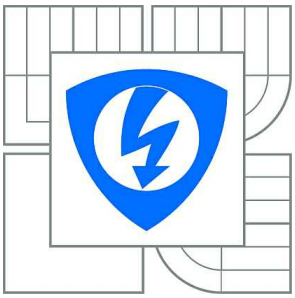
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF TELECOMMUNICATIONS

IMPLEMENTACE NÁSTROJE HONEYPOT PRO SLEDOVÁNÍ A ANALÝZU SÍŤOVÝCH ÚTOKŮ

IMPLEMENTATION OF HONEYPOT TOOL FOR MONITORING AND ANALYSIS OF NETWORK ATTACKS

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

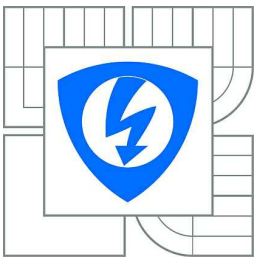
AUTOR PRÁCE — LADISLAV NĚMEČEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. MILAN BARTL

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Ladislav Němeček	***ID:***	110612
Ročník:	3	***Akademický rok:***	2011/2012

NÁZEV TÉMATU:

Implementace nástroje Honeypot pro sledování a analýzu síťových útoků

POKYNY PRO VYPRACOVÁNÍ:

Zvolit vhodný typ Honeypotu s vysokou interakcí pro monitorování a anylýzu aktivit síťových útočníků. Prozkoumat možnosti implementace v rámci sítě VUT a tuto implementaci realizovat. Popsat detailně funkce a možnosti využití daného typu Honeypotu. Provést analýzu útoků provedených na Honeypot ve VUT síti a z analýzy vyvodit závěry.

DOPORUČENÁ LITERATURA:

[1] N. Provos, T. Holz. Virtual Honeypots: From Botnet Tracking to Intrusion Detection. Addison-Wesley, 2007. 440 p. ISBN 978-0321336323.

[2] NoAH - European Network of Affined Honeypots. http://www.fp6-noah.org/.

[3] MLČOCH, T., Zachycení síťových útoků pomocí Honeypotů, bakalářská práce, FIT VUT v Brně, Brno, 2011.

Termín zadání:	6.2.2012	***Termín odevzdání:***	31.5.2012

Vedoucí práce: Ing. Milan Bartl
Konzultanti bakalářské práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

Abstrakt

Náplní této práce je popis a rozdělení škodlivého software. Rozebrány jsou způsoby síťových útoků a ochrana proti nim. Popisuje také, jak útoky detekovat a analyzovat pomocí vhodných nástrojů. Další část práce je věnována tématu honeypotů a možnostmi detekce za pomoci tohoto software. Konkrétně se pak jedná o nástroj Argos. Je popsána jeho instalace, použití a způsob detekce. Dále je v práci popsáno, jak zabezpečit honeypot proti zneužití útočníkem. V neposlední řadě práce také obsahuje výsledky monitorování síťového provozu, útoky na honeypot a popisuje log soubory, kterými Argos interpretuje výsledky detekce útoků.

Klíčová slova

malware, bezpečnost, síťové útoky, analýza a detekce útoků, honeypot, Argos

Abstract

The goal of this thesis is to describe and categorize the malicious software. Thesis deals with the network attacks and the protection against them as well as how to detect and analyze the attack by the eligible tools. The next part of the thesis deals with the honeypot topic and the possibilities of detection using this software, specifically then the Argos tool. The installation, usage, and the methods of detection of the tool are also being described. The next chapter describes how to secure the honeypot against abuse. Last but not least, the thesis also contains the results of the network monitoring, attacks on the honeypot, and describes the log files used by Argos to interpret the results of the attack detection.

Keywords

malware, security, network attacks, analysis and detection of attacks, honeypot, Argos

Bibliografická citace

NĚMEČEK, L. *Implementace nástroje Honeypot pro sledování a analýzu síťových útoků*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 49 s. Vedoucí bakalářské práce Ing. Milan Bartl.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma „*Implementace nástroje Honeypot pro sledování a analýzu síťových útoků*“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne 12. srpna 2012 ...…….

podpis autora

Poděkování

Děkuji vedoucímu práce Ing. Milanu Bartlovi za velmi užitečnou metodickou pomoc a cenné rady při zpracování bakalářské práce.

V Brně dne 12. srpna 2012 ...…….

podpis autora

Obsah

Úvod

Moderní technologie, zejména pak počítačové systémy a vše s nimi spojené, se staly nedílnou součástí našich životů. Bez počítače a internetu je dnes pro většinu z nás nepředstavitelné a nemožné existovat. Slouží nám jako zdroj zábavy, vzdělávání, prostředek pro komunikaci, ale i jako nástroj pro výzkum, práci a její usnadnění. Díky tomu se dnes v síti a v počítačích nachází většina dat – některá pro mnoho lidí nedůležitá, jiná naopak užitečná a pak také data důvěrná, osobní a tajná. S tím se ruku v ruce nese obrovské riziko jejich zneužití či krádeže.

S rozvojem počítačových technologií tak na sebe dlouho nenechal čekat do té doby neznámý druh kriminální činnosti – *kyberkriminalita*. Jedná se především o útoky ze sítě za použití různých typů *malware*. První pokusy o útok nebyly vedeny s cílem data odcizit nebo je dále využít ve prospěch útočníků. Většina malware byla vyvíjena za účelem data v počítačových úložištích buď vymazat nebo je jiným způsobem poškodit. S přibývajícím množstvím systémů, jejich uživatelů, a tím i důležitých a tajných dat, začaly být moderní především útoky s cílem tato data využít. Převážná většina počítačů má dnes přístup buď přímo do internetu a nebo do lokální sítě a s rostoucím počtem takových stanic se pak i zvyšuje riziko napadení ze sítě.

Právě z tohoto důvodu je třeba se těmto útokům účinně bránit, detekovat je a předcházet jim. Existuje mnoho způsobů, jak s pokusy o napadení bojovat, počínaje prevencí a konče aktivní ochranou. Prevence není zdaleka spolehlivá metoda a navíc převážná většina uživatelů není tak technicky vyspělá, aby dokázala určit, co je a co není nebezpečné. Skupina metod aktivní ochrany je založena především na detekci a obraně před kompromitací systému.

V dnešní době lze využít mnoho druhů softwarových i hardwarových řešení k realizaci účinné obrany. Ta však také není stoprocentní. Je třeba neustálý vývoj, aby byla zaručena bezpečnost a ochrana před novými a novými pokusy o napadení systému. S tím je současně potřeba neustále analyzovat nové vzorky a typy malware, a udržovat tak databáze těchto škodlivých kódů aktuální. Jedním ze způsobů, jak nebezpečné aktivity detekovat a analyzovat, je technologie *honeypotů*. Tato práce bude mít za cíl popsat jejich technologii a způsoby, kterými zachytávají známé, ale i nové pokusy o infiltraci a zneužití bezpečnostních děr.

1.1 Obsah a cíle práce

Obsahem této práce je téma analýzy a detekce útoků za pomoci honeypotů. Je zaměřena na popis jejich technologií a způsoby použití. Věnuje se také samotnému tématu škodlivého kódu, útokům ze sítě a jejich detekce. Za cíl by pak měla mít nastudování principů funkce honeypotů, jejich používání a správu. Budou také porovnány dostupné honeypoty. Na základě těchto testů bude zvolen nejvhodnější nástroj pro nasazení do experimentálního provozu a dále pak do sítě VUT. Tento původní plán však nebyl z důvodů potencionální hrozby napadení povolen. Bylo tedy nutné využít náhradního řešení, a to sestavení samostatného stroje a jeho umístění do soukromé sítě. Zde bude vystaven pokusům o napadení z veřejné sítě a také sadě testovacích útoků.

Kapitola 2 pojednává o tom, co je to škodlivý software a jakým způsobem a rychlostí se v dnešní době šíří. Bude popsáno jeho rozdělení do kategorií a také uvedeny některé ukázky známých zástupců škodlivého software. V neposlední řadě tato kapitola také přiblíží současné trendy a budoucí vývoj.

Projevy škodlivého kódu a různé typy útoků za pomocí malware pak popisuje kapitola 3. Bude zde popsána také prevence a ochrana před útočníky.

Kapitola 4 popisuje hlavní téma této práce, a tím je problematika samotných honeypotů. Bude vysvětlena funkce a princip, jakým útoky detekují, dále pak jejich rozdělení do jednotlivých skupin a jejich použití. Bude zde také popsán postup testování vybraných zástupců.

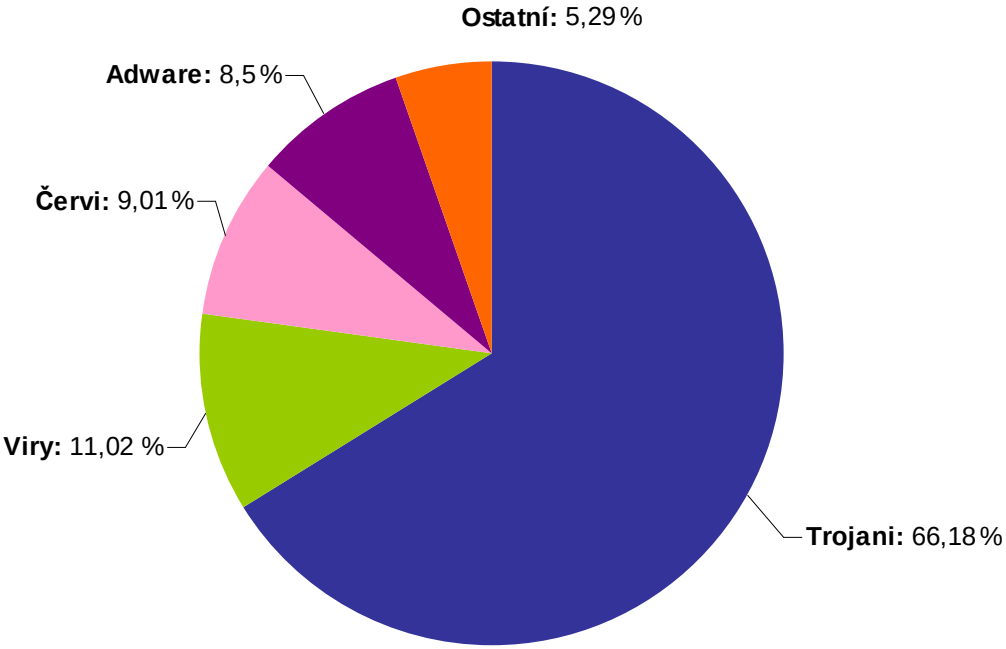
Implementaci honeypotu Argos popisuje kapitola 5. Bude zde popsán postup instalace, síťového nastavení a spuštění. Další část se pak věnuje zabezpečení honeypotu proti napadení a zneužití.

Poslední kapitola 6 obsahuje postup a způsob, jakým byly provedeny útoky na běžící honeypot. Součástí je také popis struktury log souborů generovaných honeypotem. Další část kapitoly se věnuje analýze získaných dat. Výstupní log soubor je podroben důkladnějšímu zkoumání a jsou také vytvořeny statistiky provozu honeypotu.

2 Škodlivý software

Škodlivý software je v anglickém jazyce nazýván malware (zkráceno ze sousloví malicious software). Jde všeobecně o jakýkoliv kód nebo software, který má za cíl infiltrovat se do počítačového systému, a to bez vědomí uživatele. Tento neoprávněný průnik je veden za účelem poškození nebo krádeže uživatelských dat, získání přístupu k systému a následně pak jeho použití k další škodlivé činnosti.

V dnešní době lze většinu škodlivého kódu připsat na účet čínských útočníků a převážné procento útoků je směřováno z počítačů umístěných v některé z asijských zemí, jako jsou Čína, Thajsko, Taiwan, ale i Rusko a Ukrajina. V roce 2010 byla vytvořena a rozšířena více než třetina všech virů. To znamená, že za dvanáct měsíců bylo vytvořeno 34 % veškerého malware, který existuje a který byl vypuštěn do světa [1]. Z analýzy v roce 2011 vyplývá, že bylo identifikováno přes 26 milionů souborů jako nový malware. Tomuto číslu odpovídá asi 73000 nových vzorků za den, což představuje největší počet, jaký byl zaznamenán. Největší podíl na tomto čísle mají stále *trojské koně.* Za zmínku také stojí i to, že 11,6 % z celkového počtu všech souborů zaujímají programy tvářící se jako antivirový software [2].



Obr. 2.1: Zastoupení jednotlivých typů malware v analyzovaném vzorku. [2]

Pojem malware lze rozdělit do několika kategorií. Ty jsou však velmi úzce provázány, a proto je někdy těžké kód zařadit do určité skupiny. Pod souhrnným označením malware si můžeme představit *počítačové viry, červy, trojské koně, rootkity, spyware* a další formy.

2.1 Viry

Pod pojmem virus je často mylně označován jakýkoliv typ škodlivého kódu. Název počítačový virus vznikl díky jeho vlastnostem podobajícím se biologickému viru. Jedná se o program schopný replikovat sebe sama. K tomu však potřebuje hostitele v podobě spustitelného souboru, skriptu, makra nebo jiných specifických programů. Jakmile tedy dojde ke spuštění hostitelského souboru, dojde i ke spuštění počítačového viru, který je k němu připojen a chová se jako jeho součást. Více propracované a složitější viry disponují sofistikovanějšími metodami, jak se bránit antivirovým programům a mohou déle „přežívat" v systému. Za jeden z nejdestruktivnějších virů lze považovat *Win32.CIH*, také známý jako *Černobyl*. Šířil se na strojích běžících pod Windows 95, 98 a ME, kde působil škodu tím, že přepsal paměť Flash-BIOS[1] a současně s tím znehodnotil i data na pevných discích [3].

2.2 Červi (Worms)

Svými vlastnostmi sebe-replikace se velmi podobají počítačovým virům. K této činnosti však nepotřebují hostitelské soubory. Šíří se ve formě síťových paketů a systém napadají prostřednictvím jeho bezpečnostních děr. To má pak za následek šíření dalších paketů z infiltrovaného systému na další počítače v síti. Výsledkem toho pak mohou být škody ve formě zahlcení přenosové sítě.

Jedním z nejznámějších příkladů byl červ *Blaster,* označovaný také jako *Lovsan* nebo *Lovesan*. Ten se začal šířit 11. 8. 2003 a jednalo se pravděpodobně o nejrozsáhlejší infiltraci v historii Internetu [4]. Červ napadal stanice, které běžely na operačním systému Windows 2000 a Windows XP. Využíval k tomu bezpečnostní díru v *RPC*[2] rozhraní. Více lze o této bezpečnostní díře najít v MS Bulletinu *MS03-26*[3]

---

[1] Basic Input-Output Systém – slouží pro inicializaci a konfiguraci připojeného hardware a k zavedení operačního systému.

[2] Remote Procedure Call – protokol umožňující programu vzdálené volání procedur.

[3] http://technet.microsoft.com/en-us/security/bulletin/ms03-026

2.3 Trojské koně (Trojans)

Mají formu spustitelného souboru a na rozdíl od virů nemají schopnost sebereplikace a infekce nových souborů. Původní definice popisuje trojského koně jako program, který navenek vypadá užitečný, obsah souboru však skrývá škodlivý kód samotného trojanu. Tomuto popisu odpovídají tzv. *destruktivní trojani*. Výsledkem jejich spuštění bývá ztráta nebo poškození dat. Dalším typem jsou *Password-stealing trojani*, tzv. *Keyloggery*. Ty sledují jednotlivé stisky kláves a následně je pak odesílají autorovi. Výsledkem pak může být odcizení důvěrných osobních dat a údajů.

Patří sem i *Backdoors*, jejichž úkolem je vytvořit v napadeném stroji zadní vrátka a poté umožnit útočníkovi vzdálenou správu. Dále *Droppery* a *TrojanDownloadery*, což jsou vlastně přenašeči dalšího škodlivého kódu. TrojanDownloader však nemá škodlivý kód v sobě, ale stahuje jej z nadefinovaných *URL*[4] adres. Lze sem také zařadit *Proxy trojany*, které napadený systém promění na proxy server a slouží např. k rozesílání *spamu* (nevyžádané pošty).

Dnes lze mezi nejznámější trojany považovat falešné antivirové programy. 40 % z nich bylo vytvořeno v roce 2010. Jedná se o velice účinný typ škodlivého kódu, neboť díky sofistikovanému návrhu, věrohodnému vzhledu a důvěryhodnosti spousta lidí podlehne a „spadne do pasti“ útočníků. Principem jejich výdělečné činnosti je přesvědčení uživatele ke koupi plné licence programu. Tímto způsobem jsou schopni výrobci vydělávat neuvěřitelných 34 milionů dolarů měsíčně. Žebříček těchto programů vedou programy jako *SystemGuard2009* (obrázek 2.2), *MSAntiSpyware2009* nebo *MalwareDoctor* [1].

2.4 Rootkity

Tyto programy mají za úkol především získat neomezený přístup k počítači a následně maskovat útočníkovu činnost v systému za pomoci skrývání běžících procesů, modifikací operačního systému, registrů nebo maskováním zvýšené síťové aktivity. V podstatě tedy nepředstavují přímé nebezpečí, ale ve spojení s některým druhem malware se už jedná o nebezpečný software. Instalace rootkitu tak bývá prvním z kroků při napadení stroje.

Mezi představitele této skupiny můžeme uvést např. *Hacker Defender* nebo *FU*. Více o tomto tématu pojednává [5].

[4] Uniform Resource Locator – definuje adresu serveru a protokol, kterým je možno ke zdroji přistupovat.

Obr. 2.2: Falešný antivirus SystemGuard2009 byl původcem nejvíce infekcí v roce 2010 [zdroj: http://infogeeker.wordpress.com/2010/12/04/el-rogueware-la-amenaza-mas-extendida-de-2010]

2.5 Spyware

Jedná se o software, jehož úkolem je odesílání statistických informací bez vědomí uživatele. Mezi tyto informace patří například seznam navštívených serverů, údaje o IP adresách, seznamy nainstalovaných programů, ale také údaje o přístupových heslech, informace o bankovních transakcích, účtech, platebních kartách atd. Velké množství spyware je šířeno prostřednictvím *shareware softwaru*. Výsledky tohoto sběru informací pak může využívat i *adware*, což je program, který obtěžuje uživatele různými reklamami nebo pop-up okny. Hlavní rozdíl mezi spyware a adware je ten, že spyware se instaluje bez vědomí uživatele.

2.6 Očekávané bezpečnostní hrozby

V současné době útočníci pracují sofistikovaněji a lze dále očekávat rostoucí počet útoků a bezpečnostních hrozeb. Jak již bylo zmíněno na začátku kapitoly, za poslední rok bylo vytvořeno nejvíce malware a největší podíl na tom mají trojské koně.

Hrozbou byly i útoky na sociálních sítích nebo kyberkriminalita a s ní spojené útoky na bankovní instituce a herní společnosti. Další skupinu představuje nevyžádaná pošta. Nejedná se o klasický spam, ale spíše o rozesílání velkého množství nevyžádaných obchodních

nabídek. Tato situace vzniká nejčastěji poskytnutím emailové adresy samotným uživatelem, zejména při nakupování přes internet. Zprávy jsou velmi často jazykově lokalizovány a velmi špatně se detekují jako spam. S tímto typem útoků pak souvisí i hrozba emailů obsahujících nebezpečnou přílohu. Typicky se v dnešní době jedná o zneužití za pomoci souboru PDF.

Co se tedy dá v budoucnu očekávat? V oblasti sociálních sítí to bude zejména využívání technik *sociálního inženýrství* (bude vysvětleno v kapitole 3). Útočníci se tak budou zaměřovat na hlavní témata dnešní doby, aby dále shromažďovali osobní údaje uživatelů.

Počet nových vzorků malware roste rok od roku exponenciálně. Tento trend bude tedy dle předpokladů pokračovat i nadále. Skutečnost, že tři ze čtyř nových vzorků byly trojské koně, stále nasvědčuje jejich převaze v použití.

V současné době, kdy naprostá většina informací je v digitální podobě, a to i ty nejdůvěrnější, vládne *kyber-válka*, nebo spíše *kyber-špionáž*. Útočníci s potřebnými znalostmi budou stále častěji napadat světové společnosti a vládní agentury s cílem ukrást tajné informace či narušit přístup k nejstřeženějším tajemstvím.

S očekávaným uvolněním Windows 8 to pak bude vývoj škodlivého kódu pro zařízení, pracující na tomto operačním systému a hledání bezpečnostních děr v něm. Útoky se jistě nebudou vyhýbat ani ostatním platformám v čele s OS Mac. Další skupinou, která stojí za zmínku, jsou útoky na mobilní telefony a tablety [2].

3 Síťové útoky a jejich detekce

Do kategorie útoků ze sítě můžeme zařadit i využití „nepočítačových“ technik, které se nazývají sociální inženýrství. Jde vlastně o psychologické působení a přesvědčování uživatele počítače k poskytnutí informací potřebných k následnému přístupu do systému nebo jejich zneužití v další podvodné činnosti. Tyto průniky jsou jedny z nejhorších. Útočník se nesnaží prolomit žádná silná hesla ani zabezpečení, ale využívá chybu lidského faktoru.

Do této kategorie patří *hoax*. Jedná se o šíření poplašných a zbytečných zpráv. Výsledkem je pak zmiňované ovlivňování uživatelů a po stránce technické například nadměrné vytěžování linek a serverů.

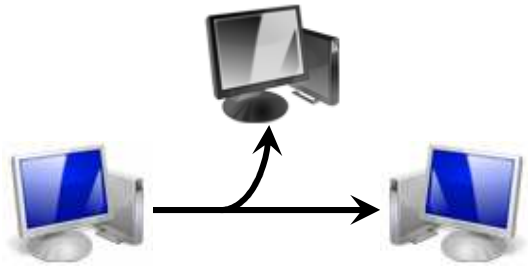
Další technikou je *phishing*, což je metoda, při které se útočník snaží získat důvěrné údaje pomocí falešných formulářů, webových stránek a podobně. Více o tématu sociálního inženýrství lze nalézt v [6].

Pro tuto práci bude více zajímavější kategorie využívající spíše technických možností, chyb a bezpečnostních děr v systémech a komunikačních protokolech.

Tyto síťové útoky lze podle [7] rozdělit do dvou základních kategorií, a to na *pasivní* a *aktivní*. Hlavním rozdílem je, zda útočník do komunikace zasahuje nebo ne. Pasivní útoky mají za cíl získání informací a dat pomocí monitorování komunikace, avšak nijak do ní ani do systémových prostředků nezasahují. Aktivní se oproti tomu přímo pokouší o zásah do síťového spojení a mají vliv na systém nebo jeho funkci. Aktivní útoky představují opačné vlastnosti pasivních. Zatímco pasivní je obtížně detekovat, aktivním útokům je na druhou stranu dost obtížné zabránit, neboť existuje široká škála možných síťových a softwarových zranitelností.

3.1 Pasivní útoky

Jak již bylo zmíněno, pasivní útoky mají v povaze pouze naslouchat a sledovat provoz. Vlivem toho, že do komunikace nijak nezasahují, je obvykle přenos zpráv zdánlivě v pořádku a odesílatel ani příjemce si není vědom toho, že jsou odposlouchávání třetí stranou. Princip pasivních útoků je znázorněn na obrázku 3.1. Úspěchům těchto útoků lze zabránit obvykle šifrováním přenosu. Skupinu pasivních útoků můžeme rozdělit na dva typy: *sběr informací (release of message contents)* a na *analýzu nasbíraných dat (traffic analysis).*



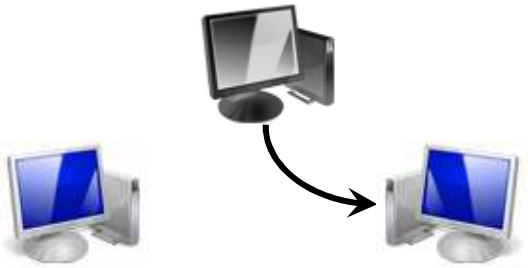
Obr. 3.1: Princip pasivního útoku

3.2 Aktivní útoky

Skupina aktivních útoků je charakterizována některou z úprav toku dat nebo vytvořením nových proudů dat. Úloha útočníka je zde nazývána *Man-In-The-Middle*, tedy prostředníka v komunikaci. Tuto skupinu dělíme do čtyř kategorií:

3.2.1 Maškaráda (Masquerade)

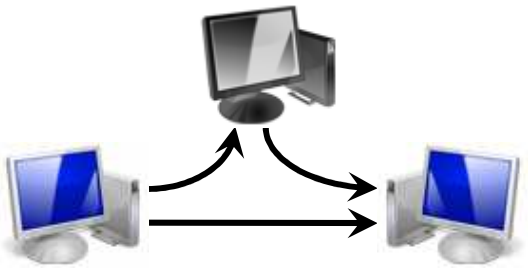
Útočník se vydává za někoho jiného a využívá této metody k ověření přístupu k systému. Útok typu maškaráda je znázorněn na obrázku 3.2.



Obr. 3.2: Útok typu Maškaráda

3.2.2 Zachycení (Replay)

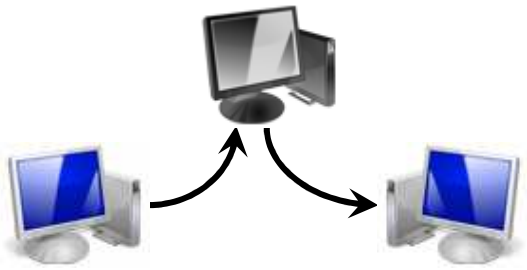
Tento útok zahrnuje použití pasivní techniky zachycení dat a následně pak jejich použití k neoprávněnému přístupu. Data se však nijak nemění, útočník je pouze příjme a zase odešle. Princip těchto útoků je na obrázku 3.3.



Obr. 3.3: Útok typu Zachycení

3.2.3 Modifikace zpráv (Modification of messages)

Tento útok vychází z předchozího typu. Útočník, stejně jako při zachycení, využívá odchytávání komunikace, ale část legitimní zprávy modifikuje nebo posílá v novém uspořádání či zpožděně. Jako příklad může sloužit zjednodušená zpráva: „`Povolit čtení souboru hesla.txt pro uživatele USER`“. Ta je pak změněna na tvar: „`Povolit čtení souboru hesla.txt pro uživatele ÚTOČNÍK`“.

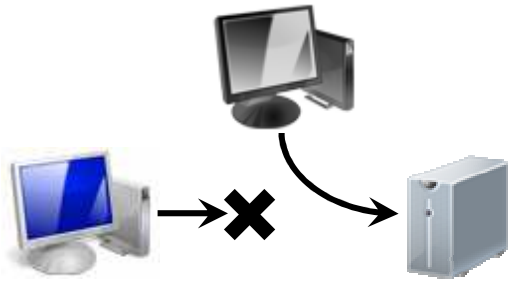


Obr. 3.4: Útok typu Modifikace zpráv

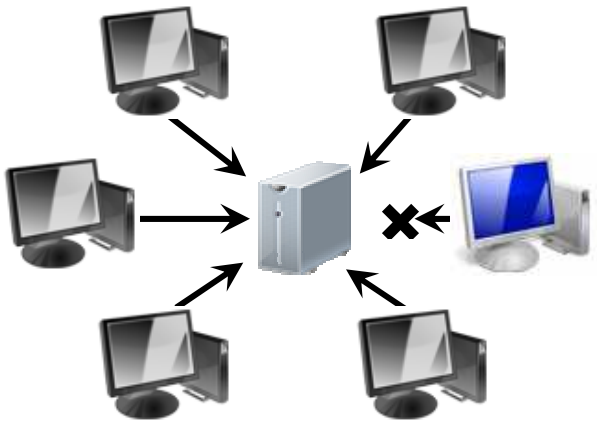
3.2.4 DoS útok (Denial of Service)

Jedná se o útok, který zabraňuje nebo omezuje běžné použití a řízení stanice. Cílem tak tedy není získání důvěrných dat, ale pouze přerušení komunikačních kanálů. V principu se buď jedná o zaslání „správně“ upraveného paketu, který způsobí zacyklení softwarového vybavení nebo zhroucení celého serveru, nebo o útoky záplavou paketů. Výsledkem je vyčerpání zdrojů serveru, který pak není schopen obsloužit ostatní uživatele. Obrana proti DoS je identifikace zdroje útoku a jeho následná izolace od serveru.

Vážnější hrozbu pak představuje *DDoS útok (Distributed DoS)*, kdy je pro záplavový útok využito mnoho strojů (*zombies*). Jako zombies slouží počítače v internetu infikované v minulosti pomocí některého z malware. Útočník pak může hromadnou komunikací přikázat takovýmto zombies vyslat záplavový útok na určený server. Z popsaného principu je zřejmé, že způsob obrany proti DoS útokům zde není příliš účinný a identifikace zdroje útoku není možná. V posledních letech jsou nástroje a metody útoků stále důmyslnější a efektivnější. Je tak obtížnější vysledovat skutečné útočníky a obranné technologie nejsou schopny odolat rozsáhlým útokům.



Obr. 3.5: Útok typu DoS



Obr. 3.6: Útok typu DDoS

3.3 Detekce a analýza útoků

Důležitým krokem při obraně proti malware je prevence. Mnohdy však ani technické vybavení nepomůže, ať už se jedná o hardware nebo software prvky v síti, jako je například *firewall*[5]. Jejich funkční činnost může být na útoky vedené proti nim příliš slabá nebo je mnohdy podkopávají chyby lidského faktoru. Když už se útočník dostane za hranice preventivní činnosti, je na řadě činnost aktivní. Spočívá v testování souborů, detekci a popřípadě analýze provedeného útoku a následném podnikání patřičných kroků k zamezení neautorizovaných kroků. To, zda testovaný soubor obsahuje škodlivý kód, lze zjistit různými metodami [8].

[5] Síťové zařízení definující pravidla pro komunikaci v síti mezi zdrojovou a cílovou adresou.

3.3.1 Detekce na základě otisků

Metodu rozpoznávání škodlivého kódu na základě otisků (*signatures)* využívají hlavně antivirové programy. Je založena na principu porovnávání specifických otisků. Z každého nově získaného škodlivého souboru se vytvoří specifický otisk, který je uložen do databáze. Pokud je pak při testování nalezena shoda, je soubor považován za malware. Tato technika je však z principu funkce závislá právě na databázi získaných vzorků. Nové útoky, nazývané *zero-days* (vysvětleno dále v kapitole 4), jsou vůči tomuto testování imunní až do doby, než je jejich signatura přidána do databáze antimalwarových programů. I tak je ovšem díky kompresi, kryptografickým metodám nebo polymorfním kodérům možno otisk známého malware pozměnit a ten se znovu stává neznámým.

3.3.2 Analýza souborů

Existují dvě metody, které mohou být použity k provedení analýzy souboru, a tím pak i k pochopení, co vlastně kód provádí.

Jedná se především o *statickou analýzu*, kdy se testovaný soubor vůbec nespouští, ale jeho kód se převede do jazyka symbolických instrukcí. Následně se pak zkoumají potenciální škodlivé části. To ale vyžaduje znalosti především *assembleru*[6], techniky ladění software a metod šifrování. V neposlední řadě to v některých zemích může být problematické díky příliš striktním zákonům týkajících se software.

Druhým typem je *behaviorální analýza.* Není tak náročná na technické znalosti jako předchozí typ, je však nebezpečnější. Soubor se nijak zpětně nerozkládá, ale je spuštěn a jeho aktivity jsou sledovány. Sleduje se především podezřelá síťová aktivita, zápis do systémových registrů nebo na disk a další anomálie naznačující škodlivou činnost. Spuštění samozřejmě neprobíhá na hostitelském systému, ale v tzv. *sandboxu*. Jedná se o virtuální monitorované izolované prostředí, v němž je kód spuštěn.

Na principu analýzy souborů fungují systémy nazvané *IDS (Intrusion Detection System)* a *IPS (Intrusion Prevention System).* Jedná se o nástroje detekující podezřelou aktivitu v síti nebo v systému. IPS se pak kromě sledování aktivit v síti (systémy IDS) snaží na útoky i reagovat. Většinou se používají systémy *IDPS* kombinující funkce obou zmíněných [9].

---

[6] Jazyk symbolických adres.

4 Honeypoty

Jednou z dalších možností, jak síťové útoky předvídat a bránit se jim, je technologie *honeypotů*. Jejich činností je sledování a analýza veškeré komunikace směřující k nim. Honeypot, v překladu „hrnec s medem," je podle [10] systém, který využívá „*jail-technology*" přitahující potenciální útočníky. Lze si je představit jako stroj anebo soubor programů, které v síti nevykazují žádnou aktivitu ani funkci. Pouze emulují záměrně nezabezpečenou stanici nebo i celou síť těchto stanic a vyvolávají v útočníkovi pocit, že se pokouší infiltrovat do systému obsahujícího užitečná data. Z tohoto popisu funkce lze pak říci, že veškerá komunikace s ním je považována za neautorizovanou, a tím i škodlivou.

Celá tato komunikace je různými nástroji sledována a následně analyzována. To pak napomáhá v budoucnosti k úspěšné obraně proti těmto útokům.

Dalším cílem této analýzy je detekce zero-day útoků. Jde o útok s cílem využít zranitelnosti, která není ještě obecně známa nebo nebyla doposud opravena softwarovou záplatou nebo aktualizací [11].

Jeden způsob, jak rozdělit jednotlivé honeypoty do skupin podle funkce, je dělení podle účelu použití [12].

4.1 Fyzické a virtuální honeypoty

Fyzické honeypoty jsou realizovány jako hardwarový nástroj. Jde o stanici v síti určenou pouze pro potřeby a funkci honeypotu. Tento způsob je náročný z hlediska finanční investice a pro emulaci velkého adresového prostoru jsou nepoužitelné.

To naopak umožňují *virtuální honeypoty*. Díky *virtualizaci* lze provozovat souběžně několik operačních systémů na jednom fyzickém stroji. Za použití jednoho nebo několika strojů pak můžeme provozovat v síti velké množství honeypotů.

4.2 Serverové a klientské honeypoty

Princip *serverových honeypotů* představuje dosud popisovaný systém. Jde o stanici bez jakékoliv funkce, která v síti „čeká" na to, až na ni útočník zaútočí. Tyto útoky nebo pokusy o komunikaci jsou pak sledovány a analyzovány.

Opakem jsou *klientské honeypoty*. Ty simulují běžného uživatele koncové stanice. Aktivním způsobem pak procházejí internetové stránky a vyhledávají pochybné a infiltrované servery.

4.3 Bezdrátové honeypoty

Tato skupina je mírně odlišná od ostatních. Jejich úkolem je chránit bezdrátové sítě, a to formou vytváření velkého množství fiktivních bezdrátových přístupových bodů bez anebo se slabým zabezpečením. To pak dává útočníkovi možnost pokusit se o přístup do takové bezdrátové sítě, například za účelem neoprávněného využívání sdíleného připojení k internetu.

Dalším způsobem je dělení podle interakce:

4.4 Honeypoty s nízkou mírou interakce

Tyto honeypoty jsou schopny emulovat určité funkce, programy nebo služby operačního systému. Tato emulace je však do jisté míry omezena. Jde o pár předem daných odpovědí na vybrané útočníkovy akce. Na jednu stranu jde o jednoduché systémy z hlediska implementace a údržby, na druhou stranu však neposkytují příliš velké množství informací o útocích (zejména se jedná o statistiky).

Nejznámější z této skupiny je asi *Honeyd.*[7] Jedná se o honeypot schopný provozovat na volných IP adresách virtuální hosty a emulovat na nich řadu služeb a operačních systémů. Dále sem lze zařadit například *HoneyC*[8] , *Tiny Honeypot*[9] nebo *Dionaea*[10]

4.5 Honeypoty s vysokou mírou interakce

Druhým typem jsou *honeypoty s vysokou mírou interakce*. Ty už jsou na rozdíl od předchozí kategorie schopny emulovat celé systémy s velkým množstvím služeb a aplikací. Možnost nasadit mnohem více nástrojů umožňuje zjistit o útočníkovi více informací a jeho úmysly. Za jejich nevýhody lze považovat větší náročnost na implementaci a správu. Nevýhodou je také případné ovládnutí celého systému útočníkem a jeho následné využití k další škodlivé činnosti. Kategorie honeypotů s vysokou mírou interakce je pro další vývoj této práce nejzásadnější a nejdůležitější.

---

[7] http://www.honeyd.org/

[8] https://projects.honeynet.org/honeyc

[9] http://freecode.com/projects/thp

[10] http://dionaea.carnivore.it/

V dnešní době jsou v této kategorii dostupné pouze tři nástroje. Všechny je možné získat z jejich domovských webových serverů. Ani u jednoho zástupce není zmínka o ukončení aktivního vývoje či podpory, avšak data posledních aktualizací serverů, aktivit technické podpory a vydání nových verzí ukazují na dlouhodobou nečinnost autorů všech tří zástupců. Nic z toho ale nebrání použití stále dostupných starších verzí.

4.5.1 Capture-HPC[11]

Jedná se o zástupce klientských honeypotů s vysokou mírou interakce. Je založen na architektuře *klient/server*. To dovoluje spravovat data z více klientů na jednom centrálním serveru. Umožňuje také sledování souborového systému, registrů a procesů systému na úrovni jádra. Nástroj je spouštěn ve virtualizačním prostředí, kde je schopný emulovat většinu *HTTP* prohlížečů, ale i například nástroje *MS Office*, *Adobe Acrobat Reader* a mnoho multimediálních přehrávačů. Dále je schopen automaticky shromažďovat malware zachycený v síti a na klientských stanicích. Samozřejmostí je pak vytváření log souborů.

Princip klientských honeypotů, kdy tyto nástroje aktivně prohledávají servery a získávají tak informace, není pro tuto práci předmětem zájmu. Nebude proto při následné implementaci honeypotu dále zařazen do seznamu vhodných nástrojů, ani nebudou testovány jeho funkce.

4.5.2 Honey@Home[12]

Jde o zástupce klasického serverového honeypotu. I ten je založen na architektuře klient/server. Získaná data aplikace zasílá na centrální server, který je pod kontrolou organizace *NoAH*[13].

V systému je Honey@Home spuštěn jako proces na pozadí, a nezatěžuje tak zbytečně prostředky hostitelského stroje. Je schopen v nevyužitém adresovém prostoru provozovat množství klientů. Zde pak naslouchá na portech služeb jako *TELNET, FTP, SSH, POP3, HTTP, SMTP* atd. Seznam služeb operačního systému Windows a jejich síťových portů, které mohou být nejpravděpodobněji napadeny, je uveden v tabulce 4.1. Jde pouze o malý výběr

[11] https://projects.honeynet.org/capture-hpc

[12] http://www.honeyathome.org/

[13] http://www.fp6-noah.org/

z velkého seznamu potencionálních cílů. Další služby a jejich příslušné porty lze nalézt například v [13].

Tab. 4.1: Nejpravděpodobněji napadené služby systému Windows [13]

Port	*Služba*	*Port*	*Služba*
20,21	FTP (Přenos souborů)	137-139	NetBIOS (sdílení prostředků)
22	SSH (Zabezpečená komunikace)	443	HTTPS
23	Telnet (Vzdálený přístup)	445	Microsoft DS (sdílení souborů)
25	SMTP (odchozí emailové zprávy)	3128	HTTP Proxy
53	DNS (překlad ip adres)	3306	MySQL (databáze)
80	HTTP	3389	RDP (vzdálená správa)
110	POP3 (příchozí emailové zprávy)	5900	VNC (vzdálená správa)
135	Microsoft RPC	8080	HTTP Proxy

Klientská část je nastavena tak, aby provozovala služby na specifických portech dostupných pro útočníky. Výsledky detekce jsou zasílány na centrální server, který je dále zpracovává. Výhodou toho je samotné oddělení klientů, jež mohou být napadeni, a serveru, který s nimi komunikuje přes zabezpečené připojení. K dispozici je verze pro platformu Windows i pro Linux.

V následujícím textu bude popsána instalace pod oba operační systémy, a bude tím otestována vhodnost pro další použití tohoto nástroje. Nejprve tedy instalace pod systémem Windows. Z domovských stránek je možno stáhnout honeypot ve formě instalátoru *msi*. První podmínkou pro korektní instalaci je nainstalovaný *.NET framework 2.0*. Následně pak instalace probíhá bez problémů. Další podmínkou pro kompletní dokončení instalace jak pod Windows, tak i pod Linux je registrace. Toto je možno provést na domovských stránkách s použitím validní emailové adresy, na níž je po registraci odeslán soubor *user.key*. Jedná se o textový soubor obsahující 32-znakový klíč. Je nutné jej zkopírovat do kořenového adresáře aplikace před spuštěním honeypotu. Instalace pod operačním systémem Linux vyžaduje stažení archívu *.tar.gz.* Ten obsahuje zkompilovanou aplikaci a dále pak konfigurační soubor.

```
z0scp9ctn63d1owahsxm8m0algm5xxfw
```

Výpis kódu 4.1: Obsah souboru user.key

Nejprve bude popsán provoz a konfigurace pod operačním systémem Linux. Všechna nastavení jsou prováděna pomocí konfiguračního souboru `honeyathome.config`.

Jako první je v něm uvedena adresa serveru, na který jsou odesílána logovaná data a dále z něj probíhají aktualizace. To vše je realizováno přes zabezpečené spojení. Dalším v pořadí je pak určení a nastavení síťového adaptéru. Honeypotu je možno nastavit IP adresu buď statickou, anebo ji lze dynamicky získávat od *DHCP serveru*. Lze nastavit i *MAC adresu*. Posledním parametrem je seznam portů, na kterých má honeypot naslouchat.

Spouštění aplikace je nutné provádět s právy uživatele `root.` Po spuštění se aplikace připojí k centrálnímu serveru, zkontroluje dostupné aktualizace a provede autentizaci za pomoci přiděleného klíče. Výpis kódu 4.2 obsahuje zkrácený výstup terminálu po spuštění.

Výpis kódu 4.2: Zkrácený výstup programu Honey@Home po spuštění

```
$ sudo ./home
[Thu Jan 19 18:54:53 2012] check_for_update_():  Done checking
                           Versions! Updates_Exist: 0
[Thu Jan 19 18:54:53 2012] wait_for_update(): Started! Will Check
                           every 360 seconds.
[Thu Jan 19 18:54:53 2012] Initializing Honey@Home. Please Wait...
[Thu Jan 19 18:54:53 2012] main(): *** NEW DEBUG/LOG SESSION ***
[Thu Jan 19 18:54:53 2012] #STATUS# 1:Init State
[Thu Jan 19 18:54:53 2012] wait_for_update(): Checking for Update
                           NOW!
[Thu Jan 19 18:54:53 2012] get_default_interface_name(): Possible
                           Default IF 'eth1'
[Thu Jan 19 18:54:53 2012] loadDefaults(): Local Interface set to
                           'eth1'
[Thu Jan 19 18:54:53 2012] loadDefaults(): Server Name set to
                           'gidbgs7cw6thidvi.onion'
[Thu Jan 19 18:54:53 2012] loadDefaults(): Server Port set to '80'
[Thu Jan 19 18:54:53 2012] parseConfig(): Parsing Configuration...
[Thu Jan 19 18:54:53 2012] parseConfig(): *SSL Server Hostname
                           set to: 139.91.130.199
[Thu Jan 19 18:54:53 2012] parseConfig(): *Static Address (for
                           Monitoring) set to: 192.168.88.60
[Thu Jan 19 18:54:53 2012] parseConfig(): *Static MAC Address set
                           to: '00-01-02-03-04-05'
[Thu Jan 19 18:54:53 2012] parseConfig(): *Monitoring Port: 21
[Thu Jan 19 18:54:53 2012] parseConfig(): *Monitoring Port: 22
[Thu Jan 19 18:54:53 2012] parseConfig(): *Monitoring Port: 80
[Thu Jan 19 18:54:53 2012] parseConfig(): Parsing Configuration
                           Complete!
```

Z výpisu v terminálu je patrné nastavení všech parametrů. Tedy IP adresa *192.168.88.60*, MAC adresa *00:01:02:03:04:05* a otevřené porty *21, 22, 80*. Test za pomoci

scanneru portů však nedetekuje ani jeden ze zmiňovaných portů jako otevřený a samotný honeypot také nevykazuje žádný detekovaný přístup na své porty.

Konfigurace a provoz pod systémem Windows je na rozdíl od Linux verze vyřešena pomocí grafického uživatelského rozhraní. Dalším rozdílem je připojení k centrálnímu serveru přes síť *TOR*[14], které je v Linux verzi v základním nastavení zakázáno. Autentizace přes toto šifrované spojení je ale opakovaně odmítnuta a pokus o změnu nastavení končí neošetřenou výjimkou a program je neočekávaně ukončen. Nemožnost změny na přímé připojení k serveru pak znemožňuje další pokusy o řádné zprovoznění honeypotu pod operačním systémem Windows.

Honeypot se tedy nepodařilo korektně zprovoznit ani pod operačním systémem Linux ani pod Windows. Nebylo tedy možné ověřit správnou funkčnost ani vlastnosti tohoto nástroje. Další zápornou vlastností také bylo to, že honeypot je k dispozici pouze jako již zkompilovaná aplikace bez možnosti dodatečných úprav zdrojových kódů, které by mohly vyřešit nekorektní fungování. Oproti tomu kladnou vlastností je možnost provozování mnoha klientských virtuálních stanic s možností centrální správy a zpracování výsledků detekce.

4.5.3 Argos[15]

Posledním ze seznamu honeypotů s vysokou mírou interakce je nástroj Argos. Jedná se o open-source nástroj pro operační systém Linux. Byl vyvinut na nizozemské *Faculty of Sciences, VU University Amsterdam* a primárně navržen jako nástroj pro detekci a generování otisků zero-day útoků.

Jedná se o nástroj založený na open-source emulačním prostředí *QEMU*, které rozšiřuje za účelem detekce útoků. To dokáže emulovat různé procesorové architektury a veškerý hardware jako skutečný. Poté, co je do virtuálního stroje nainstalován operační systém, se stává plnohodnotným zařízením. Při použití jaderného modulu *kqemu* lze urychlit běh virtualizovaného operačního systému a výkon je srovnatelný s fyzickým strojem. V případě nástroje Argos není tento modul podporován, a tak emulace dosahuje podstatně nižšího výkonu.

Argos je šířený ve formě zdrojového kódu napsaného v jazyce C. Jde o nástroj, který dokáže emulovat celý operační systém i s jeho službami bez nutnosti dělení fyzických disků.

[14] https://www.torproject.org/
[15] http://www.few.vu.nl/argos/

Mezi podporované operační systémy patří například Linux, Windows XP a Windows 2000 a další (všechny operační systémy podporované QEMU).

Tvůrci se při návrhu nástroje snažili o vylepšení tehdy dostupných řešení. Zkombinovali jejich nejdůležitější a nejlepší vlastnosti, a tím vylepšili schopnost detekce. Argos tak poskytuje ochranu celého operačního systému, všech jeho procesů a ovladačů. Další vlastností je podpora komplexních paměťových operací, jako je *memory mapping* a *DMA*[16], napomáhající odhalit i důmyslné typy útoků. Cílem detekce není *payload*[17], ale samotný *exploit*[18]. Tyto pojmy, představující určitý zdrojový kód, se do českého jazyka nepřekládají a běžně se využívá originální anglický název. Argos tak detekuje kód zneužívající bezpečnostní chybu, ale už dále nezjišťuje, co se stane po jeho provedení. Hlavním důvodem této metody detekce útoků je podstatně těžší způsob polymorfizace kódu exploitu a také to, že při zachycení samotného exploitu útok dále nemůže pokračovat. Poslední vlastností je využití vlastního forenzního kódu, který Argos injektuje do napadeného procesu, a dovoluje tak zjistit důležité informace o exploitu. Ze všech popisovaných vlastností je patrné, že Argos je nástroj zaměřený na útoky zneužívající chyby v kódu a nikoliv na útoky způsobené nedostatečnou ochranou systému nebo nevhodnou konfigurací.

Pro detekci útoků Argos využívá tzv. *dynamic taint analýzu*, kdy sleduje úpravy dat a detekuje pokusy o jejich využití nedovoleným způsobem. Jedná se o běžně používanou techniku analýzy v oblasti bezpečnosti. Celý princip spočívá v označování specifických vstupních dat jako tzv. *tainted*. V tomto případě se specifickými daty rozumí všechna data směřující ze sítě. Jejich pohyb způsobuje označování všech paměťových bloků a procesorových registrů, ve kterých tato data figurují. Pokud je pak takové místo v paměti použito v některé z instrukcí nebo za běhu programu, jsou podniknuty kroky k přerušení infiltrace a pokus o útok je zaznamenán.

Většina útoků získává kontrolu nad systémem přesměrováním toku programu na instrukce vložené samotným útočníkem. V architektuře *x86* je adresa na následující vykonávanou instrukci uchována v registru *EIP*. Argos tedy kontroluje nahrávání adres do tohoto registru a spolu s informacemi o stavu tainted lze zachytit širokou škálu útoků. Další kategorií exploitů, kterou dokáže Argos detekovat jsou zranitelnosti formátovacího řetězce.

[16] Direct Memory Acces – přenos dat mezi pamětí a V/V zařízením bez účasti procesoru.
[17] Kód provedený po úspěšné exploitaci.
[18] Programový kód využívající k přístupu bezpečnostní chybu.

Útočník může v tomto případě přepsat jakékoliv paměťové místo libovolnou hodnotou. Detekce je v tomto případě doplněna o schopnost zachytit injektování kódu z libovolného místa v paměti. Posledním detekovatelným způsobem jsou útoky využívající změny parametrů kritických funkcí, jako jsou například systémová volání. Parametry těchto volání jsou tedy při detekci označeny jako tainted.[14]

Všechny záznamy o útocích na honeypot jsou ukládány do adresáře, odkud byl Argos spuštěn. Pro každý útok je vytvořen nový soubor a v terminálu je zaznamenán výpis o provedeném útoku. Data zachycená na monitorovaném síťovém rozhraní jsou pak ukládána do samostatného souboru společného pro všechny útoky. Argos generuje data ve formě binárních souborů, a to jak pro log soubory k útokům, tak pro log soubor se záznamy síťového provozu.

Honeypot Argos sám o sobě nedokáže analyzovat charakteristiky zachycených útoků, ale je zaměřen především na generování signatur pro další nástroje, jako jsou systémy IDS/IPS. Poskytuje velmi důležité informace o zachycených exploitech. Vytvořené log soubory obsahují hodnoty procesorových registrů a data ze všech paměťových bloků, které se účastnily útoku. Tato data lze spolu s informacemi zachycenými na síťovém rozhraní prostřednictvím *reverzního inženýrství*[19] dále analyzovat a použít v dalších aplikacích pro obranu před útočníky. Velkou výhodou je také distribuce v podobě kompletních zdrojových kódů. Nástroj tak lze s určitými znalostmi přizpůsobit vlastní potřebě. Mezi nevýhody patří komplikovanější zpracování výstupních dat v binární podobě. Další nevýhodou jsou omezené možnosti zpracování a použitelnost získaných informací bez pokročilých znalostí technik reverzního inženýrství a analýzy paměťových otisků.

[19] Zpětná analýza, jejímž cílem je pochopení principů fungování.

4.6 Honeynety

Jedná se o specifickou kategorii. *Honeynet* lze chápat jako síť propojených honeypotů. Obvykle se jedná o honeypoty s vysokou mírou interakce, emulující různé operační systémy s různými aplikacemi. Jde tedy o záměrně nezabezpečenou síť vytvořenou s úmyslem nalákat případné útočníky, následně pak analyzovat a studovat jejich činnost a metody. Tyto informace pak lze využít ke zvýšení zabezpečení sítě. I když primárním účelem je shromažďovat informace, provozování honeynetu může být prospěšné i jinak, například tím, že odkloní zájem útočníků od reálných sítí. Hlavním cílem honeynetů je pak agregace získaných informací. Jedním z nejznámějších představitelů honeynetu je *The Honeynet Project* [15].

5 Implementace nástroje Argos

Mezi další cíle této práce patří instalace vybraného nástroje ze skupiny honeypotů s vysokou mírou interakce. Dostupné jsou zejména již dříve jmenované Capture-HPC, Honey@Home a Argos. Výběr byl proveden testováním dvou zástupců serverových honeypotů s vysokou mírou interakce – Honey@Home a Argos.

Jak již bylo popsáno, instalace nástroje Honey@Home proběhla bez problémů. Jeho schopnosti detekce však nebylo možné díky nekorektní funkčnosti ověřit. Z dostupných honeypotů zůstal pouze Argos. Zbývalo tedy otestovat, zda se jedná o vhodný výběr a provedená instalace a konfigurace bude bezproblémová. Další podmínkou při výběru bylo také to, jak Argos obstojí při některém z testovacích útoků.

Zkušební instalace byla provedena na virtuálním stroji realizovaném za pomoci prostředí *VMware*[20]. Zde je hlavní rozdíl oproti finální instalaci. Virtuální stroj neměl žádný přístup do sítě a komunikace probíhala pouze mezi hostujícím strojem a honeypotem. Odpadlo tedy zabezpečení proti zneužití honeypotu útočníkem z vnější sítě. Několik provedených útoků bylo úspěšně zachyceno a Argos vygeneroval o každém z nich log soubor. Z nich je pak možné zjistit podrobnější informace o jejich provedení. Veškeré útoky byly prováděny za pomocí programu *Metasploit framework.*[21] Tento nástroj a práce s ním budou dále popsány níže v kapitole 6 věnované útokům na běžící honeypot.

Instalace a konfigurace byla téměř shodná, jak pro testovací účely, tak i pro využití honeypotu v síti. V následujícím textu bude popsán postup instalace, konfigurace, provedené změny a realizace zabezpečení proti infiltraci z vnější sítě. Tím je zamezeno využití honeypotu k napadení ostatních stanic v síti nebo jeho zneužití ke škodlivé činnosti.

V další části pak bude popsán způsob, jakým byly provedeny simulované útoky a také to, jak je honeypot detekuje a interpretuje výsledky. S tím souvisí popis a struktura získaných log souborů v kapitole 6.1.

[20] http://www.vmware.com/
[21] http://www.metasploit.com/

5.1 Instalace

Jak již bylo výše uvedeno, postup instalace je v obou případech stejný. Jediný rozdíl spočíval v tom, že finální verze byla nainstalována přímo na fyzický stroj. Tím byly odstraněny problémy s vyšší náročností na hardware a honeypot běžel stabilněji. Původním záměrem i bodem zadání bylo využití serveru přímo ve fyzické síti Vysokého Učení Technického v Brně. Tato možnost však nebyla i přes veškerá jednání z důvodů potencionální hrozby napadení povolena. Bylo tedy nutné využít náhradního řešení, a to sestavení samostatného stroje[22] a jeho připojení do privátní sítě. Jako operační systém byla nejprve zvolena Linux distribuce Ubuntu 10.10. Zde však snaha o instalaci končila chybovým hlášením, které po jeho vyřešení střídaly další problémy. Toto nekončící řešení problémů bylo nakonec způsobeno právě zvolenou distribucí. Po instalaci Ubuntu 10.04 se tyto chyby neobjevovaly a nic nebránilo v další práci a fungování honeypotu.

Následující část textu bude tedy věnována instalaci honeypotu. Argos lze získat z domovských stránek ve formě archívu obsahujícího zdrojové kódy v jazyce C. Byla tedy zvolena poslední stabilní verze 0.4.2-1 z roku 2009. Podmínkou pro jejich zkompilování je nainstalovaný překladač gcc. Při kompilaci za pomoci gcc verze 4.4.5 je zobrazeno varování z výpisu kódu 5.1 a je třeba použít verzi 3.x.

Výpis kódu 5.1: Výpis z terminálu při kompilaci

```
$ ./configure --enable-net-tracker
WARNING : "gcc" looks like gcc 4.x
Looking for gcc 3.x
gcc 3.x not found!
ARGOS is known to have problems when compiled with gcc 4.x
It is recommended that you use gcc 3.x to build ARGOS
To use this compiler anyway, configure with --disable -gcc -check
```

Parametr `--enable-net-tracker` aktivuje funkci *Network Tracker* zajišťující logování informací ze síťového rozhraní. Tu je nutné aktivovat již při kompilaci, neboť v implicitním nastavení není podporována z důvodu velké náročnosti na paměť a procesorový čas [16]. Aktivovaný Network Tracker mód při spuštění honeypotu vytvoří samostatný log soubor, obsahující právě zachycená data ze síťového rozhraní. Obsah a struktura souborů

[22] AMD Athlon II X2 250 (x86), 4GB RAM, 500 GB HDD.

bude popsána níže v kapitole 6.1. Instalace nástroje Argos se pak provádí klasickými příkazy terminálu `make & make install.`

5.2 Virtuální stroj

Dalším krokem po úspěšné instalaci je vytvoření virtuálního stroje. Lze ho vytvořit přímo za pomoci nástroje Argos anebo pomocí QEMU. Postup je totožný, QEMU není samo o sobě tak náročné na výkon, a celý proces je proto časově méně náročný. Nejprve je ale nutné nainstalovat balík *libsdl1.2-dev* pro grafické zobrazení a pro síťové spojení pak balík *uml-utilities* obsahující TUN/TAP interface. Pomocí něj bude konfigurováno síťové propojení honeypotu s fyzickou sítí. Oba tyto balíky jsou nutné i pro fungování samotného virtualizačního prostředí QEMU

Výpis kódu 5.2: Instalace SDL a TUN/TAP

```
sudo apt-get install libsdl1.2-dev
sudo apt-get install uml-utilities
```

Následně je pomocí nástroje `qemu-img` vytvořen soubor *winxp.img* o velikosti 10 GB, reprezentující virtuální disk. Do tohoto souboru je pak nainstalován operační systém Windows XP. Parametr `-localtime` zajistí u virtuálního stroje stejný lokální čas jako u hostitelského systému. Další parametry nastavují velikost operační paměti (`-m 1G`), cestu k virtuálnímu systémovému disku (`-hda ~/win/winXP.img`) a cestu k instalačnímu médiu (`-cdrom ~/win/xp.iso`). Namísto instalace z obrazu disku ISO lze samozřejmě použít i fyzické zařízení CD-ROM použitím cesty `/dev/cdrom.`

Výpis kódu 5.3: Vytvoření virtuálního stroje

```
sudo apt-get install qemu
qemu-img create -f qcow winxp.img 10G
qemu -localtime -m 1G -hda ~/win/winXP.img -cdrom ~/win/xp.iso
```

Tím je Argos nainstalován a mělo by být možno spustit virtuální stroj pod jeho kontrolou.

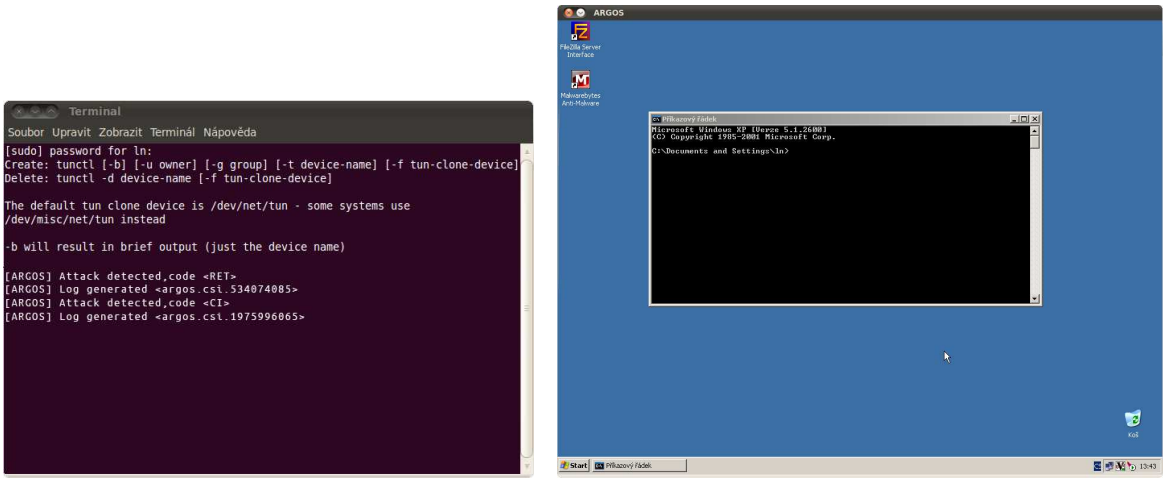
5.3 Spuštění a provoz

Argos je možné spustit stejně jako klasické emulační prostředí QEMU. Pouze s jediným rozdílem, že Argos je nutné spouštět s právy uživatele `root.` Běžící virtuální stroj je možno ovládat a konfigurovat jako běžný počítač s již dříve nainstalovaným operačním systémem Windows XP. Činnost spuštěného nástroje je patrná z obrázku 5.1.

Výpis kódu 5.4: Spuštění virtuálního stroje Argos

```
$ sudo argos -localtime \
             -m 512 \
             -hda ~/win/winXP.img \
             –winxp \
             –net tap,ifname=tap0,script=/etc/argos-ifup,\
              downscript=/etc/argos–ifdown
```

Z výpisu kódu 5.4 je patrné, že spouštění probíhá se stejnými parametry jako klasický virtuální systém. Navíc je pouze parametr `–winxp`, který udává emulovaný operační systém a je zde z důvodu korektního fungování honeypotu. Po spuštění je otevřeno nové okno s běžícím systémem Windows XP a nadále zůstává otevřen i terminál, ve kterém je indikována činnost honeypotu. Posledním parametrem definujícím vlastnosti síťového připojení je `–net tap.` Ten bude popsán v následující části této kapitoly.



Obr. 5.1: Argos monitorující virtuální systém

5.4 Konfigurace síťového nastavení

5.4.1 Připojení honeypotu do sítě

Doposud je honeypot spuštěn v režimu, kdy jeho síťová adresa je překládána pomocí *NAT* [23], v síti se tak prezentuje pod adresou svého hostitelského systému. To pro fungování a princip honeypotu, který chceme veřejně vystavit do sítě a detekovat pomocí něj útoky, není příliš vhodné. Pomocí již nainstalovaného TUN/TAP je tedy třeba vytvořit virtuální síťové rozhraní a přes síťový most propojit Argos s rozhraním fyzického stroje. Síťové rozhraní honeypotu je pak přes tento bridge spojeno přímo s fyzickou sítí a lze mu přidělit adresu z rozsahu lokální sítě. Tato konfigurace je nutná při každém startu honeypotu, vše je tedy vhodné nastavovat pomocí skriptu. Počáteční vytvoření virtuálního síťového rozhraní a jeho přidání do bridge zajistí skript `argos-ifup.sh`. Zde pak přichází na řadu vysvětlení parametru `–net tap` při spouštění honeypotu. Je patrný název síťového rozhraní, přes které se Argos bude připojovat do sítě (`ifname=tap0`). Dalšími dvěma částmi jsou pak cesty ke skriptům, které se spouští při startu `(script=/etc/argos-ifup)`a při ukončování honeypotu (`downscript=/etc/argos−ifdown`.

Výpis kódu 5.5: Obsah skriptu argos-ifup.sh

```
#!/bin/sh

echo 1 > /proc/sys/net/ipv4/ip_forward

sudo brctl addbr br0
sudo brctl addif br0 eth0
sudo tunctl −b −u ln
sudo brctl addif br0 tap0
sudo ifconfig eth0 192.168.88.2 promisc up
sudo ifconfig tap0 0.0.0.0 promisc up
sudo ifconfig br0 192.168.88.80 up
sudo route add default gw 192.168.88.1
```

Pomocí skriptu je nejprve v jádru hostujícího operačního systému povolen IP forward. Následně je pak za pomoci nástroje `brctl` vytvořen síťový most `br0` a do něj přidány jak virtuální `tap0`, tak fyzické síťové rozhraní `eth0`. Síťovému mostu je navíc ještě přidělena

[23] Network Address Translation – způsob přepisu zdrojové nebo cílové adresy v routeru.

statická ip adresa pro případnou správu. Tento krok však není pro funkci honeypotu nutný. Skript zajišťující konfiguraci síťového mostu je patrný z výpisu kódu 5.5.

Od tohoto okamžiku pak virtuální stroj monitorovaný za pomoci nástroje Argos může mít svou vlastní IP adresu v síti. Tu lze nastavit přímo v nově spuštěném okně s operačním systémem Windows XP. Klasickým způsobem je v systému nastavena buď jako statická IP adresa, anebo si ji lze nechat přidělit DHCP serverem, pokud je v síti v provozován.

Skript `argos-ifdown.sh` (výpis kódu 5.6) nedělá nic jiného, než že vrací původní konfiguraci hostitelského systému do stavu před spuštěním honeypotu Argos.

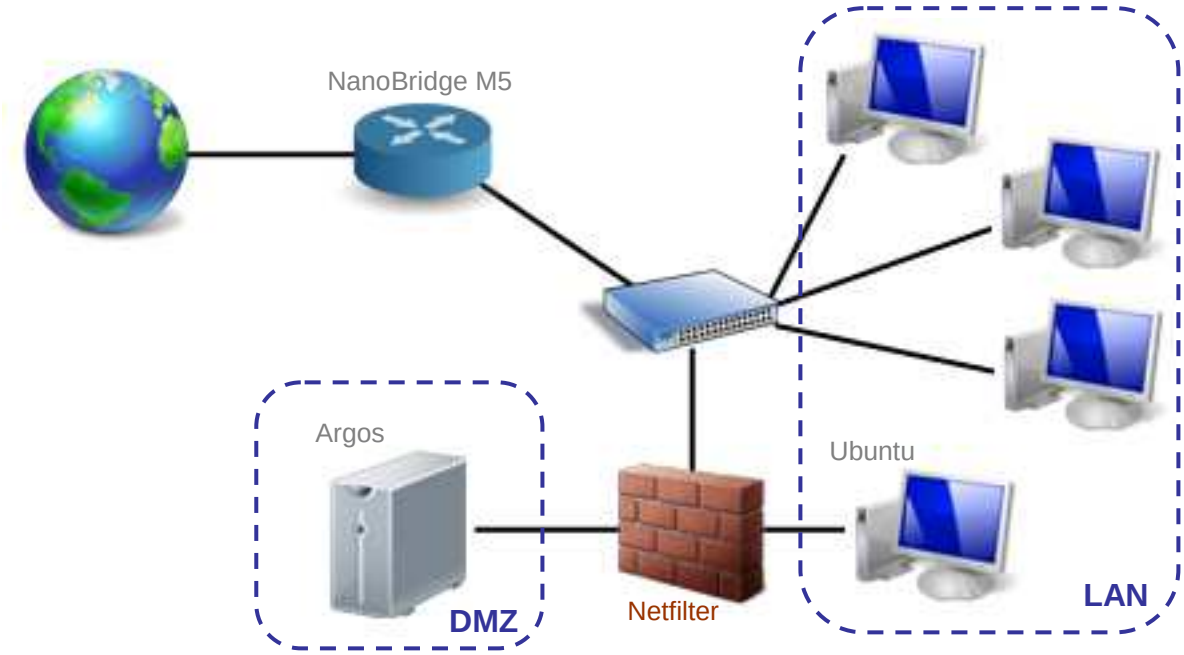
Výpis kódu 5.6: Obsah skriptu argos-ifdown.sh

```
#!/bin/bash

sudo brctl delif br0 eth0
sudo brctl delif br0 tap0
sudo ip a flush dev br0
sudo brctl delbr br0
sudo ip a flush dev tap0
sudo tunctl -d tap0
sudo dhclient eth0
```

5.4.2 Zabezpečení proti zneužití honeypotu

Argos je v tuto chvíli již schopný provozu a detekuje příchozí podezřelé chování. I když účinnost detekce dosahuje dobrých výsledků a vykazuje vysokou spolehlivost, nelze brát honeypot jako plnohodnotný prvek ochrany. Může nastat situace, kdy nebude schopen zabránit napadení sledovaného systému. Útočník ho pak může použít k další škodlivé činnosti nebo s jeho pomocí kompromitovat ostatní stroje v síti. Tím se lze dostat například k citlivým soukromým informacím, aniž by byly uloženy právě v honeypotu, nebo mohou být stroje použity jako součást *botnetu*[24]. Cílem tedy bude zabránit případnému útočníkovi tohoto využít. Prvním krokem prevence proti napadení je umístění honeypotu do *DMZ*[25]. Toto řešení odděluje zařízení umístěné v DMZ od ostatních strojů, avšak umožňuje k němu přístup jak z vnější, tak z vnitřní sítě. Obrázek 5.2 představuje strukturu sítě, v níž je Argos spuštěn.

[24] Síť počítačů infikovaných malware používané ke škodlivé činnosti.
[25] Demilitarized Zone, část sítě oddělená od ostatních zařízení.



Obr. 5.2: Struktura sítě

Samotné oddělení DMZ a *LAN*[26] obstarává jednotka *Ubiquiti NanoBridge M5*[27]. Ta také zprostředkovává překlad veřejné IP adresy 62.209.208.92 na lokální adresu honeypotu 192.168.80.66. Dalším prvkem, který zabezpečuje veškerou komunikaci v síti, je pak firewall *Netfilter* nainstalovaný na hostitelském stroji. Stačí tedy, aby filtroval veškerou příchozí a odchozí komunikaci fyzického stroje. Tím je zabezpečeno, že jsou kontrolovány i datové toky virtuálního honeypotu. Hostitelský stroj je pak spolu s dalšími stanicemi v jedné LAN z rozsahem 192.168.88.0/24.

Firewall *Netfilter* je hlavním prvkem zabezpečení proti zneužití honeypotu útočníky. Jedná se o stavový firewall, který je přímo součástí jádra operačního systému Linux. Pro jeho nastavení pak slouží nástroj *iptables*. Pravidla definujeme pomocí příkazů specifikujících, jak se má naložit s jednotlivými pakety vstupujících do filtru. Tyto pravidla lze zapsat rovnou přes standardní terminálový vstup, avšak po restartu je konfigurace firewallu obnovena do původní podoby. Nastavení je opět vhodné provádět pomocí skriptu spouštěného po startu operačního systému nebo současně se startem honeypotu [17].

Iptables implicitně pracuje se vstupními řetězci INPUT, OUTPUT a FORWARD a jejich politikou nastavenou na ACCEPT, což znamená, že veškerá komunikace je povolena.

[26] Local Area Network – lokální počítačová síť.
[27] http://www.ubnt.com

Firewall se ve většině případů vytváří způsobem, kdy veškerou komunikaci zakážeme a povolujeme pouze potřebné služby. Lze tak lépe kontrolovat, které pakety filtrem mohou projít a které naopak ne.

Díky spojení virtuálního a fyzického síťového rozhraní do bridge prochází všechny pakety řetězcem FORWARD. V jednotlivých pravidlech je navíc pomocí parametru `physdev` nutné specifikovat, které rozhraní je pro procházející pakety vstupní a které výstupní. Řetězce INPUT a OUTPUT tak mohou být ponechány i s politikou ACCEPT.

Výchozím bodem konfigurace firewallu bude tedy změna komunikace přes řetězec FORWARD na DROP, neboli zakázat vše. Následně je aplikována politika podle níže popsaných pravidel. Veškeré pakety směrem k honeypotu prochází bez zahození a ten se pak pro útočníka jeví jako nezabezpečený. Prevence proti zneužití spočívá ve filtrování jeho odchozích spojení. Tok dat směrem od honeypotu je zakázán. Výjimku tvoří pouze již navázané spojení a v případě LAN i ty, jež s nimi souvisí. Tím zabezpečíme povolení odchozích dat, která si vyžádal některý ze strojů v síti. Další pravidla pak navíc povolují odchozí spojení služeb spuštěných v honeypotu. Patří sem ICMP pakety a dále pak spojení pro HTTP, HTTPS, FTP, VNC a sdílení souborů. Odchozí pakety jiných služeb nemají důvod být propuštěny. Výpis kódu 5.7 obsahuje nastavenou politiku firewallu.

Výpis kódu 5.7: Politika firewallu Netfilter

```
sudo modprobe ip_conntrack_ftp

sudo iptables -P INPUT ACCEPT
sudo iptables -P OUTPUT ACCEPT
sudo iptables -P FORWARD DROP

sudo iptables -A FORWARD -m physdev --physdev-in eth0 --physdev-out tap0
-j ACCEPT

sudo iptables -A FORWARD -m physdev --physdev-in tap0 --physdev-out eth0
-m state --state ESTABLISHED -j ACCEPT

sudo iptables -A FORWARD -m physdev --physdev-in tap0 --physdev-out eth0
-m iprange --dst-range 192.168.88.1-192.168.88.100 -j ACTION -m state
--state RELATED -j ACCEPT


sudo iptables -N tcpfiltr
sudo iptables -N udpfiltr
sudo iptables -N icmpfiltr
```

```
sudo iptables -A tcpfiltr -p TCP --dport 80 -m physdev --physdev-in tap0
--physdev-out eth0 -j ACCEPT

sudo iptables -A udpfiltr -p UDP --dport 53 -m physdev --physdev-in tap0
--physdev-out eth0 -j ACCEPT

sudo iptables -D icmpfiltr -p ICMP --icmp-type 0 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -D icmpfiltr -p ICMP --icmp-type 8 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -D icmpfiltr -p ICMP --icmp-type 11 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -A tcpfiltr -p TCP --dport 20:21 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -A tcpfiltr -p TCP -m multiport --destination-ports 22,443
-m physdev --physdev-in tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -A tcpfiltr -p TCP --dport 137:139 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -A tcpfiltr -p UDP --dport 137:139 -m physdev --physdev-in
tap0 --physdev-out eth0 -j ACCEPT

sudo iptables -A tcpfiltr -p TCP -m multiport --destination-ports
5800,5900 -m physdev --physdev-in tap0 --physdev-out eth0 -j ACCEPT
```

6 Detekce útoků pomocí nástroje Argos

Poslední kapitola bude věnována útokům na běžící honeypot. Ty lze rozdělit do dvou skupin. První z nich jsou skutečné útoky přicházející z veřejně sítě internet. Cílem se stává jak celkový systém Windows XP, tak jednotlivé služby a programy, které na něm běží. Útočníci se snaží kompromitovat systém prostřednictvím otevřených portů a bezpečnostních děr. Celý systém běží záměrně s minimálním zabezpečením, a tak poskytuje více možností, jak provést útok využívající tohoto stavu.

Druhou skupinou jsou útoky provedené cíleně za pomoci nástroje Metasploit framework. Jedná se o open-source nástroj pro penetrační testování, vývoj a použití exploitů. Jeho funkce využívá mnoho bezpečnostních techniků, testerů i hackerů. Pochopení, jak provádět útoky na bezpečnostní díry systému a principy exploitace, jsou nad rámec této práce. S pomocí tohoto nástroje tak odpadly složité způsoby a postupy při napadání systému. Usnadnil také výběr z nepřeberného množství exploitů a payload kódů. Použitá verze 4.3 obsahuje více než 919 exploitů a 255 payload kódů. Práce s nástrojem je relativně jednoduchá oproti využití exploitu přímo.

Metasploit framework je šířen pro všechny typy operačních systémů. Pro exploitaci byl využit stroj s operačním systémem Linux. Je proto nutné z domovských stránek stáhnout archiv určený pro tento operační systém. Dále už jen stačí jej rozbalit. Posledním krokem před spuštěním je instalace *Ruby*[28].

Pro práci lze využít jak grafické rozhraní `msfgui`, tak textovou konzoli `msfcli`. Práce s konzolí je vhodnější pro opakované použití stejných nebo podobných postupů, například pomocí skriptů. Její ovládání a práce s ní je intuitivní, a tak byla zvolena pro použití tohoto nástroje.

Postup při provedení všech útoků je podobný. Spuštění nástroje se provede pomocí příkazu `msfcli` s příslušnými parametry. Jako parametr je použit název exploitu a jeho volby a nastavení. Seznam všech použitelných modulů lze vypsat příkazem `msfcli` bez parametru. Tento seznam je však díky počtu exploitů poměrně dlouhý. Jedním z nejdůležitějších parametrů každého modulu je volba `RHOST`. Ta definuje cílovou adresu počítače, na který bude proveden útok. Další volbou je `RPORT` definující cílový port. Ve

[28] http://www.ruby-lang.org/

většině případů ji však není nutné měnit z přednastavené hodnoty. Posledním parametrem je použitý modul využívající konkrétní payload. Pro všechny útoky byl použit *win32_bind*, který vrací příkazový řádek napadeného systému.

Tímto způsobem bylo provedeno patnáct útoků na nejrůznější služby a bezpečnostní díry. Ve všech případech Argos detekoval pokus o infiltraci systému a k příslušné události vygeneroval log soubor. Všem pokusům byl odepřen přístup, a tak se nepodařilo proniknout do systému. Následně byl proveden stejný test na systém spuštěný v QEMU. Ve dvanácti případech se podařilo získat přístup k příkazovému řádku Windows. Zbylé neúspěšné pokusy lze přičíst nejspíše k použití české jazykové mutace systému Windows XP.

6.1 Systém log souborů

Ke každému zachycenému útoku Argos generuje log soubor obsahující podrobnosti o využití paměťových bloků. Do dalšího log souboru pak zachycuje data ze síťového rozhraní. Záznamy o útocích na honeypot jsou generovány ve formě binárních souborů, a to jak pro log soubory k útokům, tak pro log soubor Network tracker módu. Argos jednotlivé soubory ukládá do adresáře, ze kterého byl spuštěn. Současně s tím jsou na výstup terminálu vypisovány záznamy o detekci provedeného útoku (výpis kódu 6.1).

Výpis kódu 6.1: Záznam o provedeném útoku na terminálovém výstupu

```
[ARGOS] Attack detected,code <RET>
[ARGOS] Log generated <argos.csi.241440870>
```

6.1.1 Formát log souborů

Název všech souborů obsahujících signaturu útoku má tvar *argos.csi.[rid]*, kde *[rid]* je náhodně generované číslo pro každý útok. Každý log je generován podle stejné struktury. Začíná hlavičkou, následuje část složená z hlavičky paměťového bloku a jeho obsahu. Konec souboru je rozpoznán prázdnou hlavičkou paměťového bloku.

Formát hlavičky je patrný z obrázku 6.2. Vždy začíná polem FORMAT, které určuje podobu hlavičky. Další pole ARCH specifikuje architekturu systému. Následuje pole TYPE identifikující typ útoku. Argos dokáže identifikovat celkem osm typů útoků rozlišených způsobem využití registrů a použitých instrukcí.

HEADER	
MEMORY BLOCK HEADER	MEMORY BLOCK CONTENTS
…	…
MEMORY BLOCK HEADER	

Obr. 6.1: Struktura log souboru

Následuje pole TIMESTAMP s časovým razítkem útoku. Pole REGISTER VALUES obsahuje hodnoty základních procesorových registrů. Ty jsou uloženy v pořadí EAX, ECX, EDX, EBX, ESP, EBP, ESI a EDI. Obsah těchto registrů je následován značkami jednotlivých registrů. Definují, zda byl registr označen jako tainted. Pokud ano, hodnota udává, ze které adresy v paměti byla data nahrána. Pokud jsou data nahrána přímo ze sítě, obsahují hodnotu 0xffffffff. Další dvě pole definují hodnotu a značku registru EIP, zde většinou bývá uložena adresa v paměti podstrčená útočníkem. Pole EIP NET TRACKER VALUES a NET TRACKER TAG obsahují data související s aktivovaným Net Tracker módem. Posledním polem je EFLAGS VALUE obsahující hodnotu registru EFLAGS.[18]

FORMAT	ARCH	TYPE	TIMESTAMP
REGISTER VALUES			
TAGS VALUES			
NET TRACKER VALUES			
EIP VALUE		EIP TAG	
EIP NET TRACKER TAG			
EFLAGS VALUE			

Obr. 6.2: Struktura hlavičky log souboru

6.2 Analýza zachycených dat

Jelikož jsou všechny log soubory generovány v podobě binárních dat, nelze je jednoduše číst přímo po vytvoření. Je tedy nutné nějakým způsobem převést data do textové podoby. V případě souborů generovaných pro jednotlivé útoky lze použít aplikaci *logcheck*. Konvertovaná textová data mají pak strukturu velmi podobnou struktuře dat binárních. Z log souboru je patrná architektura monitorovaného systému, typ použitého útoku, obsah všech registrů i paměťové bloky využité k útoku. Pro přehlednější výpis všech informací může být použita i aplikace *carlog*, která k tomuto účelu byla původně vytvořena jedním z autorů nástroje Argos.

Log soubor vygenerovaný Net tracker módem lze konvertovat do čitelné podoby za pomoci aplikace *netlog2pcap*. Výsledná data mají formát *pcap*[29]. Ten lze poté bez problémů otevřít a analyzovat např. za pomocí aplikace *Wireshark*[30]. Všechny aplikace pro zpracování binárních log souborů jsou přiloženy na CD v adresáři */utils*.

6.2.1 Signatury útoků

Struktura log souborů obsahujících informace o zachyceném útoku byla popsána v kapitole 6.1. Jednotlivé log soubory v binární podobě společně s daty zachycenými na síťovém rozhranní slouží jako signatury pro systémy IDS/IPS. Dle [14] jsou tato data kompatibilní s nástrojem *Snort*[31]. Argos tedy může sloužit jako zdroj signatur důležitých pro práci tohoto nástroje.

V následujícím textu bude na konkrétním případu popsáno, jakým způsobem lze analyzovat uložená data. Pro získání výsledků neexistují žádné automatizované způsoby a je nutné použít individuální přístup ke každému útoku. Bez potřebných znalostí je však toto získávání informací velmi těžké. Další překážkou je také to, že pokročilé techniky analýzy kódu, reverzního inženýrství, znalost jazyka symbolických instrukcí a paměťových operací jsou nad rámec této práce. S využitím dostupných znalostí tak budou získány základní informace z log souboru obsahujícího data o zachyceném útoku a log souboru vygenerovaného Net tracker módem.

[29] packet capture data
[30] http://www.wireshark.org
[31] http://www.snort.org/

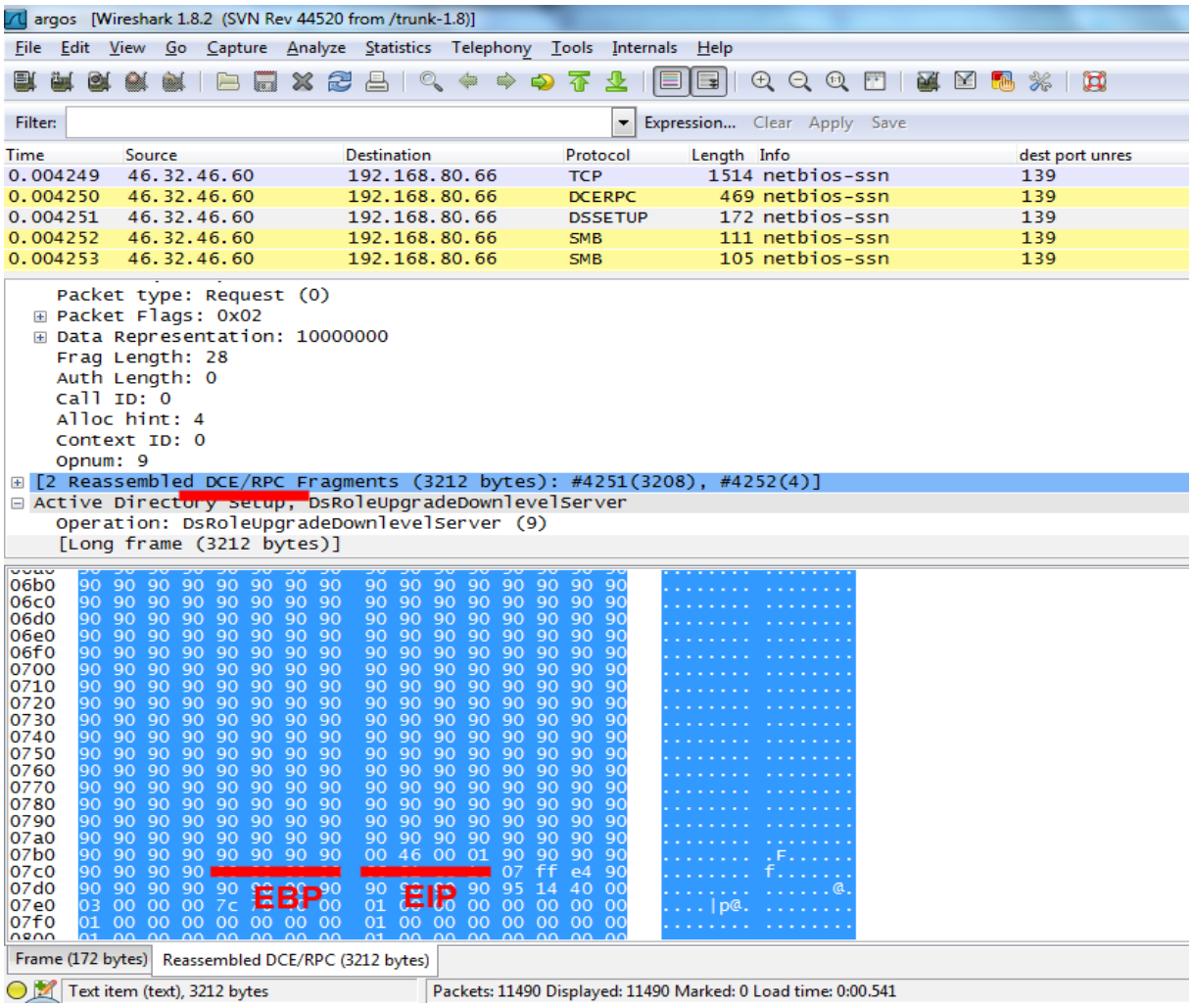
Vybraný soubor `argos.csi.534074085` je nejprve nutné převést do textové podoby, která usnadňuje jeho čtení. Výstup terminálového programu carlog je přesměrován do nového souboru. Výpis kódu 6.2 obsahuje výsledná data.

Výpis kódu 6.2: Informace o provedeném útoku

```
$ carlog argos.csi.534074085 argos.netlog >> argosdata

Net tracker data: YES

VERSION    ARCH       TYPE       TIMESTAMP
0x02       i386       RET        1334487031

EAX             ECX             EDX             EBX
0x48544950      0x017df4a4      0x017df4fa      0x017d005c
(0x0c4ed45c)    (0x00000000)    (0x00000000)    (0x0c4ed49a)
[  11720120]    [  11719584]    [  11719584]    [  11720096]

ESP             EBP              ESI            EDI
0x017df45c      0x90909090      0x017df496      0x017df444
(0x00000000)    (0x0c4ed454)    (0x00000000)    (0x00000000)
[  11719584]    [  11720112]    [  11720084]    [  11720084]

EIP             Faulty EIP      EFLAGS
0x01004600      0x77c37eb2      0x00000202
(0x0c4ed458)
[  11720116]
```

Z výstupních dat jsou patrné jednotlivé informace o obsahu všech procesorových registrů v době zachycení exploitu. Typ útoku RET vede k závěru, že během útoku byla přepsána návratová adresa. Nejdůležitější informace je obsah registru EIP (0x01004600). Adresa byla přepsána právě touto hodnotou. Lze tedy očekávat, že ji bude možno najít v některém ze zachycených paketů. Analýza datového toku pomocí nástroje Wireshark je na obrázku 6.3. Registr EBP obsahuje ukazatel na vrchol zásobníku a jeho hodnota je v tomto případě rovna 0x90909090. Tato hodnota je uložena hned v předchozím rámci, a tak je velmi pravděpodobné, že byla exploitem přepsána spolu s návratovou adresou. V neposlední řadě

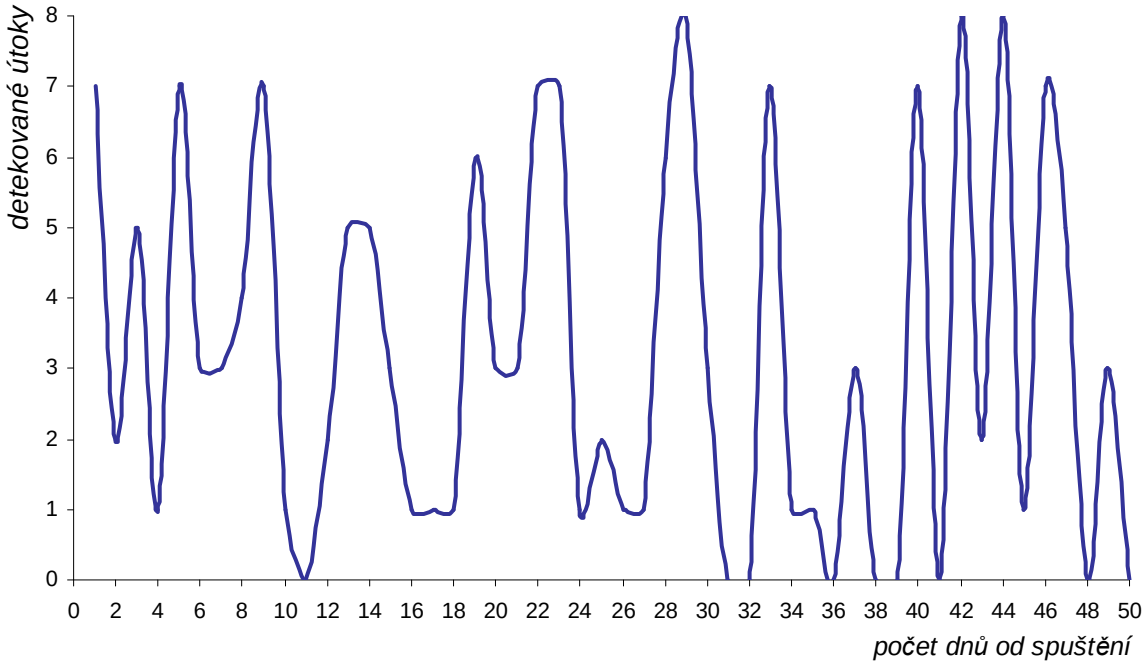
je možné si všimnout velkého množství hodnot 0x90 (instrukce NOP). Takovýto sled NOP instrukcí naznačuje, že by se mohlo jednat o útok s cílem využít přetečení zásobníku. Útočník v tomto případě využil bezpečnostní díru v rozhraní RPC a je možné, že se jedná o červ Blaster, který byl popisován v kapitole 2.2.



Obr 6.3: Obsah procesorových registrů v zachyceném paketu

6.2.2 Statistiky

Jak již bylo popsáno, zachycené útoky byly dvojího typu. V prvním případě byl honeypot spuštěn s popsanou konfigurací a připojen přímo do veřejné sítě. Jeho IP adresa nebyla nikde publikována a nebylo veřejně známo, že se jedná o kontrolovaný systém. Argos běžel nepřetržitě po dobu 50 dní. Za tuto dobu bylo detekováno a zablokováno 159 útoků. To odpovídá v průměru 3,18 útoků za jediný den, přičemž největší denní přírůstek má hodnotu 8 útoků. Z obrázku 6.4 je patrný počet zachycených útoků za dobu, po kterou honeypot běžel. V celkovém průběhu se nenachází žádné období s extrémním počtem útoků a počet napadení systému je za celou dobu sledování nepravidelný. Není známo, kolik pokusů se nepodařilo zablokovat, a útočník tak získal přístup do systému. Během 50 dní byla náhodnou kontrolou zjištěna infiltrace trojským koněm identifikovaným jako *Trojan.AlphaSoft*. Po bezpečnostním scanu systému bylo detekováno 10 souborů jako malware. Všechny soubory proto byly vymazány a systém vyléčen pomocí nástroje *Malwarebytes Anti-Malware*[32]. Díky firewallu Netfilter však zůstal trojský kůň izolován pouze v rámci virtuálního systému.



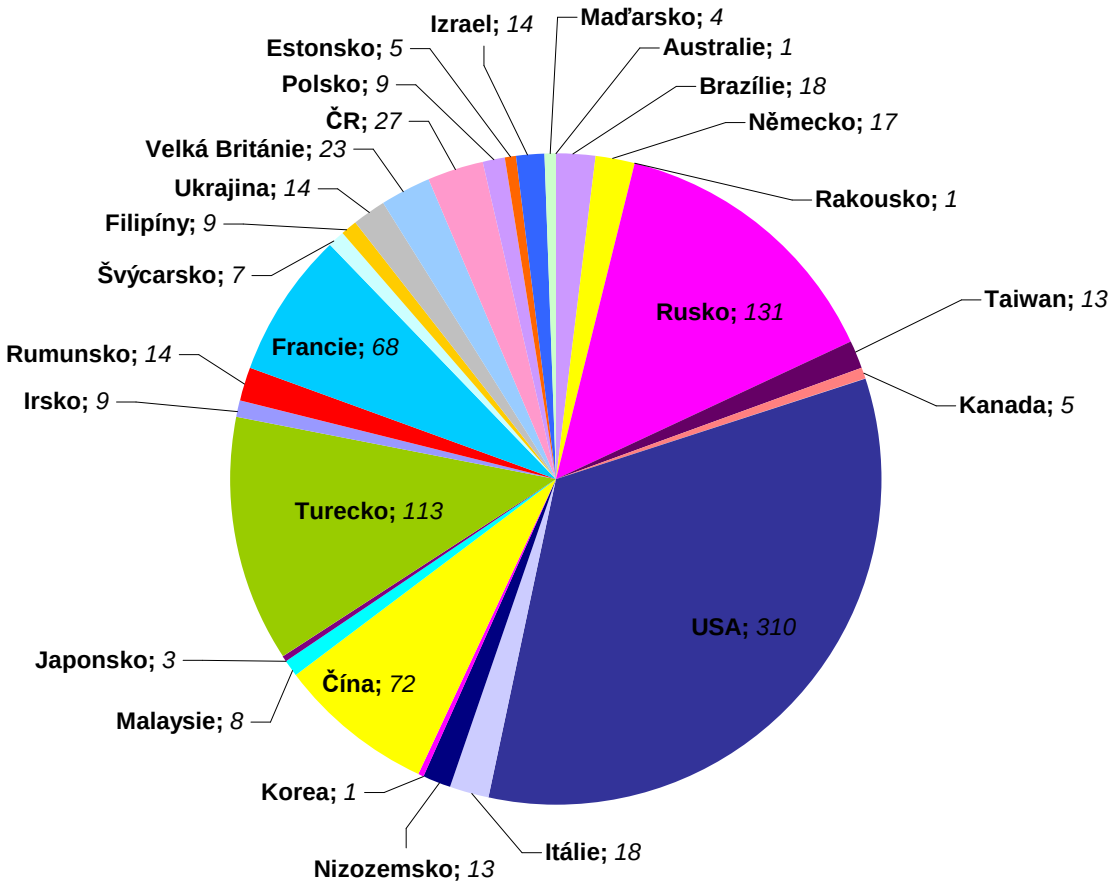
Obr. 6.4: Počty útoků na honeypot

[32] http://www.malwarebytes.org

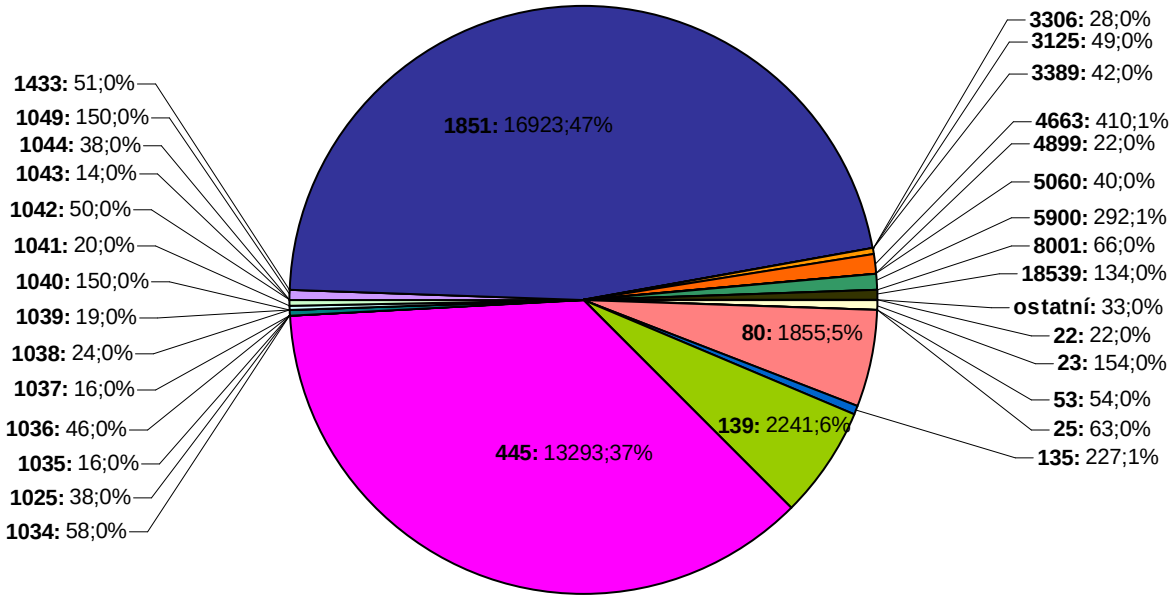
Z analýzy log souboru *argos.netlog*, který byl vygenerován Net tracker módem, bylo možné zjistit další statistické údaje o všech provedených útocích. Po analýze log souboru za pomoci nástroje Wireshark bylo možno sestavit další podrobnější statistiky. První se týká IP adres, ze kterých bylo k honeypotu přistupováno. Za celou dobu provozu byl detekován přístup z 927 unikátních IP adres. Jejich četnost podle geografického rozložení je na obrázku 6.5. Nejvíce přístupů bylo zaznamenáno podle očekávání z území USA, Ruska, Turecka a Číny.

Obrázek 6.6 se týká počtu spojení na jednotlivé porty honeypotu. Z analýzy síťových dat vyplývá, že nejvíce přístupů bylo zaznamenáno na port 1851. Tento port podle [19][19] využívá operační systém Windows pro službu *ctcd* a je častým cílem útoků. Ve velkém počtu byl také detekován přístup na porty 139 a 445, které slouží především pro sdílení souborů, tiskáren a jiných zdrojů v operačních systémech Windows. Větší množství přístupů bylo zaznamenáno i k portu 80. Mimo to je patrné, že byl detekován přístup i na velké množství portů, na kterých nebyla spuštěna žádná služba. Ve většině případů se však jednalo o počty v řádu jednotek až desítek. Skupina nazvaná *ostatní* zahrnuje veškerý zbytek portů, na které byl zaznamenán přístup a jejich počet nebyl vyšší jak 10. Jelikož se jednalo o poměrně velké množství portů, má i tato skupina výraznější zastoupení.

Statistiky všech útoků nepočítají s druhým zdrojem pokusů o kompromitaci systému, a to cílenými útoky za pomocí nástroje Metasploit framework.



Obr. 6.5: Statistika přístupů k honeypotu z jednotlivých států



Obr 6.6: Statistika přístupů na jednotlivé porty honeypotu

Závěr

Tato práce měla za cíl proniknout do tématu analýzy a detekce síťových útoků za pomocí nástrojů honeypot. Mezi další cíle patřil i výběr vhodného nástroje, následně pak jeho implementace a testování.

Práce pojednává o tématu škodlivého kódu, jeho rozdělení a možných rizicích spojených s ním. Dále popisuje současný a budoucí vývoj malware a je vzpomenuto na některé zástupce, kteří se nechvalně zapsali do dějin počítačových systémů. S tímto tématem pak úzce souvisí popsané útoky a následně pak jejich detekce a obrana proti nim.

Hlavním tématem byl popis detekce a analýzy pomocí vhodných nástrojů. Mezi ty patří honeypoty popsané v kapitole 4. Jsou zde rozděleny do kategorií podle použití a podle interakce s útočníkem. Další část popisuje jednotlivé zástupce honeypotů s vysokou mírou interakce a je zde postup výběru vhodného nástroje pro následnou implementaci. Z dostupných nástrojů byly pro důkladnější prozkoumání a testování vybrány dva honeypoty. Prvním byl Honey@Home, který však nebyl využit z důvodu přílišných komplikací při instalaci a konfiguraci. Druhým pak byl Argos, u nějž se nevyskytovaly při instalaci větší potíže a byl uživatelsky přívětivější. Následně byl podroben zběžnému prozkoumání a pokusům o útok. I zde bylo dosaženo pozitivnějších výsledků než u předchozího nástroje. Dosavadní znalosti a výsledky tak vedly k rozhodnutí, který nástroj vybrat. Z dostupných honeypotů s vysokou mírou interakce se tak pro implementaci do zkušební sítě a následně pak do sítě VUT jevil jako nejvhodnější Argos.

Celá 5. kapitola se věnuje samotné implementaci vybraného honeypotu. Je zde popsán postup instalace a konfigurace nástroje Argos, rozdíly mezi zkušební a finální instalací. V tomto bodě bylo nutné využít náhradního řešení a odchýlit se od zadání. Jako prostředek pro testování nástroje Argos nebylo možné využít síť VUT v Brně, a honeypot tak musel být spuštěn a provozován v soukromé síti na samostatném stroji postaveném k tomuto účelu. Další část kapitoly je věnována konfiguraci zabezpečení proti zneužití honeypotu.

Poslední kapitola 6 popisuje výsledky zjištěné za dobu, kdy byl Argos spuštěn a monitoroval provoz v síti. I když se nejednalo o síť, ve které by se nacházely pro útočníky lukrativní stroje a data, bylo po dobu provozu detekováno značné množství útoků. Dalším zdrojem pak byl soubor testovacích útoků. Jejich provedení a postupy jsou popsaných v téže kapitole. Další část pak pojednává o struktuře a formátu log souborů generovaných při práci

honeypotu. V neposlední řadě je kapitola věnována výsledkům detekce. Výstupní data jsou podrobena detailnější analýze, ve které je popsán postup, jakým způsobem lze získat informace o útocích. Poslední část je věnována statistikám síťového provozu.

Výstupem této práce je výběr nástroje Argos pro implementaci a další zkoumání. Byl zvolen jako nejvhodnější ze všech dostupných honeypotů. Splňoval požadavky po stránce zadání a při základních testech fungoval ze všech nejspolehlivěji. Dále se při jeho samotné instalaci vyskytlo jen několik menších problémů. Ty se podařily vyřešit, na rozdíl od problémů a chyb u ostatních nástrojů. Argos je také schopný zachytit široké spektrum útoků, včetně zero-day útoků. O každém z nich je následně veden podrobný log soubor. Všechny získané soubory obsahují důležité informace o místech v paměti, která měla něco společného s útokem. Dokáže navíc zachytit veškeré síťové pakety spojené s útokem. Výstupní log soubory ve formě binárních dat mohou sloužit jako paměťové otisky jednotlivých útoků. Ty lze použít pro budoucí preventivní činnost, například je lze ukládat do databáze a tu následně využít jako knihovnu signatur pro systémy IDS/IPS. Mezi další výhody tohoto nástroje patří také to, že je šířen ve formě open-source zdrojových kódů, které lze v případě potřeby přizpůsobit vlastním potřebám a konkrétním aplikacím.

Celá bakalářská práce je vytvořena s cílem prohloubit dosavadní znalosti a získat nové informace v oblasti malware, síťových útoků a jejich detekcí a také samozřejmě o nástrojích typu honeypot. Dále nabízí prostředky pro další podrobné zkoumání projevů bezpečnostních hrozeb, dovoluje případné použití informací a dat z log souborů pro generování signatur síťových útoků a jejich následné použití v účinné obraně.

Literatura

[1] PandaLabs. *Annual Report 2010* [online]. 2011 [cit. 2011-11-01]. Dostupné z: http://press.pandasecurity.com/wp-content/uploads/2010/05/PandaLabs-Annual-Report-2010.pdf

[2] PandaLabs. *Annual Report 2011* [online]. 2012 [cit. 2012-13-3]. Dostupné z: http://press.pandasecurity.com/wp-content/uploads/2012/01/Annual-Report-PandaLabs-2011.pdf

[3] F-Secure. Threat Description: Virus: DOS/CIH [online]. 2009 [cit. 2011-11-16]. Dostupné z: http://www.f-secure.com/v-descs/cih.shtml

[4] Security-FAQs. Malware That Changed The World – The Blaster Worm. [online]. 4.9. 2009 [cit. 2011-10-23]. Dostupné z: http://www.security-faqs.com/malware-that-changed-the-world-the-blaster-worm.html

[5] *Rootkits part 1 of 3: The Growing Threat* [online]. April 2006 [cit. 2011-11-20]. Dostupné z: http://download.nai.com/products/mcafee-avert/WhitePapers/AKapoor_Rootkits1.pdf

[6] MITNICK, Kevin a William SIMON. *Umění klamu*. 1. vyd. Gliwice: Helion, 2003. ISBN 83-736-1210-6.

[7] STALLINGS, William. *Cryptography and network security : principles and practices*. 4th ed. Upper Saddle River: Pearson Prentice Hall, 2006. ISBN 01-318-7316-4

[8] ROSS, Jason. *Malware Analysis for the Enterprise* [online]. Black Hat DC, 2010 [cit. 2011-12-01]. Dostupné z: http://www.blackhat.com/presentations/bh-dc-10/Ross_Jason/Blackhat-DC-2010-Ross-Malware-Analysis-for-the-Enterprise-wp.pdf

[9] SCARFONE, Karen a Peter MELL. *Guide to Intrusion Detection and Prevention Systems (IDPS)* [online]. Gaithersburg: National Institute of Standards and Technology, 2007 [cit. 2011-12-02]. Dostupné z: http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf

[10] 4safety. 4safety: Technologie – Honeypots [online]. 2011 [cit. 2011-10-23]. Dostupné z: http://www.4safety.cz/text/honey

[11] SearchSecurity. Zero-day exploit [online]. July 2010 [cit. 2011-12-04]. Dostupné z: http://searchsecurity.techtarget.com/definition/zero-day-exploit

[12] PROVOS, Niels a Thorsten HOLZ. *Virtual honeypots: from botnet tracking to intrusion dedction.* Upper Saddle River: Addison-Wesley, 2008. ISBN 978-0-321-33632-3.

[13] IANA. Service Name and Transport Protocol Port Number Registry [online]. 2012-05-18 [cit. 2012-05-20]. Dostupné z: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml

[14] PORTOKALIDIS, Georgios, Asia SLOWINSKA a Herbert BOS. *Argos: an Emulator for Fingerprinting ZeroDay Attacks* [online]. 2006 [cit. 2012-04-06]. Dostupné z: http://www.few.vu.nl/argos/papers/argos_eurosys06.pdf

[15] *The honeynet Project* [online]. 1999 [cit. 2011-10-23]. Dostupné z: http://project.honeynet.org

[16] VERMEULEN, Remco. *Automated Post-Attack Analysis of Injected Payloads* [online]. Amsterdam, 2011 [cit. 2012-04-13]. Dostupné z: https://gforge.cs.vu.nl/gf/download/frsrelease/153/1190/argos-0.5.0.zip. Master thesis. Vrije Universiteit Amsterdam.

[17] PAVLÍČEK, Martin. Linuxové DMZ - VII. [online]. 19. 3. 2003 [cit. 2012-05-20]. Dostupné z: http://www.abclinuxu.cz/clanky/bezpecnost/linuxove-dmz-vii

[18] Argos An emulator for capturing zero-day attacks. Argos Logs Explained. [online]. 2008 [cit. 2012-05-15]. Dostupné z: http://www.few.vu.nl/argos/docs/logs.html

[19] Corrupted Data Recovery. What is Port 1851? [online]. 2007 [cit. 2012-07-16]. Dostupné z: http://www.corrupteddatarecovery.com/Port/1851-Port-Type-tcpudp-ctcd.asp