku, které však nedoběhne a po přerušení se nenaleznou nezbytné položky.

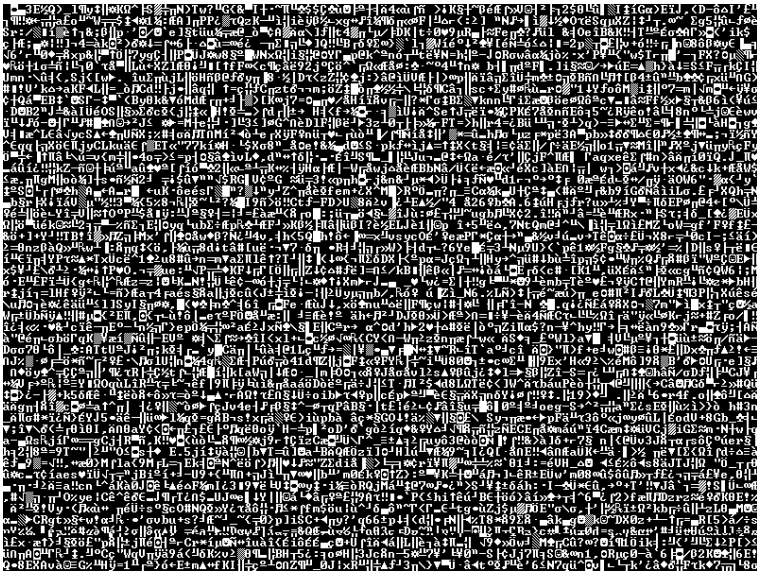
Následuje vypsání černé obrazovky. Na závěr se program ještě zeptá, zda vás vystrašil. Tento vzorek je tedy z kategorie vtipů. Tento vzorek škodlivého kódu se nepokouší inicializovat žádnou síťovou komunikaci.



Obrázek 12: Screenshot systému Win7 v experimentu 6



Obrázek 13: Screenshot systému Win7 v experimentu 6



Obrázek 14: Screenshot systému Win7 v experimentu 6

Pomocí vytvořeného prostředí jsem pozoroval chování vzorku. Vědomí o tom, že je program spuštěn ve virtuálním prostředí, mě poměrně uklidňovalo, obzvláště u hlášek o formátování sytému. Uvědomil jsem si, že pokud se mi virtuální systém smaže, tak se vlastně nic nestane, jen si obnovím předchozí snímek systému.

Symantec[17] popisuje, že zkoumaný vzorek zobrazí několik falešných chybových hlášek a poté předstírá formátování pevného disku. To se přesně shoduje s tím, co jsem pozoroval.

17# http://www.symantec.com/security_response/writeup.jsp?docid=2003-080712-4819-99

6.4. Zhodnocení

Celkové zhodnocení, jak se testovalo. Testovalo se dobře, nicméně je třeba počítat s tím, že ne všechen škodlivý kód je spustitelný na 64-bitových Windows 7.

Ve všech pokusech jsem zkoumal síťovou aktivitu škodlivého kódu spuštěného na operačním systému Windows 7 na stoji Win7-64bit_Clean. Řada vzorků projevila škodlivou aktivitu v mém virtuálním prostředí, ačkoliv jsou většinou staršího data a byly vytvořeny pro starší operační systém. Jejich komunikace obvykle brzo skončila, přesto se podařilo zachytit pokusy škodlivého kódu o navázání spojení se svým řídícím serverem.

Informace o škodlivém kódu jsem zjišťoval na stránkách antivirové firmy Symantec http://www.symantec.com a na stránkách Microsoft Malware Protection Centre http://www.microsoft.com/security/portal/.

Protože různé firmy přidělují stejnému škodlivému kódu různé názvy, potřeboval jsem nejprve zjistit pod jakým názvem vyhledávat škodlivý kód, který jsem zkoumal. Najít požadované názvy škodlivého kódu umožňuje například databáze https://www.virustotal.com. V této databázi lze vyhledávat škodlivé kódy podle MD5 hashe souboru škodlivého kódu, který je již poměrně jednoznačný. Na nalezené stránce je seznam antivirových společností spolu s názvy, které danému škodlivému kódu přidělili tyto společnosti při jeho identifikaci. Takto zjištěná jména škodlivého kódu jsem vyhledával na výše uvedených stránkách firmy Symantec a Microsoft.

Informace zjištěné pro hledaný škodlivý kód nejsou často na těchto a podobných stránkách uvedeny příliš podrobně. Pokud se firma hledaným škodlivým kódem zabývala podrobněji, obsahují uvedené informace většinou seznam změn, které daný škodlivý kód provádí, spuštěné procesy, změny v registrech Windows, v souborech, případně v systémových souborech Windows. Není zde kladen důraz na informace o jeho síťové aktivitě.

Používaný stroj Win7-64bit_Clean je navržen pro testování škodlivého kódu z vně systému na kterém běží vzorek, pomocí síťové aktivity, která je v mém případě zachytávána na serveru Remnux.

Kromě rizika, že škodlivý kód zaútočí přímo na virtualizační nástroje a obejde izolační mechanismy virtualizace, což lze považovat za akceptovatelné riziko podle [4], již nemá škodlivý kód jiné možnosti úniku, pokud mi postačí analýza ve virtuálním prostředí bez exportu dat.

U všech vzorků škodlivého kódu, které jsem testoval, jsem zachytil síťovou aktivitu, kterou vzorky vytvořily, pokud nějaká byla. Zachycenou komunikaci je možné si podrobně prohlédnout. Pokud se jedná o DNS dotaz, můžu pomocí záznamů Whois vystopovat, kdo vlastní danou doménu a kam se škodlivý kód pokoušel připojit.

Výhody prostředí oceněné během testování:

- vytvořené prostředí je dále rozšiřitelné podle vlastních požadavků
- v izolovaném prostředí je možno analyzovat podezřelé soubory bez vědomí tvůrců škodlivého kódu
- možnost průběžně a samostatně testovat jakékoliv podezřelé soubory
- prostředí není vázáno placenými licencemi
- rozpoznání jak a s jakými doménami vzorek komunikuje
- možnost vystopovat tvůrce škodlivého kódu, například podle zjištěných domén, aniž by se o pátrání dozvěděl
- možnost spouštět vzorek v předem připravených specifických situacích
- uživatel si sám ověří, co vzorek dělá, nemusí se spoléhat na testy cizích společností a tím se eventuálně vystavit úniku citlivých informací

7. Návrhy budoucích řešení

Jedním směrem, kterým by bylo vhodné se zabývat do budoucna je zlepšení simulace síťových služeb přidáním dalších protokolů. Většina současného škodlivého kódu vyžaduje síťovou konektivitu a proto lze očekávat, že její lepší simulace v řadě případů povede k zachycení více projevů síťové aktivity zkoumaného vzorku.

Omezit detekovatelnost virtuálních strojů škodlivým kódem. Řadu detekčních metod lze omezit změnou různých nastavení virtuálního prostředí, například změnou parametrů abstraktních ovladačů, nicméně zásadnější postupy vycházející z podstaty virtualizace založené například na časování instrukcí nebo zpomalení výkonu jsou obtížněji omezitelné. Na druhou stranu s rostoucí popularitou virtualizace bude třeba zhodnotit, zda vůbec škodlivý kód bude vykazovat snahu neprojevit se ve virtuálním prostředí.

Další vývoj práce může spočívat v rozšíření virtuálního prostředí přidáním honeypotu na Linux server, v automatizaci simulace uživatelské aktivity na Windows klientech či implementaci dalších analytických nástrojů na Win7-64b_Analyze. Též lze rozšířit portfolio obětí virtuálního prostředí o další verze Windows, případně jiné operační systémy.

Celé prostředí řídit z fyzického počítače s linuxovým operačním systémem místo z Windows a zhodnotit přínos systému s otevřeným kódem pro automatizaci řízení analýzy. V případě práce s VirtualBoxem z Linuxu zvážit využití emulačního nástroje IQEmu[18] Následně využít k analýze škodlivého kódu techniku introspekce virtuálního stroje VMI (Virtual Machine Introspection), která podle [2] vede k odhalení aktivity vzorku, jenž používá pokročilé metody skrývání (například rootkitů na úrovni kernelu).

Více se zaměřit na možnosti behaviorální analýzy aplikovat metody strojového učení (evoluční algoritmy, neuronové sítě, aj.) na data získaná prostředím k rozpoznávání škodlivého kódu, například klasifikovat vzorky do kategorií.

18# http://mirage335.dyndns.org/wiki/IQEmu

8. Závěr

V bakalářské práci jsem navrhl a implementoval virtuální prostředí pro analýzu škodlivého kódu. Nejprve jsem v teoretické části popsal škodlivý kód, statickou, dynamickou a behaviorální analýzu. Po zhodnocení stávajících řešení analýzy dynamické analýzy jsem se rozhodl v prostředí síťovou konektivitu škodlivému kódu poskytovat simulací síťových služeb. Většina řešení jen filtruje přístup na Internet. Aktuálnost mého řešení oproti řadě stávajících je pak v použití Windows 7 64bit jako systému určeného k infikování.

Prostředí bylo navrženo jako východisko pro aplikaci metod behaviorální analýzy.

V implementační části je popsána řada technických překážek, které bylo třeba během tvorby prostředí vyřešit, především konfigurace virtuální sítě.

Popsal jsem postup, jak testovat vzorky škodlivého kódu a funkčnost prostředí jsem následně na několika vzorcích otestoval. Ověřil jsem tím, že škodlivý kód se v prostředí projeví, řada jeho projevů byla zaznamenána a následně jsem poznatky z testování zhodnotil.

Výsledky získané testováním vzorků škodlivého kódu ve vytvořeném prostředí a následné porovnání mých zjištění se stávajícími poznatky ukazují, že rozboru síťové aktivity je ve zprávách, jak řady velkých firem v oboru, tak on-line analyzátorů, často věnována jen malá pozornost. Práce potvrdila, že navzdory omezeným prostředkům lze dosáhnout zajímavých zjištění o škodlivém kódu.

Vytvořené virtuální prostředí pro analýzu škodlivého kódu je tudíž použitelné pro základní analyzování známých i neznámých binárních souborů.

Cíle bakalářské práce jsem splnil. Virtuální prostředí bylo navrženo, implementováno a prostředí jsem aplikoval na analýzu několika vzorků škodlivého kódu s následným zhodnocením výsledků analýzy.

Seznam použité literatury

[1] COHEN, Frederick. Computer Viruses. Computer & Security. Volume 6, Issue 1. Pages 22–35. 1987

[2] HOOPES, John. *Virtualization for Security: including sandboxing, disaster recovery, high availability.* Burlington, MA: Syngress, 2009. 377 s. ISBN: 978-1-59749-305-5.

[3] LIGH, Michael H., et al. *Malware Analyst's Cookbook and DVD*. Indianapolis, IN: Wiley Publishing, 2011. 746 s. ISBN: 978-1-118-00336-7.

[4] EGELE, Manuel., et al. *A Survey on Automated Dynamic Malware Analysis Techniques and Tools [online].* ACM, 2010. Dostupné z WWW: https://www.seclab.tuwien.ac.at/papers/malware_survey.pdf

[5] BOWEN, Brian M., et al. *BotSwindler: Tamper Resistant Injection of Believable Decoys in VM-Based Hosts for Crimeware Detection [online]*. Columbia, CO: Department of Computer Science, Columbia University, 2010. Dostupné z WWW: http://people.csail.mit.edu/stelios/papers/botswindler_raid10.pdf

[6] DOSTÁLEK, Libor, KABELOVÁ, Alena. *Velký průvodce protokoly TCP/IP a systémem DNS*. Computer press, 2005.

[7] Microsoft Corpoation. *Microsoft Security Intelligence Report Volume 13 [online].* Microsoft, 2012. Dostupné z WWW: http://download.microsoft.com/download/C/1/F/C1F6A2B2-F45F-45F7-B788-32D_2CCA48D29/Microsoft_Security_Intelligence_Report_Volume_13_English.pdf [100 stránkový dokument od Microsoftu o stavu bezpečnosti]

[8] International Data Corporation. *The Dangerous World of Counterfeit and Pirated Software [online].* IDC, 2013. Dostupné z WWW: http://www.microsoft.com/en-us/news/download/presskits/antipiracy/docs/IDC 030513.pdf

[9] TU Wien. *International Secure Systems Lab*. Dostupné z WWW: www.seclab.tuwien.ac.at/

[10] Oracle Corporation. *VirtualBox User Manual [online]*. Dostupné z WWW: http://www.virtualbox.org/manual/UserManual.html

[11] Oracle Corporation. *Changelog for VirtualBox 4.2 [online].* Dostupné z WWW: http://www.virtualbox.org/wiki/Changelog

[12] REYS, Gleb. *What Hardware Virtualization Really Means [online]*. Desktop virtualization, 2008. Dostupné z WWW: http://www.desktop-virtualization.com/2008/05/14/what-hardware-virtualization-really-means/

[13] Microsoft Technet. *Scripting with Windows PowerShell [online]*. Dostupné z WWW: http://technet.microsoft.com/en-us/ scriptcenter/powershell.aspx

[14] Python Software Foundation. *Python Programming Language [online]*. Dostupné z WWW: http://www.python.org/

[15] MOWBRAY, Thomas J. *Advanced Log Analysis for Cyber Network Defense [online]*. 2011. Dostupné z WWW: http://www.antipatterns.com/resources/AdvancedLogAnalysis.pdf/

[16] BUENAVENTURA, Val. Pinoysecurity. *Data Mining Using Wireshark [online]*. 2010. Dostupné z WWW: http://pinoysecurity.blogspot.cz/2010/02/data-mining-using-wireshark.html/

[17] The Honeynet Project. Capture-BAT Download Page. Dostupné z WWW: http://www.honeynet.org/project/CaptureBAT

[18] SEIFERT, Christian. *Capture – A behavioral analysis tool for applications and documents [online]*. Digital Investigation, 2007. Dostupné z WWW: http://dfrws.org/2007/proceedings/p23-seifert.pdf

[19] Ubuntu. Official Documentation. *Iptables How To [online].* 2013. Dostupné z WWW: https://help.ubuntu.com/community/IptablesHowTo

[20] Superuser. Community blog. *Windows 7 Network Awareness: How Windows knows it has an internet connection [online].* 2011. Dostupné z WWW: http://blog.superuser.com/2011/05/16/windows-7-network-awareness/

[21] Oracle Corporation. End user forums for VirtualBox. *VirtualBox on Windows hosts*. Dostupné z WWW: https://forums.virtualbox.org/viewforum.php?f=6

[22] BOS, Herbert. Wombat Project. *Analysis Report of Behavioral Features.* 2012. Dostupné z WWW: http://www.wombat-project.eu/WP4/FP7-ICT-216026-Wombat_WP4_D16_V01_Analysis-Report-of-Behavioral-features.pdf

[23] SCHILLER, Craig A. *Botnets: The killer web app*. Syngress, 2007. 480 s. ISBN 1597491357.

[24] KEIZER Gregg. 64-bit Windows safer, claims Microsoft. *Computerworld*, 2009. Dostupné z WWW: http://www.computerworld.com/s/article/9141017/64_bit_Windows_safer_claims_Microsoft

[25] W3Schools. OS Platform Statistics. Dostupné z WWW: http://www.w3schools.com /browsers/browsers_os.asp

[26] StatCounter Global Stats. Top Operating Systems Per Country, Nov 2012. Dostupné z WWW: http://gs.statcounter.com/#os-ww-monthly-201211-201211-map

[27] NetMarket. Dostupné z WWW: http://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qptimeframe=M&qpsp=165&qpnp=1

[28] RAGAN Steve. Overview: Inside the Zeus Trojan's source code. The Tech Herald, 2011. Dostupné z WWW: http://www.thetechherald.com/articles/Overview-Inside-the-Zeus-Trojans-source-code/13567/

[29] KEIZER, Gregg. Windows 7 malware infection rate soars in 2012. Computerworld, 2012. Dostupné z WWW: https://www.computerworld.com/s/article/9232188/Windows_7_malware_infection_rate_soars_in_2012

[30] CASSELLA, Dena. Most Corporate PCs to Run 64-bit Windows by 2014, Says Gartner. Digital trends, 2009. Dostupné z WWW: http://www.digitaltrends.com/computing/most-corporate-pcs-to-run-64-bit-windows-by-2014-says-gartner/

[31] Steam. Hardwarový a softwarový výzkum služby Steam: March 2013. Dostupné z WWW: http://store.steampowered.com/hwsurvey/

[32] YEGULALP, Sedar. VMware Workstation 9 vs. VirtualBox 4.2 review. Techworld, 2012. Dostupné z WWW: http://review.techworld.com/virtualisation/3400693/vmware-workstation-9-vs-virtualbox-42-review/

[33] BOWEN, Brian M., et al. Botswindler: Tamper resistant injection of believable decoys in vm-based hosts for crimeware detection. In: Recent Advances in Intrusion Detection. Springer Berlin Heidelberg, 2010. p. 118-137.

[34] McDONALD, Doug. Zeus: God of DIY Botnets. Fortiguard. Dostupné z WWW: http://www.fortiguard.com/analysis/zeusanalysis.html

[35] BRANDT, Andrew. Hamweq Worm Brings a Mountain of Malware. Solera Network, 2013. Dostupné z WWW: http://www.soleranetworks.com/blogs/hamweq-worm-brings-a-mountain-of-malware/

[36] REMnux: *A Linux Distribution for Reverse-Engineering Malware.* Dostupné z WWW: http://zeltser.com/remnux/

[37] Microsoft Corporation. *DreamSpark for Academic Institutions.* Dostupné z WWW: http://msdn62.e-academy.com/jihoceskau_info

[38] Offensive Computing. *Open Malware About [online]*. 2005. Dostupné z WWW: http://www.offensivecomputing.net/?q=node/2

[39] RAINS, Tim. *Operating System Infection Rates: The Most Common Malware Families on Each Platform.* Microsoft Security Blog, 2013. Dostupné z WWW: http://blogs.technet.com/b/security/archive/2013/01/07/operating-system-infection-rates-the-most-common-malware-families-on-each-platform.aspx

[40] http://www.microsoft.com/en-us/windows/endofsupport.aspx

[41] LEBLANC, Brandon. *64-Bit Momentum Surges with Windows 7*. Microsoft Blog, 2010. Dostupné z WWW: http://blogs.windows.com/windows/b/bloggingwindows/archive/2010/07/08/64-bit-momentum-surges-with-windows-7.aspx

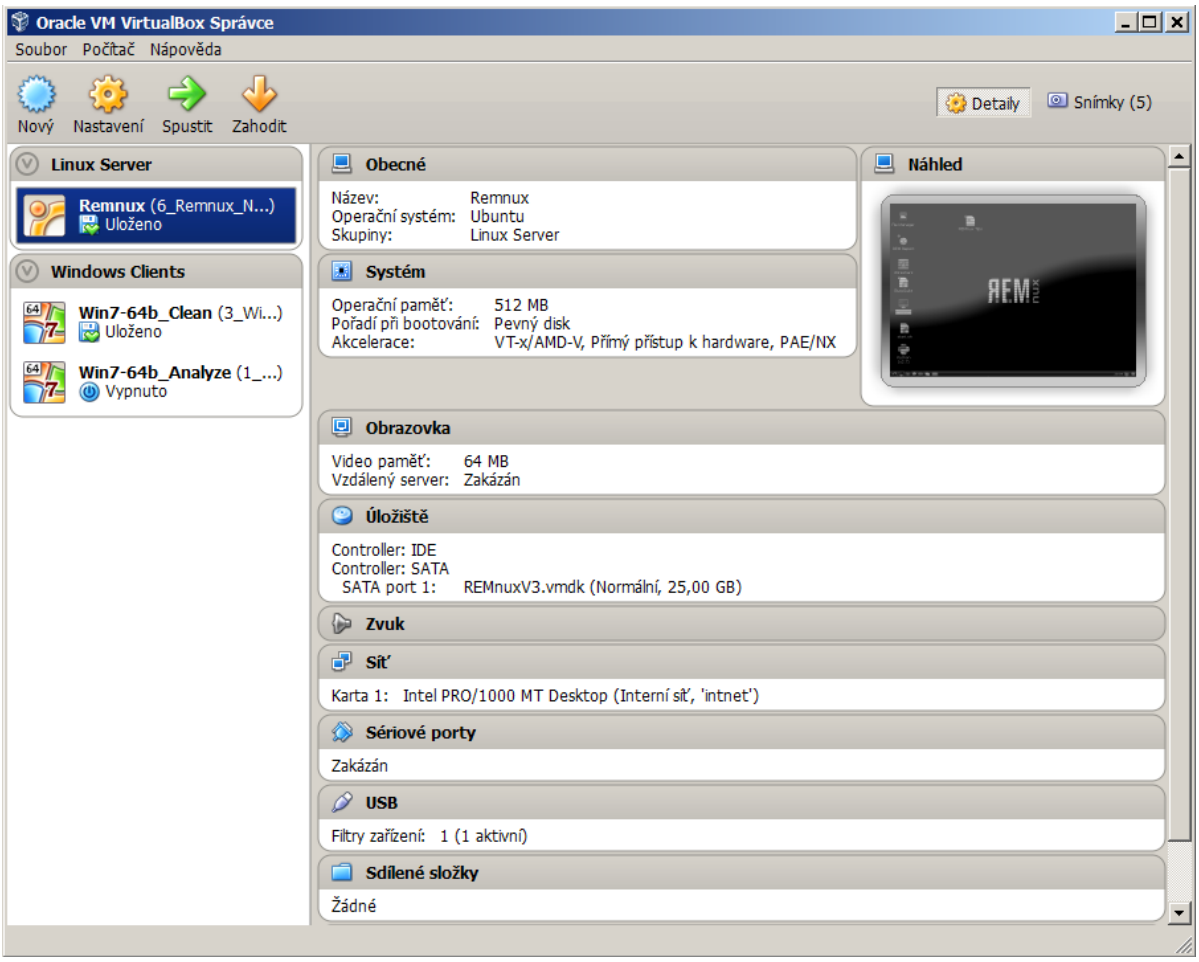
[42] WEBER, Laurent. Dynamic Analysis of Malware. Horts-Götz Institute, Ruhr-University Bochum, 2010. Dostupné z WWW: http://www.docstoc.com/docs/91857229/Dynamic-Analysis-of-Malware

Přílohy

Na přiloženém DVD se nachází vytvořené a v této práci popsané prostředí, které je uloženo v archivu 7zip[19].

Prostředí se skládá z připravených snímků následujících virtuálních strojů:

- virtuální stroj Remnux
- virtuální stroj Win7-64bit_Clean
- virtuální stroj Win7-64bit_Analyze



Obrázek 15: Snímek konfigurace Oracle VM VirtualBox

19# 7zip je otevřený formát s vysokým komprimačním poměrem. Použitá kompresní metoda je LZMA2.