Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Diplomová práce

Forenzní analýza chování útočníka pomocí nástroje Honeypot

Bc. Martin Zelenka

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ DIPLOMOVÉ PRÁCE

Bc. Martin Zelenka

Informatika

Název práce

Forenzní analýza chování útočníka pomocí nástroje Honeypot

Název anglicky

Forensic analysis of the attacker's behavior using Honeypot

Cíle práce

Cílem této práce je implementace nástroje Honeywall, pomocí kterého bude provedena analýza chování útočníka po připojení do napadeného systému. Dílčím cílem této práce je shrnout a analyzovat možné typy útoků na informační systémy

Metodika

Metodika řešené práce je založena na studiu a analýze odborné literatury. Praktická část práce je realizována formou návrhu modelu informačního systému, který je založen na nástroji Honeywall. Model bude zapojen po dostatečně dlouhou dobu tak, aby byla sesbírána data pro následnou analýzu útoku. Na základě syntézy teoretických a praktických poznatků budou vydedukovány závěrečné výsledky

Doporučený rozsah práce

40 – 60 stran

Klíčová slova

Honeywall, Honeypot, Roo, Hacking, Honeynet, Backdoor, Rootkit, Exploit

---

Doporučené zdroje informací

Joel Scambray, Stuart McClare, George Kurtz. Hacking Exposed – network security secrets&solutions, McGraw-Hill Education, 2012, ISBN-10: 0071780289

Martti Lehto, Pekka Neittaanmäki. Cyber Security: Analytics, Technology and Automation, Springer International Publishing, 2015, ISBN: 978-3-319-18301-5

Pavel Kameník. Příkazový Řádek v Linuxu – praktická řešení, Computer Press, 2011, ISBN:978-80-251-2819-0

Peter Kim. Hacking – praktický průvodce penetračním testováním, Zoner Press, 2015, ISBN:9788074133138

Sherri Davidoff, Johnathan Ham. Network Forensis – tracking hackers through cyberspace, Prentice Hall, 2012, ISBN-10: 0132564718

Předběžný termín obhajoby

2018/19 LS – PEF

Vedoucí práce

Ing. Alexandr Vasilenko, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 11. 9. 2018

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 19. 10. 2018

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 26. 03. 2019

Čestné prohlášení

Prohlašuji, že svou diplomovou práci "Forenzní analýza chování útočníka pomocí nástroje Honeypot" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené diplomové práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 27.3 2019 _____

Poděkování

Rád bych touto cestou poděkoval Ing. Alexandru Vasilenkovi Ph.D za vedení této práce a za odborné rady.

Forenzní analýza chování útočníka pomocí nástroje Honeypot

Abstrakt

Tato diplomová práce je zaměřena na forenzní analýzu chování útočníka v napadeném systému. Aby mohla být tato analýza úspěšně aplikována, byl navržen systém vysokointeraktivních honeypotů. Celý systém byl implementován v experimentálním segmentu domácí sítě pomocí klíčového nástroje – Honeywall. Pomocí tohoto nástroje bylo možné sledovat a analyzovat jednotlivé pokusy o navázání spojení s honeypoty. Druhým klíčovým prvkem navrženého systému byl nástroj Sebek, díky kterému bylo možné zachytit jednotlivé kroky, které útočník v napadeném systému učinil.

V této práci je podrobně popsán implementovaný systém spolu s dílčími nástroji, které byly v systému nasazeny. Výsledkem práce je již samotná forenzní analýza, jenž je demonstrována na jednom z mnoha útoků, které byly vůči honeypotu vedeny. Forenzní analýza odhaluje jednotlivé kroky, které útočník po napadení systému učinil, popisuje nástroje, které byly při útoku využity, a analyzuje síťovou komunikaci vztahující se k útoku.

Klíčová slova: Honeywall, Honeypot, Roo, Hacking, Honeynet, Backdoor, Rootkit, Exploit

Forensic analysis of the attacker´s behaviour using Honeypot tool

Abstract

This diploma thesis is focused on forensic analysis of the attacker's behavior in the infected system. In order to successfully apply this analysis, a high interactive honeypot system has been designed. The entire system was implemented in the experimental home network segment using the Honeywall key tool. With this tool, it was possible to track and analyze individual attempts to establish a honeypot connection. The second key element of the proposed system was the Sebek tool, which enabled it to capture the individual steps the attacker had taken in the attacked system.

In this work, the implemented system is described in detail together with the sub-tool that was deployed in the system. The result of the thesis is the forensic analysis itself, which is demonstrated in one of the many attacks, that was targeted against the honeypot. Forensic analysis reveals the individual steps the attacker has taken after attacking the system, describing the tools used during the attack, and analyzing the network communication related to the attack.

Keywords: Honeywall, Honeypot, Roo, Hacking, Honeynet, Backdoor, Rootkit, Exploit

Obsah

1 Úvod

V 21. století jsme svědky obrovského rozmachu technologické platformy – internet. Od vojenského projektu ARPANET[1] uplynulo již 50 let a internet se rozšířil od univerzitních počítačů přes osobní počítače, tablety až do telefonů a chytrých hodinek. Lidé si již navykli využívat tuto multifunkční platformu v každodenním životě – dle statistických údajů dostupných ze serveru internetworldstats.com (1), se v roce 2018 množství uživatelů internetu na území Evropy pohybovalo na 85% z celkového obyvatelstva Evropy. Ve Spojených státech amerických se dokonce množství lidí využívající internet vyšplhalo na 95% z celkového množství obyvatelstva. Dnes je již zcela normální provádět obchodní transakce, využívat internetového bankovnictví, či provádět obchodní činnosti v prostředí e-business. Všechny tyto činnosti mají jedno společné – osobní data, která uživatelé posílají prostřednictvím internetu. Na tato osobní data jsou právě cíleny útoky s cílem získání citlivých informací, ať už se jedná o přístupové údaje do internetového bankovnictví, e-mailové schránky či osobní údaje o dané osobě. S rozmachem nových technologií vznikají i nové postupy a techniky, jak tyto technologie zneužít a využít v prospěch třetích osob.

V druhé polovině roku 2017 bylo dle serveru statista.com (2) napadeno 98% organizací po celém světě útokem typu malware. Každá organizace, ať už se jedná o soukromou firmu či vládní instituci, by měla dbát na dodržování bezpečnostních zásad a pravidel tak, aby se mohla efektivně bránit těmto útokům a především takovýmto útokům předcházet. Malware přitom není jediný typ útoku, se kterým organizace přicházejí do kontaktu. Ve stejné statistice bylo uvedeno, že 69% organizací bylo konfrontováno útokem typu phishing a social engeneering. Tyto dva typy útoků jsou cíleny především na jednotlivce a snaží se na základě nevědomosti a nepozornosti daného uživatele, získat citlivá data. Ze statistik je patrné, že by organizace neměly dbát jen na zabezpečení samotných dat a vnitřní síťové infrastruktury, ale také dbát na vzdělávání svých zaměstnanců v této oblasti. Dobře zabezpečená organizace tedy neznamená jen zabezpečení dat a síťové infrastruktury, ale také dobře proškolení zaměstnanci, kteří dokáží například rozpoznat podvodný email, či podezřelý telefonát.

Při implementaci bezpečnostních prvků v zabezpečení síťové infrastruktury můžeme využít mnoha nástrojů, jako jsou například firewall, síťová sonda, anti-spam. Každý bezpečnostní pracovník má k dispozici nástroje, které mu pomáhají především v

[1] ARPANET - Počítačová síť spuštěna v roce 1969 – předchůdce dnešního internetu

monitorování síťového provozu, detekci podezřelých síťových toků a vyhodnocování bezpečnostních incidentů. Mezi takovéto nástroje patří i takzvané „honeypoty“ neboli síťové pasti. Jsou to aktivní síťová zařízení – nejčastěji počítače, ale také třeba routery či switche, které se umístí do prostředí počítačové sítě a pasivně čekají na to, až se s nimi někdo pokusí navázat spojení. Vzhledem k tomu, že tyto honeypoty nejsou určeny k jakékoliv síťové komunikaci, je každý pokus o připojení vyhodnocen jako podezřelý. Bezpečnostní pracovník může tedy velmi rychle zareagovat na vzniklou událost.

Honeypoty nacházejí uplatnění i v jiných oblastech bezpečnosti. Jsou to vhodné nástroje k odchytávání takzvaných zero-day exploit, kdy se po napadení systému celý počítač odstaví od síťového připojení, a škodlivý kód je zde podrobně zkoumán a analyzován. Na základě takovýchto analýz vznikají bezpečností update, které záplatují systém proti takovýmto hrozbám.

Honeypoty mohou také sloužit k psychologicko-sociální analýze crackerů, kteří se do systémů nabourávají. Tento typ honeypotu se po napadení systému neodstaví, ale místo toho se nechává útoku volný průběh. Poté je provedena analýza logů, a na základě takovéto analýzy jsou vyhodnoceny patřičné závěry. Můžeme tedy odhalit, za jakým účelem cracker systém naboural, jaké nástroje při útoku použil a jak při útoku postupoval. Takto získané informace nám pomohou odhalit myšlenkové procesy, které se odehrávají u těchto osob a dokážeme se tak daným útokům efektivně bránit, či zcela zamezit útoku.

2 Cíl práce a metodika

2.1 Cíl práce

Cílem této práce je implementace nástroje Honeywall, pomocí kterého bude provedena analýza chování útočníka po připojení do napadeného systému. Dílčím cílem této práce je shrnout a analyzovat možné typy útoků na informační systémy.

2.2 Metodika

Metodika řešené práce je založena na studiu a analýze odborné literatury. Praktická část práce je realizována formou návrhu modelu informačního systému, který je založen na nástroji Honeywall. Model bude zapojen po dostatečně dlouhou dobu tak, aby byla sesbírána data pro následnou analýzu útoku. Na základě syntézy teoretických a praktických poznatků budou vydedukovány závěrečné výsledky.

3 Teoretická východiska

Teoretická část práce je zaměřena na analýzu a popis druhů útoků na informační systémy. Tato část je zaměřena především na druhy škodlivých kódů takzvaný malware. Jsou zde popsány metody sociálního inženýrství – phishing, spare phishing, phone phishing a biting. Dále je popsán malware typu počítačoví virus, počítačový červ, trojský kůň ransomware, spyware, backdoor a rootkit. Také jsou zde popsány útoky proti webovým aplikacím, útoky typu DoS a DDoS a také útoky hrubou silou vůči autentizačnímu mechanismu. V další části jsou poté popisovány bezpečnostní nástroje – honeypoty. Jsou zde popsány typy honeypotů dle míry jejich interakce a dle účelu, ke kterému mohou být využity. Dále jsou zde popsány systémy, ve kterých je zapojeno více vysokointeraktivních honeypotů tak, že vytváří ucelený síťový segment, takzvaný honeynet. Jsou zde popsány generace GENI, GENII a GENIII honeynetů a dále jsou zde popsány možnosti, jakými lze honeypoty logovat.

3.1 Kybernetický útok

Kybernetický útok je dle oxfordského slovníku definován jako „*snaha hackerů poškodit nebo zničit počítačovou síť nebo systém*“ (3). Tato definice však vzhledem k sofistikovanosti informačních systémů vykazuje určité nepřesnosti. Dle dokumentu *National information assurance* je kybernetický útok definován jako „*útok prostřednictvím kyberprostoru cílený na podnikovou síť za účelem narušení, oslabení, zničení nebo zlomyslné kontroly počítačové infrastruktury, nebo zničení integrity dat či krádež informací*“ (4). Jak je vidět, definice kybernetického útoku není zcela jednoznačná. Význam tohoto pojmu se liší i napříč mezi vědními obory – orgány vymáhající právo považují kybernetický útok za zločin, armádní složky ho chápou jako potencionální válečný akt a technolog v něm vidí zneužití dostupných informačních zdrojů (5). Pro správné pochopení tohoto pojmu se musíme na tuto problematiku podívat i z hlediska prizmatu práva. Výklad pojmu kybernetický útok je totiž úzce spojen s legislativou každého státu, proto může být kybernetický útok chápan více způsoby, vždy v závislosti na příslušných jurisdikcích. Proto činnost, která je v jedné zemi trestná, nemusí být v druhé zemi vůbec zahrnuta do legislativy nebo může být legální. Pohled skrze legislativu ovšem přináší další zásadní problém – neustále se měnící definice

kybernality[2], která se mění s vývojem informačních technologií a změnami charakteru informačních systémů (6).

Dle výše popsaných závislostí mezi legislativou, kybernalitou a definicemi kybernetických útoků lze vyvodit, že jednoznačná a obecná definice kybernetického útoku je velmi obtížně aplikována. Pro povahu této práce budeme chápat, že kybernetický útok je „*jakýkoli útočný manévr vůči informačnímu systému, infrastruktuře, počítačové síti či osobnímu počítači a zařízení. Kybernetický útok může být veden národními státy, jednotlivci, skupinami či organizacemi, nebo může být veden z anonymního zdroje. Prostřednictvím kybernetického útoku mohou být zcizena či zničena určitá data, která jsou považována za cíl útoku*“ (7).

Obecně lze rozdělit kybernetické útoky na aktivní a pasivní. Při pasivním útoku se útočník zaměřuje na odposlouchávání komunikace, nebo zaznamenávání síťové aktivity. Útočník do komunikace nezasahuje, proto je velmi obtížné takovýto typ útoku odhalit. Jedinou obranou proti těmto typům útoků je použití šifrované komunikace.

Při aktivním útoku se již útočník aktivně podílí a snaží se změnit informaci, či vytvořit novou falešnou zprávu, získat neautorizovaný přístup či odcizit data. Tyto útoky vyžadují větší úsilí a ve většině případech se o nich oběť útoku dozví (8). V následujících podkapitolách budou popsány jednotlivé typy útoků na informační systémy.

3.1.1 Sociální inženýrství

Při zajišťování celkové bezpečnosti organizace, je člověk, aniž by si to sám uvědomoval mnohdy nejslabším článkem v tomto řetězci bezpečnosti. Sociální inženýrství představuje metody, jak podvodně manipulovat s lidmi a přimět je ke sdělení určité informace či k provedení určité činnosti (9). Tyto metody jsou založeny na principech lidského rozhodování, známých jako kognitivní chyby úsudku[3] (10). Metody sociálního inženýrství jsou:

- *Phishing* – je to metoda podvodného získávání osobních informací. Oběť phishingu obdrží e-mail, který se jeví zcela legitimně. Může se jednat o e-mail

[2] Kybernalita – kybernetická kriminalita, páchání trestné činnosti, v níž je aktivní složkou informační technologie jako souhrn technického a programového vybavení včetně dat

[3] Kognitivní chyba úsudku – je systematická, opakovaná chyba v myšlení, rozhodování, odhadech, vzpomínkách, zapamatování a jiných myšlenkových procesech, přičemž závěry o jiných lidech a situacích mohou v těchto případech být vyvozeny nelogickým způsobem

z banky, sociálních sítí, aukčních webů a podobně. Tyto e-maily vyzývají oběť k zadání některých osobních informací, například čísla kreditní karty, hesla k sociální síti či hesla a ID k internetovému bankovnictví. Pokud jsou tyto osobní informace odeslány, útočníci získají přístup k dané agendě (11).

- *Spear phishing* – je druh phishingu, ve kterém útočník odesílá e-maily, které jsou vysoce přizpůsobené konkrétnímu uživateli, organizaci či podniku. Často za účelem krádeže dat, či k instalaci malware na cílený počítač (12). Úspěšnost tohoto druhu phishingu je závislá na množství a kvalitě takzvané open source inteligence, což jsou data sbíraná z veřejně dostupných zdrojů, například ze sociálních sítí.
- *Phone phishing* – je druh sociálního inženýrství který využívá telefonní systémy. Útočník může využít tzv. interactive voice response, který bude konfigurován obdobně jako IVR využívané například u bankovních společností. Metodou klasického phishingu rozešle e-maily ve kterých uvede číslo na podvodný IRV s prosbou o zavolání na toto číslo pro kontrolu údajů. Pokud oběť na toto číslo zavolá, je vyzvána ke sdělení osobních informací (13).
- *Baiting* – tato metoda využívá fyzické médium, například CD nebo USB disk na kterém je malware. Útočník toto médium ponechá na místě, kde ho bude snadné nalézt, například na toaletách, ve výtahu či chodbě v budově dané společnosti. Pro vyšší autenticitu může útočník vyobrazit na daném médiu logo společnosti, či vytvořit obal s popisem média. Poté již stačí čekat, až některý zvědavý zaměstnanec vloží médium do svého počítače a následně dojde k infekci systému (14).

3.1.2 Malware

Slovo malware je složeno ze dvou slov, slova *malicious* a *software*, což lze volně přeložit jako škodlivý software. Pod tímto slovem je označován veškerý software, který je nějakým způsobem nebezpečný uživateli. Tento software může být naprogramován například k účelům odcizení, zašifrování či smazání osobních dat či k monitorování uživatelské aktivity (15). Malware se může šířit v podobě spustitelných kódů, scriptů, aktivního obsahu na webových stránkách či pomocí jiného software který si uživatel nainstaluje na svém PC (16). Kódy malware mají podobu především jako:

počítačové viry, červi, trojské koně, ransomware, spyware, adware, scareware, rootkit a backdoor.

Počítačový virus

Počítačový virus je typ malware, který se dokáže šířit z jednoho počítače na druhý bez vědomí uživatele a dokáže se replikovat kopírováním sebe sama do jiného programu či dokumentu (17). Takový program se tedy chová obdobně jako biologický virus, který se šíří vkládáním svého kódu do živých buněk. V souladu s touto analogií se procesu šíření počítačového viru říká nakažení či infekce a napadenému souboru hostitel. Jako hostitel můžou například posloužit spustitelné soubory, systémové oblasti disku, popřípadě soubory, které nelze vykonat přímo, ale za použití specifických aplikací (dokumenty MS Wordu apod.). Pokud je tento hostitel spuštěn, spustí se i kód viru. V tomto okamžiku se virus snaží zajistit další sebe replikaci, a to připojením k dalším vhodným hostitelům (18).

Aby byl počítačový virus schopen provádět výše zmíněnou aktivitu, musí obsahovat určité funkcionality. První funkcionalitou je takzvaný mechanismus infekce. Touto funkcionalitou se virus rozšiřuje – vyhledává v již nakaženém systému další soubory, dokumenty či diskové oddíly k nakažení (19). Další funkcionalitou, kterou virus disponuje je takzvaný spouštěč (trigger), který určuje, kdy se nebezpečný kód viru aktivuje nebo deaktivuje. Může to být konkrétní datum, čas, přítomnost jiného programu či v momentě, kdy kapacita disku dosáhne určité hodnoty (20). Poslední funkcionalitou je samotný škodlivý kód, který se v daný okamžik spustí. Tato funkcionalita se nazývá *payload*, neboli náklad. Po spuštění payload se provede zákeřný kód, ke kterému byl virus naprogramován (19).

Životní cyklus počítačového viru je rozdělen do čtyř fází. V první fázi je virus nečinný, tato fáze se nazývá fáze spánku. Viru se podařilo získat přístup k počítači či software cílového uživatele, prozatím virus „spí“ a čeká až bude aktivován spouštěčem. Tuto fází nemusí mít všechny viry (19). V druhé fázi se virus začne šířit, vytváří kopie sebe sama a kopíruje je do ostatních programů, souborů či do systémových oblastí na disku. Kopie viru nemusí být shodná s originálem, virus se při replikaci může měnit tak, aby se vyhnul odhalení antivirových programů. V následující fázi již virus čeká na spuštění, po kterém vykoná funkci, pro kterou byl určen. Poslední fází životního cyklu je již samotné provedení

payload – tedy provedení zákeřného kódu. To může mít fatální následek pro nakažený systém, záleží na účelu viru (19).

Počítačový červ

Jedná se o typ malware, jehož primární funkcí je rozšiřovat sebe sama v prostředí počítačové sítě a infikovat další počítače, zatímco zůstává aktivní na již nakažených systémech. Počítačoví červi často využívají automatizované části systému, které jsou skryty před uživatelem (21). Klasičtí červi jsou programovány pouze k tomu, aby rozšiřovali sebe sama na ostatní systémy. Takovíto červi nemusí nakaženému systému způsobit žádnou škodu, nanejvýš zpomalení síťového provozu využíváním síťových prostředků (22).

Červi však mohou být i vysoce škodliví, a to v případě, pokud červ obsahuje „náklad“ neboli payload. Takový červ může na hostitelském PC mazat či šifrovat soubory nebo krást citlivá data. Nejvíce rozšířený payload, který může červ přenášet je instalace backdoor, neboli zadních vrátek. Pokud se tak stane, může autor červa získat vzdálený přístup k nakaženým počítačům a ty poté použit na další typy útoků (21).

Trojský kůň

Trojský kůň je program, který se na první pohled jeví jako neškodný, avšak obsahuje škodlivý payload. Na rozdíl od viru či červu, se trojský kůň neumí replikovat a rozšiřovat sebe sama na další systémy – z tohoto důvodů musí útočníci pomocí sociálního inženýrství přimět uživatele, aby si trojského koně stáhli do svého PC (23).

Payload, který s sebou trojský kůň nese je zaměřen na několik oblastí. Především se jedná o password stealing trojany, kteří mají v sobě implementovaný nějaký druh keylogeru který zaznamenává klávesy, které uživatel stiskl a tyto záznamy posílá útočníkovi, který trojského koně nasadil. (18). Dalším typem jsou destruktivní trojani, kteří po spuštění mažou soubory, či zformátují celý disk. Trojští koně také mohou obsahovat backdoor, který zajistí útočníkovi vzdálenou kontrolu počítače. Trojský kůň typu downloader stahuje bez vědomí uživatele další malware z předem daných URL adres. Trojský kůň může být využit i jako takzvaná trojan proxy, kdy se z nakaženého PC stane anonymizační proxy pro útočníka, který může skrze tuto proxy provádět nelegální činnosti na internetu a stopy povedou pouze k napadenému PC (18).

Ransomware

Ransomware jakožto složenina ze slov *ransom* a *software* označuje typ malware, který lze volně přeložit jako vyděračský software. Tento typ malware je uzpůsoben k tomu, aby bránil uživateli k přístupu do počítače, či k datům a následně požadoval výkupné které musí uživatel zaplatit, aby znovu získal přístup ke svým datům. Jednoduché typy ransomware pouze uzamknou počítač, do kterého se nedá přihlásit. Sofistikovanější druhy dokáží zašifrovat veškerá data na pevném disku (24). Ransomware se typicky šíří pomocí sociálního inženýrství, především pomocí taktik phisingu a exploitací ve zranitelnostech software (25). Výkupné je požadováno především v některém druhu kryptoměny, tudíž totožnost pachatelů je nezjistitelná, vzhledem k povaze kryptoměn (24).

Koncept šifrovacího ransomware je nazýván cryptoviral extortion a samotný protokol je rozvržen do třech částí:

- Útočník vygeneruje pár klíčů – veřejný a soukromý. Veřejnou část klíče umístí do ransomware, který je následně vypuštěn k oběti
- Poté, co se ransomware dostane do počítače oběti, je vygenerován náhodný symetrický klíč pomocí kterého jsou zašifrována uživatelská data. Pomocí veřejného klíče je zašifrován i symetrický klíč. Výstup z této hybridní enkrypce je asymetrický ciphertext a symetrický ciphertext uživatelských dat. Tím je vynulován symetrický klíč. Poté je uživateli předložena zpráva, která obsahuje asymetrický ciphertext a pokyny k zaplacení výkupného. Oběť útoku pošle asymetrický ciphertext a požadovanou částku v kryptoměně útočníkovi.
- Útočník obdrží výkupné, dešifruje asymetrický ciphertext pomocí svého soukromého klíče a poté pošle symetrický klíč oběti. Oběť následně rozšifruje svá data pomocí poskytnutého symetrického klíče (26).

Spyware

Jedná se o typ malware, který je zaměřen na špionáž – z anglického slova *spy* a *malware*. Tento malware je designován tak, aby shromažďoval data z počítače, aniž by o tom uživatel věděl a tato data dále posílal útočníkovi, který spyware nasadil. Spyware může shromažďovat informace o heslech, číslech kreditních karet, zaznamenávat veškeré zmáčknuté klávesy které uživatel učinil, sledovat pohyb uživatele po webu nebo sbírat

emailové adresy (27). Nasbíraná data mohou být také kombinována s daty z jiných databází, a tak mohou být vytvořeny profily jednotlivých uživatelů, rodin, pracovních skupin či celých organizací. Takto vytvořené profily jsou primárně využívané k marketingovým účelům (28).

Nakažení spyware se nejčastěji odehrává pomocí metod sociálního inženýrství, především pomocí phishingu, kdy se útočník snaží donutit uživatele ke stažení souboru, či ke kliknutí na odkaz, skrze který se spyware stáhne. Spyware se také může šířit jako trojský kůň, kdy je schován jako „payload" v legitimně vypadajícím software (29).

Adware

Tento typ malware slouží k šíření reklamních kampaní. Z anglického *advertising-supported software*, volně přeložený jako reklamní software. Tento software se projevuje zobrazováním reklam různého rozsahu, od bannerů v pracovním prostředí programu, až po neustále vyskakující okna, či změně domovské stránky v internetovém prohlížeči. Určitou formu adware obsahují volně šiřitelné programy, které pomocí reklamní kampaně zajišťují určitou peněžní návratnost vývojářům (30). Adware může být i mnohem nebezpečnější, než se na první pohled může zdát. Mnoho adware totiž sbírá uživatelská data, která jsou dále využívány k cílené reklamě (31). Takovýto typ adware stojí na pomezí mezi adware a spyware. Pokud adware sbírá osobní uživatelská data bez vědomí uživatele, jedná se o spyware, ale pokud uživatel se sběrem dat souhlasil a ví, jaká konkrétní data jsou sbíraná, nejedná se o nelegitimní malware ale o adware. Adware je nejčastěji součástí freeware a shareware programů. Pokud se jedná o nelegitimní adware, ten se do počítače nejčastěji dostane z infikované webové stránky, v takovém případě je adware injektován do počítače pomocí zranitelnosti webového prohlížeče (31).

Scareware

Scareware představuje formu malware která je zaměřena na zastrašení a vydírání uživatele. Využívá metod sociálního inženýrství, a pomocí manipulace nutí uživatele k nákupu nevyžádaného software. Toho je docíleno tím, že se scareware snaží uživatele přesvědčit o tom, že je jejich počítač nakažen některým z mnoha druhů malware, a nutí uživatele ke stažení a zaplacení falešného anti-virového programu. Obvykle je tento software

nefunkční a zpráva o infekci počítače je fiktivní, někdy může být sám nainstalovaný antivirový software malwarem (6).

Rootkit

Rootkit je nástroj který umožňuje útočníkům získat administrátorská nebo root práva a zamaskovat útočníkovu aktivitu v napadeném systému. Aby mohl rootkit zamaskovat útočníkovu aktivitu, modifikuje způsob, jakým uživatelský program přijímá informace od operačního systému. Rootkity tedy modifikují procesy nebo modifikují systém tak, aby maskoval jejich činnost (32).

První a nejjednodušší rootkity dělaly to, že modifikovaly systémové utility jako například utilitu *ls*[4], tak aby změnila funkcionalitu a při použití této utility uživatelem, nezobrazila určité soubory, které útočník definoval. Rootkity se dělí na *user mode rootkit* a *kernel mode rootkit,* dle toho, jakým způsobem rootkity ovlivňují činnost systému. V případě user mode rootkitu, je činnost zaměřena pouze na jeden samostatný proces, který je spuštěn uživatelem a rootkit tento proces skryje. Kernel mode rootkity jsou zaměřeny na celý systém a dokáží skrýt informace od všech procesů.

Backdoor

Backdoor je metoda, kterou se mnohdy skrytě obchází autentikační mechanismus v počítačovém systému ale i na úrovni hardware – například v chipsetu. Backdoor má formu skryté části (funkcionality) programu, či kódu ve firmware hardwaru, nebo může být i na úrovni operačního systému – takovouto backdoor funkcionalitou disponuje systém Widnows a to již od verze XP a Vista. Systémy Windows 7 a 8 využívaly stejnou metodu a v systému Windows 10 již tato funkcionalita není ani skryta (33). Takovéto druhy backdoor mají své legitimní využití, například výrobcem prováděný vzdálený servis zařízení. Backdoor může být také využit k získání hesel v zařízení, poškození dat na úložišti či k odeslání informací o uživatelově aktivitě. Funkcionalita backdoor je využívaná jak k legitimním činnostem, tak k nelegitimním. Je zneužívána blackhat komunitou, výrobci mobilních telefonů, software developerů v oblasti proprietárního sofware či vládními organizacemi – například organizace NSA vkládá své backdoor do produktů Cisco (34).

[4] Utilita ls slouží k vypsání obsahu adresáře v systémech GNU/linux

3.1.3 Útoky proti webovým aplikacím

Útoky proti webovým aplikacím jsou oblíbené z toho důvodu, že prostřednictvím webových aplikací lze přistupovat do zajímavých a útočníky velmi lákavých oblastí – například do aplikací internetového bankovnictví, do databází tajných vládních organizací či k soukromým emailům. Útoky, které budou v této podkapitole představeny jsou útoky typu Cross-site scripting a SQL injection. Zranitelnosti ve webových aplikací vznikají již v samotném procesu tvorby, kdy nejsou programátory dodržovány bezpečnostní opatření, či jsou opomenuta. Pokud je například validace vstupů u přihlašovacího pole do aplikace špatně naimplementována, bude obsahovat zranitelnost, kterou dokáže útočník zmapovat, zneužít a získá neautorizovaný přístup do aplikace.

Cross-site scripting

Pokud validace vstupů webové aplikace umožní spustit útočníkovy škodlivý script, se stejnou úrovní přístupu jako má kód legitimní, jedná se o útoku typu cross-site scripting. Tento typ útoku je uživatelský – je cílen tedy na uživatele dané aplikace. Útočník dokáže spustit program napsaný v javascriptu v prohlížeči oběti útoku. Útočník například dokáže získávat informace z prohlížeče oběti vztahující se ke zranitelné stránce, nebo, pokud tuto zranitelnost obsahuje aplikace webového bankovnictví, dokáže útočník manipulovat s účtem oběti. Útočník může také získat autentizační cookie ke zranitelné stránce pomocí které se později může přihlásit. Takovéto útoky se nazývají jako *reflektivní cross-site scripting* neboť pozměněnou webovou stránku uživateli podstrčíme pomocí falešné URL a pozměněná webová stránka běží z našeho serveru – tím pádem ovlivní pouze jednoho uživatele. Druhý typ útoků je takzvaný *persistentní cross-site scripting* který je velmi nebezpečný z toho důvodu, že se útočníkovi podaří infikovat a pozměnit obsah webové stránky která je uložena v databázi na legitimním serveru. Takováto změna stránky poté ovlivní jakéhokoli uživatele, který se ke stránce připojí. Například pokud je tato zranitelnost přítomna na webové stránce, jejíž součástí je sekce pro komentáře, kdy každý komentář je uložen v databázi a může být různými uživateli zobrazován. Pokud nejsou vstupy řádně ošetřeny, může útočník vložit prostřednictvím těchto vstupů nebezpečný kód, který následně bude spouštěn v prohlížeči každého uživatele, který stránku navštíví (35).

SQL Injection

K tomuto útoku se využívá zranitelnost webových aplikací, která vychází z nedostatečných nebo žádných validací vstupů. Pokud aplikace, která komunikuje s databází a odesílaná data do databáze nijak nekontroluje, může dojít k tomu, že bude do databáze odeslán škodlivý SQL kód, který se v databázi provede a může mít devastující následky, například útočník může získat z databáze citlivá data, změnit obsah webové stránky či danou databázi úplně smazat.

SQL je dotazovací jazyk, který umožňuje programům komunikovat s relačními databázemi. Jazyk funguje na základě dotazů, které jsou databázovým systémem vykonávány a jsou vraceny výsledky dotazů.

Pokud vstupy aplikace nebudou řádně ošetřeny, nebude aplikace vědět jak má se zadaným SQL dotazem ve vstupu pracovat. Dojde tedy k tomu, že se vstupem aplikace nebude pracovat jako s daty, ale jako s legitimním SQL dotazem. Toho lze snadno zneužít (35). Na následujícím obrázku č.1 je uveden příklad. Text v šedém rámečku zobrazuje SQL

Obrázek 1 - SQL Injection - příklad vložení SQL kódu do neošetřeného vstupu (35)

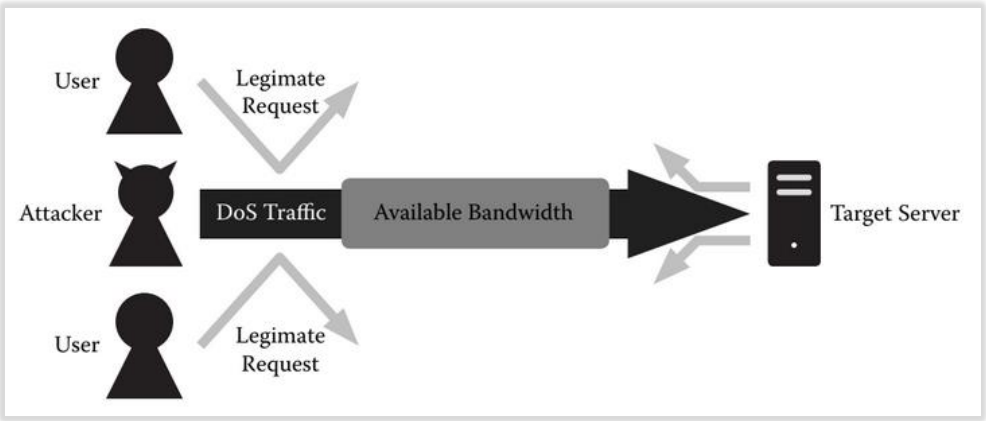
kód, kterým je konstruovaný příjem vstupu do databáze. Text v černém rámečku zobrazuje vstup, který může uživatel do pole zapsat. Na řádku č.1 uživatel vložil jako vstup user1, což je legitimní vstup, nedojde k žádnému ohrožení databáze. Na řádku č.2, je zobrazen vstup který obsahuje SQL kód, který může ovlivnit databázi. Zmíněný SQL kód by mohl být vykonán v tom případě, pokud by v databázi existoval uživatel *x*. Pokud by takovýto uživatel existoval, provedl by se zbytek kódu, který by vypsal sloupce *uname* a *pwd* z relace *users.*

3.1.4 DoS a DDoS útoky

Útoky typu DoS – *denial of service* jsou útoky, které jsou mířeny na systémy, které jsou připojeny do Internetu a jsou veřejně přístupné – především služby e-commerce, finanční a vládní instituce apod. Servery těchto organizací nabízejí určité služby – například webové stránky či FTP servery, s kterými můžou klienti komunikovat. Pokud nastane situace, že daný server či služba nemůže odpovědět na dotazy od klientů, z důvodu přetížení ze strany

útočníka, mluvíme o útoku typu DoS, čili odepření služby. Daná webová stránka se vlivem přetížení stane nedostupnou, čehož lze zneužít například v oblasti e-commerce, kdy jeden subjekt může vyřadit konkurenta.

Server poskytující určité služby může v každém okamžiku odesílat a přijímat jen určité množství dat. Všechna data, která jsou odeslána nad tento rámec, se již ke svému cíli určení nedostanou. Pokud je tedy síť zahlcena, a uživatel odešle na server dotaz, který je již nad rámec této propustnosti, vrátí se mu chybová stavová stránka. Aby útočník dosáhl takovéhoto zahlcení dané služby, musí na tuto službu odesílat velké množství dotazů. Na obrázku č.2 je zobrazen příklad, kdy útočník odesílá na daný server takové množství dotazů,



Obrázek 2 - Útočník zahlcující server dotazy (35)

že je server zahlcen a již není schopen odpovídat na požadavky od legitimních uživatelů. Aby mohl útočník dosáhnout takového množství dotazů, využívá technik jako například *flooding*. Další metodou, jak zajistit DoS je zahlcení systémových prostředků serveru. Příkladem může být zahlcení CPU, či operační paměti, které vede k selhání systému. Další metodou jak dosáhnout tohoto útoku je metoda DDoS. Tedy velké množství distribuovaných systémů spojí síly, a začnou společně zasílat požadavky na danou službu. Pro tento typ DDoS útoku se využívají *botnety.*

3.1.5 Útoky hrubou silou a slovníkové útoky

Tyto útoky cílí na autentizační systémy, které pomocí hesla ověřují uživatelovu identitu. Každý takovýto autentizační systém je silný pouze tak, jak silné je uživatelské heslo k danému účtu. Tyto systémy jsou však náchylné především ke dvěma typům útoků – útoku hrubou silou a slovníkovému útoku. Autentizační systémy využívají k ověření uživatele metodu porovnání aktuálně zadávaného hesla s heslem, které bylo při zakládání účtu

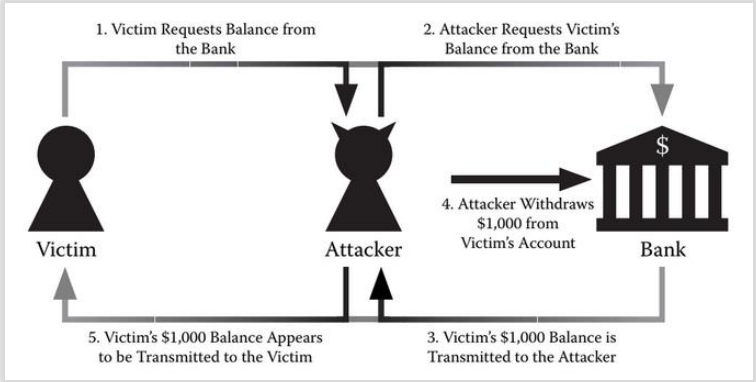
deklarováno. Tato hesla jsou ukládána v databázi v podobě takzvaných *hashů* tedy v zašifrované podobě. V takovéto podobě by měla být hesla skladována vždy, neboť z hlediska bezpečnosti je nevhodné, aby hesla k účtům byla uložena v databázi ve formě prostého textu.

Slovníkový útok je takový typ útoku, kdy útočník využívá textové soubory obsahující množství hesel (například nejčastěji používaná hesla, hesla v různých světových jazycích a podobně) a textové soubory, které obsahují množství uživatelských jmen. Při samotném útoku jsou zkoušeny kombinace z těchto dvou textových souborů, nebo, pokud je uživatelské jméno známo, využívá se pouze textový soubor s hesly vůči tomuto známému uživatelskému jménu. Takovéto hádání může být časově náročný proces a autentizační systémy mohou využívat některou z metod ochrany proti takovýmto útokům – například možnost zadání hesla u stejného uživatele pouze několikrát a poté dojde na určitou dobu k odmlčení. Pokud ale útočník získá databázi s hesly v *hash* formátu, dokáže takovýto útok provést offline na svém počítači. Pomocí hashovacího algoritmu, který musí být totožný s algoritmem, který byl použit k hashi hesel v databázi, zašifruje textový soubor s hesly a porovnává již samotné hashe. Také může použít takzvané *rainbow* tabulky které obsahují již zašifrovaná hesla. (32).

Útoky hrubou silou dokáží prolomit jakékoliv heslo, avšak náročnost na výpočet se dramaticky zvyšuje s délkou hesla a s jeho složitostí. Pokud je heslo dostatečně dlouhé, obsahuje speciální znaky a čísla, výpočet může trvat v řádu stovek let. Útok hrubou silou využívá takzvaný *set* což je soubor, který obsahuje pravidla pro vypočítávané heslo. Tyto pravidla určují, kolik znaků bude mít vypočítávané heslo, a z jakých znaků se má heslo skládat. Následný útok tedy vyžaduje výpočet všech možných kombinací o určité délce znaků a každé vyhodnocené heslo se zkouší vůči autentizačnímu systému. Pokud má útočník k dispozici databázi hashovaných hesel, může útok provádět offline. Při tomto útoku musí být ale každé vyhodnocené heslo zašifrováno příslušným algoritmem a až následně může být porovnáváno vůči databázi hesel (35).

3.1.6 Man in the middle

Při tomto útoku útočník dokáže zachytit, zobrazit a pozměnit citlivá data, která jsou přenášena mezi dvěma subjekty. Na následujícím obrázku č.3 je zobrazeno schéma tohoto



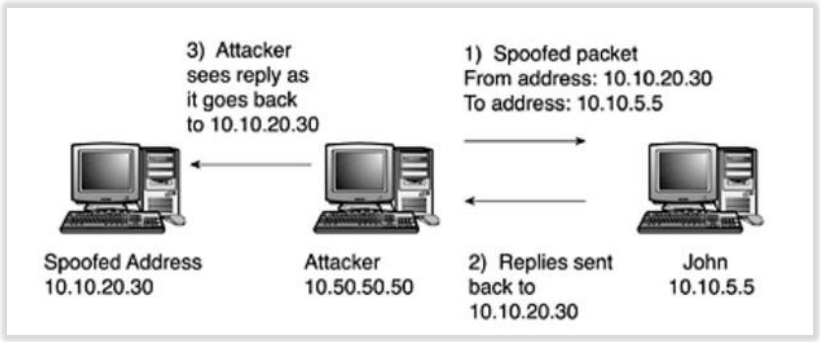
Obrázek 3 - Útok typu man in the middle (35)

typu útoku, při kterém útočník vykrade bankovní účet oběti. Útočník dokáže zachytávat komunikaci mezi těmito dvěma subjekty a může jí pozměnit. Přitom žádný z koncových subjektů neví, že jsou odesílaná data pozměňována a komunikace se jeví koncovým uživatelům legitimně. K útoku typu man in the middle může být zneužit protokol ARP – pokud útočník rozešle v určité síti falešný ARP paket který deklaruje, že je útočník vlastníkem všech IP adres v dané síti, může být následně komunikace směřována přes tyto adresy, při použití proxy pak může být komunikace mezi dvěma koncovými uzly směřována právě přes tuto proxy, a útočník tak může nahlížet do komunikace, aniž by si oběť něčeho všimla. K tomuto útoku lze zneužít i DNS protokol. Pokud útočník dokáže změnit překlad určité domény na jinou IP adresu, může oběť snadno začít komunikovat se serverem, který není legitimní, ale patří útočníkovi. Další metoda je ta, že útočník může nabízet veřejnou wifi síť která je nakonfigurována tak, aby umožňovala útočníkovi nahlížet do komunikace (35).

3.1.7 Spoofing

Spoofing je metoda, pomocí které útočník úspěšně zamaskuje svoji identitu a vydává se za jiného, mnohdy legitimního uživatele. Spoofing může být na úrovni protokolu IP – tedy IP spoofing, na úrovni e-mailové komunikace – email spoofing dále na úrovni world wide webu – tedy web spoofing a spoofing může být i na netechnické úrovni, kdy útočník kompromituje lidské zdroje určité společnosti pomocí metod sociálního engeneringu (36).

Při IP spoofingu dochází k tomu, že útočník změní svoji IP adresu a tím pádem je útočníkova identita v prostředí počítačové sítě změněna. Na následujícím obrázku č.4 je zobrazen příklad, kdy útočník zaujal pozici mezi počítačem oběti, a počítačem, jejíž adresu

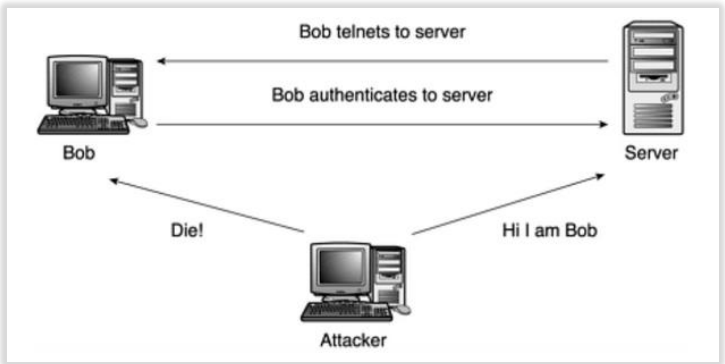


Obrázek 4 - Zachycení odpovědí od oběti útoku (36)

spoofinguje. V tomto případě dokáže útočník zachytit odpovědi, které jsou obětí odesílány na adresu, která je útočníkem podvržená. V prvním kroku je odeslán paket z adresy 10.10.20.30 na adresu oběti – 10.10.5.5. V následujícím kroku je odpověď odeslána od oběti na adresu 10.10.20.30 – tuto odpověď je útočník schopen zachytit.

3.1.8 Session hijacking

Při této metodě útoku je útočník schopen zmocnit se takzvané *session,* což je navázané a ověřené aktivní spojení mezi uživatelem a serverem, či aplikací. Hlavním účelem tohoto útoku je získání neautorizovaného přístupu tím, že útočník obejde autentizační systém. Autentizace uživatele je provedena pomocí jeho loginu a hesla. Jakmile je spojení navázáno, útočník vynutí odpojení uživatele a pomocí uživatelovi identity opět vstoupí do spojení, aniž by musel znovu zadávat přihlašovací údaje. Tento útok může být pasivní i aktivní. Při pasivním útoku útočník pouze odposlouchává *session* mezi uživatelem a serverem. Při



Obrázek 5 - Session hijack (36)

aktivním útoku se útočník zmocní této *session* a získá přístup do systému. Na obrázku č.5 je zobrazeno schéma tohoto útoku. Poté, co legitimní uživatel naváže spojení se serverem, útočník tohoto uživatele odpojí a pomocí uživatelovy identity se připojí k probíhající session.

3.2 Koncepce honeypotu

Hlavní účel honeypotu je ten, aby byl proti němu zahájen kybernetický útok jakéhokoliv druhu, jakýmkoliv způsobem, tak, aby bylo možné takový útok vzhledem k povaze a způsobu zapojení daného konkrétního systému zachytit a zaznamenat. Přitom nezáleží na tom, zda jde o implementaci honeypotu založenou na bázi routeru, skriptu emulujícího určitou službu či reálného operačního systému (37). Data, která honeypot nashromáždí, jsou velmi zajímavá, jedná se totiž o data, která zachycují ucelený kybernetický útok. V prostředí počítačové bezpečnosti je pod pojmem honeypot označována řada nástrojů, metod a mechanizmů, které Lence Spitzner definuje jako:

„*Honeypot je bezpečnostní prostředek, jehož hodnota spočívá v tom, být napadán, prozkoumáván, využíván, či jinak ohrožován* “ (37)

Honeypoty jsou vysoce flexibilní, modifikovatelné a přizpůsobivé nástroje, které mohou být využity v nejrůznějších oblastech počítačové bezpečnosti. Mohou sloužit jako další rozšiřující prvek v celkovém zabezpečení organizací, kde mohou sloužit jako nástroje, které mají za cíl včas detekovat kybernetický útok. Mohou být využity k analýze automatických útoků v podobě malware, či k odlákání potencionálních útočníků od skutečně cenných zdrojů. Mohou být také využity ve výzkumných oblastech počítačové bezpečnosti, kdy je možné pomocí honeypotů odhalovat nástroje a postupy, které blackhat komunita využívá ve svých průnicích do systémů (37). Za touto diverzitou se skrývají nástroje, pomocí kterých se dají implementovat jednotlivá řešení, dle konkrétního účelu.

3.2.1 Vysokointeraktivní honeypoty

V problematice honeypotů znamená míra interakce hloubku, do jaké je možno honeypot napadat, prozkoumávat či jinak zneužívat. Vysokointeraktivní honeypoty nabízejí

útočníkům kompletní systémy, které mohou být napadeny. To znamená, že honeypot neemuluje žádnou službu či funkcionalitu, ale poskytuje skutečný operační systém se službami a funkcionalitami stejným způsobem, jako je poskytují organizace v 21. století. Útočník tedy může kompletně kompromitovat daný systém a získat nad ním úplnou kontrolu. Díky zachycení takového útoku získáme data, díky kterým dokážeme zjistit motivy, taktiky, postupy a nástroje, které blackhat komunita využívá (38).

Vysokointeraktivní honeypoty vzhledem ke své povaze skýtají jistá rizika. Mezi největší rizika patří především to, že honeypot, který byl kompromitován, může útočníkovi posloužit jako nástroj pro další útoky na jiné systémy uvnitř sítě, k zachycení komunikace či může honeypot zneužít předem nepředvídatelným způsobem. Při implementaci vysokointeraktivních honeypotů je nutno brát tyto rizika v úvahu, a celý systém zapojit tak, aby byly tyto hrozby minimalizovány (37). Jedním z nástrojů, který toto riziko dokáže ošetřit, je nástroj Honeywall, který veškerou síťovou komunikaci, která směřuje ven z honeypotu, kontroluje, a do jisté míry zakazuje. Funkcionalita tohoto nástroje bude blíže popsána v následujících kapitolách.

3.2.2 Nízkointeraktivní honeypoty

Nízkointeraktivní honeypoty poskytují pouze velmi povrchní interakci mezi útočníkem a systémem, který je honeypotem představován. Takovýto honeypot dokáže emulovat operační systém spolu se službami, které jsou běžně dostupné v produkčních systémech. Tento systém je však emulován programem, který dokáže do jisté míry odpovídat na útočníkovy dotazy, nejedná se však o reálný systém, který by mohl být zneužit. Účelem nízkointeraktivních honeypotů není odhalování nových hrozeb, či zachycení uceleného kybernetického útoku. Jejich hodnota spočívá především v odhalování neautorizovaných síťových scanů nebo neautorizovaných pokusů o navázání spojení či v zaznamenávání již známých hrozeb. Nízkointeraktivní honeypoty dokáží poskytnout následující informace:

- Datum a čas útoku
- Zdrojová IP adresa a zdrojový port
- Cílová IP adresa a cílový port útoku

Pokud emulované služby dokáží poskytnout interakci mezi útočníkem a honeypotem, dokáží tyto honeypoty zachytit například i informace o tom, jakým loginem a heslem se útočník snaží prolomit autentizaci dané služby.

Výhodou nízkointeraktivních honeypotů je jejich malá hardware náročnost, můžeme tak implementovat stovky honeypotů na volných IP adresách, a to pouze na jednom fyzickém či virtuálním stroji. Nízkointeraktivní honeypoty přinášejí i výrazně nižší riziko oproti vysokointeraktivním honeypotům – útočník nemá možnost získání přístupu do reálného systému, z kterého by mohl vést další útoky (37). Následující tabulka č.1 zobrazuje srovnání mezi nízkointeraktivními honeypoty a vysokointeraktivními honeypoty. Z tabulky

Míra interakce	**Konfigurace**	**Zapojení & údržba**	**Informace o útoku**	**Riziko**
Nízká	Snadná	Snadné	Limitované	Nízké
Vysoká	Obtížná	Obtížné	Rozsáhlé	Vysoké

Tabulka 1 - Srovnání vysoko a nízkointeraktivních honeypotů - vlastní zpracování

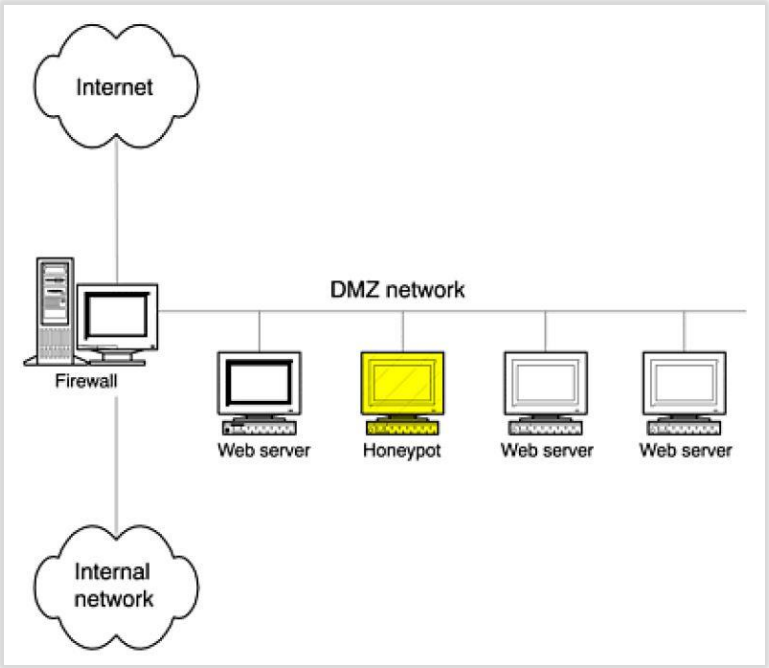
vyplývá, že nároky, které jsou při implementaci kladeny na nízkointeraktivní honeypoty, jsou mnohem nižší než nároky, které jsou kladeny na vysokointeraktnivní honeypoty. Díky této jednodušší implementaci však máme i mnohem omezenější přehled o útocích, které jsou vůči honeypotu vedeny.

3.2.3 Produkční honeypoty

Produkční honeypoty jsou nasazovány do prostředí počítačových sítí mnohých organizací za účelem včasné detekce kybernetického útoku a tím snížení rizik spojených s těmito útoky. Honeypoty jsou zde nasazovány jako další prostředky k detekci podezřelých síťových toků. Mimo honeypoty existují technologie, jako například IDS které slouží k monitorování počítačové sítě či systému za účelem odhalení škodlivé aktivity. IDS systémy však fungují na jiném principu nežli honeypoty – porovnávají síťové toky s databází, která obsahuje již známé typy podezřelých toků (39). Pokud se tedy zachycený síťový tok shoduje s databází, je tento tok vyhodnocen jako podezřelý a je o tom informován administrátor. Takovéto vyhodnocení nemusí však být vždy správné, může se jednat o běžné produkční síťové toky. V takovém případě může být bezpečnostní administrátor zahlcen množstvím upozornění, které však nemají žádnou hodnotu. Pokud se sítí bude šířit například

zero-day exploit který není obsažen v databázi IDS, nebude takovýto útok vůbec zaznamenán.

Naproti tomu honeypoty, které se do produkční sítě zapojí, dokáží takovéto podezřelé toky zaznamenat. Honeypot je nasazen do prostředí produkční sítě a sám o sobě není součástí žádné síťové komunikace – jen zde vyčkává, než bude kompromitován. Jakýkoliv síťový tok, který se pokusí daný honeypot kompromitovat je okamžitě vyhodnocen jako podezřelý a o útoku je informován bezpečnostní administrátor který může na vzniklou událost reagovat (37). Pokud však útok nebude veden přímo na konkrétní honeypot, nikdy se o takovém útoku nedozvíme. Na obrázku č.6 je zobrazeno zapojení honeypotu v prostředí DMZ segmentu, ve kterém je nasazen společně se třemi produkčními web servery. Na firewallu je vytvořen prostup na portu 80 (HTTP) tak, aby byly servery dostupné z internetu. Pouze web servery mají vytvořen DNS záznam, tudíž pouze tyto tři servery mohou přijímat požadavky na zobrazení web stránek. Honeypot je nastaven stejně jako ostatní web servery, avšak bez DNS záznamu. Pokud tedy útočník bude skenovat každý systém v DMZ a bude hledat web-based zranitelnosti, bude s největší pravděpodobností oskenován i honeypot.



Obrázek 6 - Schéma zapojení produkčního honeypotu (37)

3.2.4 Výzkumné honeypoty

Aby mohly být informační systémy, jakožto i aplikace či další informační zdroje dostatečně zabezpečeny před kybernetickými hrozbami, musíme znát slabiny těchto systémů, musíme znát postupy, nástroje a metody, které jsou při útocích používány blackhat komunitou. Čím více budeme mít k dispozici informací o tom, jak tato komunita postupuje, jaké nástroje využívá či jaké jsou její motivy a cíle, tím lépe dokážeme na příští útoky reagovat. K získávání těchto informací slouží výzkumné honeypoty. Tyto honeypoty neslouží pro zajištění další vrstvy v bezpečnosti organizace tak, jako produkční honeypoty. Účel těchto honeypotů je přilákat na sebe množství kybernetických hrozeb, které mohou být následně studovány a analyzovány. Na základě takovýchto analýz mohou být navržena nová bezpečnostní opatření, či vydány aktualizace, které slabiny v systému zazáplatují (40). Za výzkumný honeynet lze považovat i systém, který je v této práci navržen a spuštěn v experimentálním provozu. Daný systém je navržen tak, aby splňoval kladené požadavky na možnost následné analýzy zaznamenaných kybernetických útoků. Výzkumné honeypoty jsou nesmírně cenné na zdroje informací, které nám mohou poskytnout, neboť mohou zaznamenat úplně nové postupy v chování blackhat komunity, které do té doby nebyly známé.

Výzkumné honeypoty mohou být využívány k:

- Zachycení automatických útoků v podobě malware (červů nebo auto-rooters). Pomocí zachycení těchto hrozeb a analýzy jejich payload mohou bezpečnostní organizace lépe reagovat a neutralizovat hrozbu

- Zachycení neznámých nástrojů, technik či zero-day exploit. Po analýze těchto neznámých hrozeb mohou být například vydány bezpečnostní aktualizace.

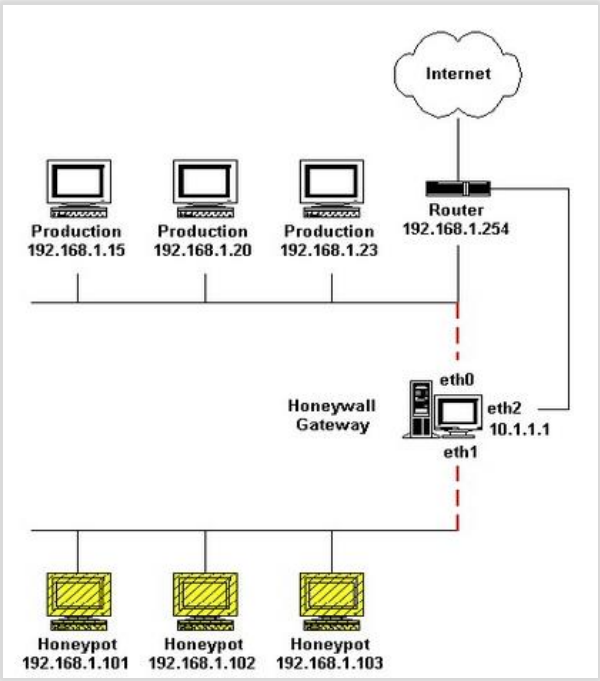
- Lepšímu pochopení motivů, které útočníka vedly k útoku na systém. Díky zachycení uceleného kybernetického útoku můžeme vyvodit závěry o tom, kdo za útokem stojí a jak a proč byl útok veden.

- K získání informací o pokročilých členech blackhat komunity, zkoumání jejich schopností a dovedností.

3.2.5 Honeynet

Honeynet je architektura, která je tvořena vysokointeraktivními honeypoty. Klíčovým prvkem honeynetu je Honeywall, který odděluje honeynet od zbytku produkční sítě, a je implementován na druhé vrstvě ISO/OSI modelu – bridge. Tato architektura zajišťuje, že bude Honeywall neviditelný pro kohokoliv, kdo bude navazovat s honeynetem spojení (41).

Na obrázku č.7 je zobrazen diagram této architektury. Z diagramu je patrné, že Honeywall disponuje třemi síťovými rozhraními, konkrétně eth0, eth1 a eth2. Každé z těchto



Obrázek 7 – Architektura zapojení honeynetu (41)

rozhraní plní v celé architektuře specifickou funkci. Eth0 společně s eth1 zajišťují oddělení celého honeynetu od zbytku sítě a mezi těmito dvěma rozhraními, probíhá síťový provoz, který je monitorován a kontrolován. Rozhraní eth2 umožňuje vzdálenou správu Honeywallu.

Aby mohla být architektura honeynetu úspěšně implementována, musí Honeywall splňovat určité požadavky na kontrolu, zachycení, analýzu a uskladnění dat. Tyto požadavky lze shrnout následujícími vlastnostmi Honeywallu:

- Data Control
- Data Capture
- Data Analysis
- Data Collection

Data control - je vlastnost, která omezuje činnost honeynetu a tím zajišťuje snížení možnosti útoků na další informační systémy, pokud je honeynet napaden. Pokud je totiž honeynet napaden, je zde vždy určité riziko, že se útočník, či škodlivý kód pokusí napadnout systémy, které existují mimo honeynet nebo se pokusí honeynet zneužít předem nepředvídatelným způsobem. Omezením činnosti se zde rozumí, že bude síťová komunikace mezi honeynetem a útočníkem kontrolována a určitým způsobem omezována. Míra tohoto omezení závisí na organizaci, která honeynet provozuje. Pokud dáme útočníkovi větší volnost, získáme obsáhlejší data o útoku, ale také se tím zvyšuje riziko, že bude honeynet útočníkem zneužit k další škodlivé činnosti. Pokud útočníkovi dáme minimální volnost, může pojmout podezření, že se jedná o nějaký druh honeypotu. Honeynet je od produkční sítě oddělen, a je umístěn ve speciálním segmentu počítačové sítě. Tento segment je kontrolován firewallem, který je nastaven tak, aby komunikace, která směřuje do honeynetu i z honeynetu ven, byla kontrolována. Čím více odchozích spojení je na firewallu povoleno, tím má útočník větší volnost při útoku, naopak, pokud veškerou odchozí komunikaci zakážeme, útočník nebude schopen napadnout jiné systémy (37).

Data capture - zajišťuje veškeré monitorování a logování aktivity honeynetu. Cílem této funkcionality je především možnost monitorování síťové aktivity na více úrovních. Tento bod je kriticky důležitý, neboť využití více úrovní monitoringu zajistí získání dat, která budou vysoce komplexního charakteru. Čím více úrovní je využito, jak na straně monitorování síťové aktivy, tak na straně monitorování samotného honeypotu, tím ucelenější pohled získáme na celý uskutečněný kybernetický útok (41). Bez toho, aniž by si útočník této aktivity všiml (37).

Data Analysis – aby bylo možné veškerá nasbíraná data podrobit analýze, musí honeynet disponovat funkcionalitou, která zajistí, že budou nasbíraná data zobrazena ve správné formě, v podobě informací, která jsou člověkem čitelná (41).

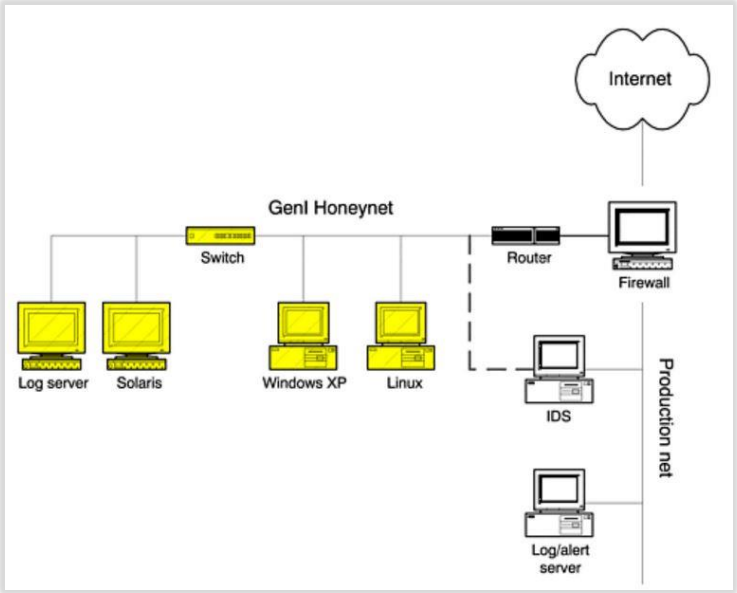
Data collection – tato funkcionalita je aplikována pouze na organizace, které mají zapojeno více honeynetů ve své produkční síti, či se jedná o organizace globálního měřítka, které mají honeynety rozmístěny po celém světe. V takovém případě je třeba data skladovat na centrálním místě. Tato data mohou být následně kombinována a porovnávána a tím se zvyšuje jejich hodnota (41). Data musí být uchovávána na bezpečném místě, a musí být zajištěna jejich integrita a autenticita (42).

3.2.6 Architektura honeynetů

The Honeynet Project (42) postupně vyvinul tři hlavní architektury honeynetů:

- GEN I
- GEN II
- GEN III

GEN I byla první architektura, která byla vyvinuta a její hlavní účel byl v zaznamenávání aktivit blackhat komunity. Na obrázku č.8 je zobrazen diagram této GEN I architektury. Tato architektura však měla omezené možnosti ve sběru a kontrole dat, proto byla efektivní v zachycení automatických útoků a útoků typu zero-day. K zachycení útoků pokročilé blackhat komunity byla však nedostatečná – byla útočníkem snadno odhalitelná, neboť disponovala specifickými signaturami ve vlastnosti data control a také nenabízela

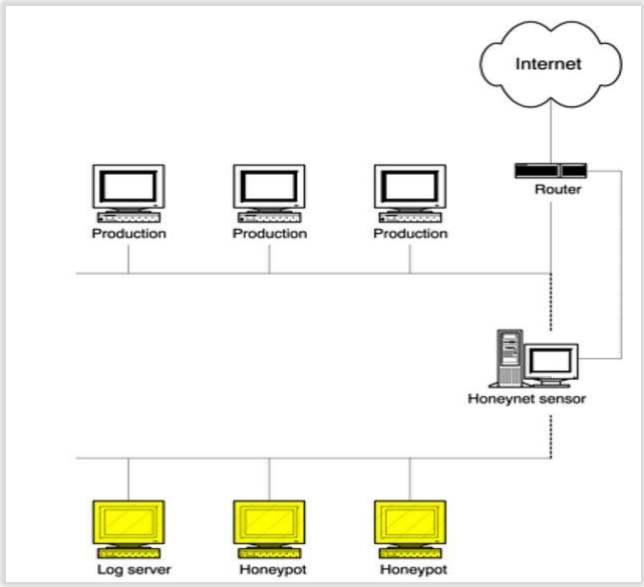


Obrázek 8 - Architektura GEN I honeynetu (37)

lákavé služby pro blackhat – jednalo se o defaultní instalace operačních systémů. Navzdory tomu, jednalo se o první skutečně vysokointeraktivní architekturu na poli honeypotů (37). Architektura GEN I je velmi jednoduchá, je vytvořen izolovaný síťový segment za přístupovým síťovým zařízením – obvykle za firewallem. Tím je docíleno toho, že veškerá síťová komunikace přicházející nebo odcházející z honeynetu, jde přes tento firewall. Většinou je zde přítomný i další síťový segment oddělený od honeynetu, který slouží pro jeho administraci.

GEN II architektura byla vyvinuta k eliminaci problémů v GEN I architektuře a zlepšuje tak flexibilitu, možnosti konfigurace a bezpečnost celého honeynetu. Tato architektura již

využívá honeynet senzor - Honeywall, který je kombinací IDS a firewallu. Tento senzor je implementován na druhé vrstvě ISO/OSI obdobně jako síťový most. Z toho důvodu je jeho odhalení pro útočníka velmi obtížné. Díky Honeywallu, který je představován jedním zařízením, je výrazně usnadněno zapojení celého systému. Obrázek č.9 zobrazuje tuto architekturu.



Obrázek 9 - Architektura GEN II honeynetu (37)

GENIII vychází z architektury GEN II a je obohacena o komponentu Sebek server a Sebek klient. Sebek klient je nástroj pro sběr dat, který je nainstalován a skryt na honeypotu a je propojen se Sebek serverem, který je umístěn na Honeywallu. Sebek zaznamenává útočníkovu aktivitu a tu zasílá na Honeywall, toto propojení je realizováno formou client-server architektury.

3.2.7 Logování honeypotu

Zatímco u nízkointeraktivních honeypotů jsou možnosti logování vcelku omezeny a jsou realizovány samotným programem, který emuluje služby na volných IP adresách sítě a zaznamenává do logfilu pokusy o připojení na jednotlivé služby a porty, je situace u vysokointeraktivních honeypotů značně rozdílná. Vzhledem ke složitosti architektury vysokointeraktivních honeypotů, je řešení logování rozděleno do několika úrovní. V první úrovni je třeba zajistit logování na úrovni síťového provozu. Tedy logování o tom, z jaké zdrojové IP adresy z jakého zdrojového portu byl veden útok na honeypot. Dále pak na jaký

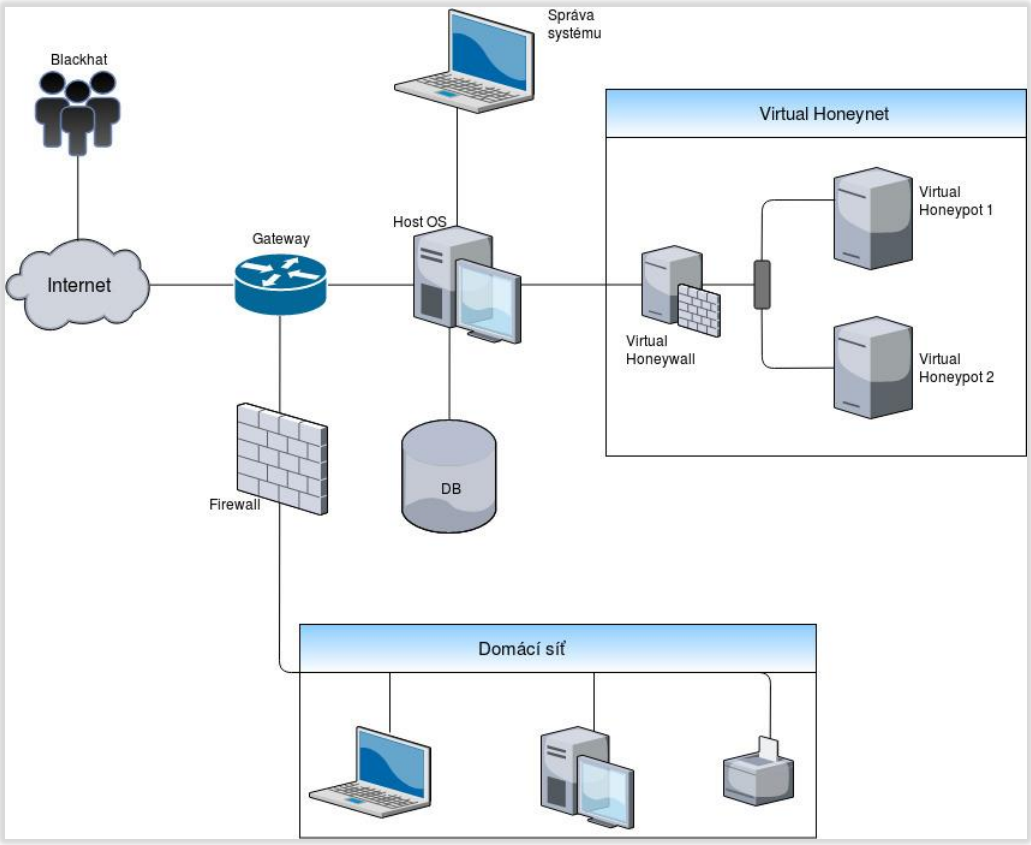
cílový port a na jakou cílovou službu byl útok veden. Mimo tyto logy je dále potřeba logovat i datum a čas těchto událostí. V další úrovni se logují samotné pakety, které byly mezi zdrojovou a cílovou IP adresou přeneseny. Díky tomu můžeme do paketů nahlížet a získáváme tak další rozšiřující pohled na kybernetický útok. K tomuto logování lze využít například nástroj Roo Honeywall (43), který je blíže popsán v kapitole 4.2.1. V další úrovni logování je třeba zajistit, abychom byly schopni rekonstrukce událostí, které na honeypotu důvodem napadení ze strany blackhat komunity vznikly. K tomuto účelu může být využit například jaderný modul Sebek (44) který funguje na principu klient-server architektury kdy serverová část je nainstalována na Honeywallu a klient na honeypotu. V momentě, kdy je honeypot napaden odesílá Sebek-klient data o aktivitě na Sebek-server. Tento nástroj je blíže popsán v kapitole 4.2.5. V další úrovni logování je zahrnuto logování na úrovni operačního systému. To znamená, že je operační systém, který je použit na daném vysokointeraktivním honeypotu, nakonfigurován tak, aby logoval události daných služeb, které jsou na tomto honeypotu spuštěny. Například v operačním systému GNU/Linux jsou pokusy o připojení k terminálu logovány tak, že daemon *rsyslog* naslouchá log-zprávám z jednotlivých částí systému a tyto logy následně třídí a ukládá do textových souborů v adresáři /var/log (45). Díky takovýmto rozsáhlým možnostem logování vysokointeraktivních honeypotů získáváme mnohem přesnější data o uskutečněných útocích, než je tomu u nízkointeraktivních honeypotů. Nízkointeraktivní honeypoty dokáží logovat pouze zdrojový port a zdrojovou IP adresu, cílový port a cílovou IP adresu, čas vzniku události a v některých případech, pokud je na dané emulované službě spuštěn script, který dokáže odpovídat útočníkovi na jeho dotazy, jsou logovány i tyto příkazy, které útočník zadá (38).

4 Vlastní práce

Hlavním cílem této práce je forenzní analýza kybernetického útoku. Aby byla tato analýza možná, musí být navržen systém, který takovéto útoky dokáže zachytit a dokáže zaznamenat potřebná data k následné analýze. Praktická část této práce je zaměřena na implementaci systému, který je tvořen virtuálním honeynetem. Tento systém je navržen a nakonfigurován tak, aby dokázal zaznamenávat data o útocích na informační systémy a monitoroval aktivitu, která stojí za různými druhy útoků na tyto systémy. V následujících kapitolách bude postupně popsána architektura tohoto systému, budou popsány jednotlivé částí a funkcionality systému. Výstupem této části práce budou data, která budou dále podrobena analýze. Na základě této analýzy bude zpracován závěr práce.

4.1 Implementovaná architektura

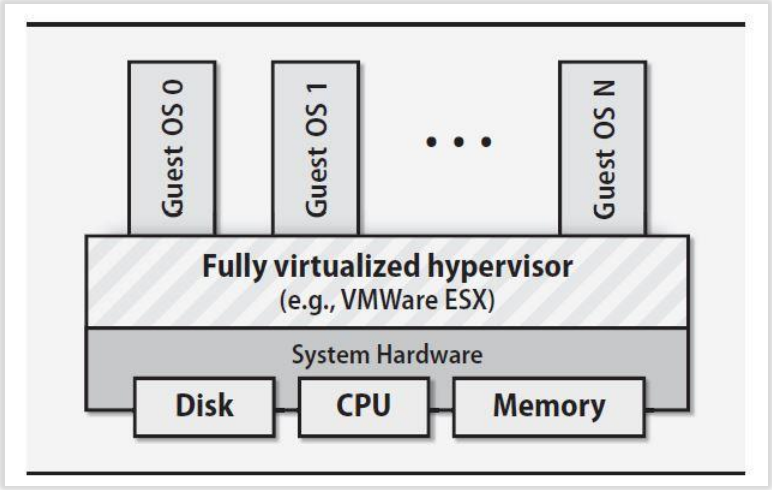
V této práci byla implementována architektura virtuálního honeynetu za použití virtualizačního software Virtualbox (46) a nástroje Honeywall (42). Tento virtuální honeynet byl nasazen v prostředí domácí sítě na odděleném síťovém segmentu. V architektuře je využit jeden fyzický stroj, který zde vystupuje jako hostitelský systém, na kterém je nainstalován virtuální Honeywall a dva virtuální honeypoty s různými operačními systémy. Druhý fyzický stroj slouží ke vzdálené správě systému. Správu systému lze provádět několika způsoby. Prvním způsobem je využití konfiguračních souborů, které jsou součástí Honeywallu. Druhý způsob správy systému představuje nástroj Walleye – jedná se o webové rozhraní, které slouží jak pro správu systému, tak pro analýzu nasbíraných dat. Tento nástroj bude blíže popsán v následujících kapitolách. Databáze, která je nainstalovaná na hostitelském počítači, slouží pro zálohu nasbíraných dat, které se primárně ukládají do databáze, která je součástí virtuálního Honeywallu. Vzhledem k tomu, že je celý systém závislý na jedné veřejné IP adrese, nebylo možné mít spuštěny oba virtuální honeypoty ve stejný okamžik, protože se služby, které honeypoty poskytují, do jisté míry překrývají. V první části experimentu byl síťový provoz přesměrován na honeypot 1 a v druhé části byl provoz přesměrován na honeypot 2. Díky tomu, bylo možné nasbírat data z honeypotů založených jak na systému GNU/Linux tak na systému Windows. Na obrázku č.10 je zobrazeno schéma zapojení. Části virtuální honeynet, se podrobněji věnuje následující podkapitola, kde bude systém blíže popsán.



Obrázek 10 - Implementovaná architektura honeynetu – vlastní zpracování

4.1.1 Virtuální honeynet

Virtualizace je proces běhu virtuální instance operačního systému ve vrstvě, která je oddělena od fyzického hardware. Použitím virtualizace můžeme zajistit běh několika virtuálních strojů současně, a to pouze za použití jednoho dedikovaného stroje (47). Při virtualizaci se o přidělování fyzických zdrojů, jako například úložiště, paměť a CPU, stará virtualizační software. Virtualizační architektura, nazývaná plná virtualizace, emuluje virtuální vrstvu zvanou „hypervisor“ která odděluje virtuální OS od fyzického hardware a stará se o přidělování systémových zdrojů (48). Na obrázku níže je zobrazeno schéma architektury plné virtualizace. Hypervizor má přímý přístup k fyzickému hardware, a tento hardware přiděluje jednotlivým virtuálním instancím. Tato architektura virtualizace je nejbezpečnější, neboť virtualizované instance jsou plně odděleny od fyzického hardware (48).



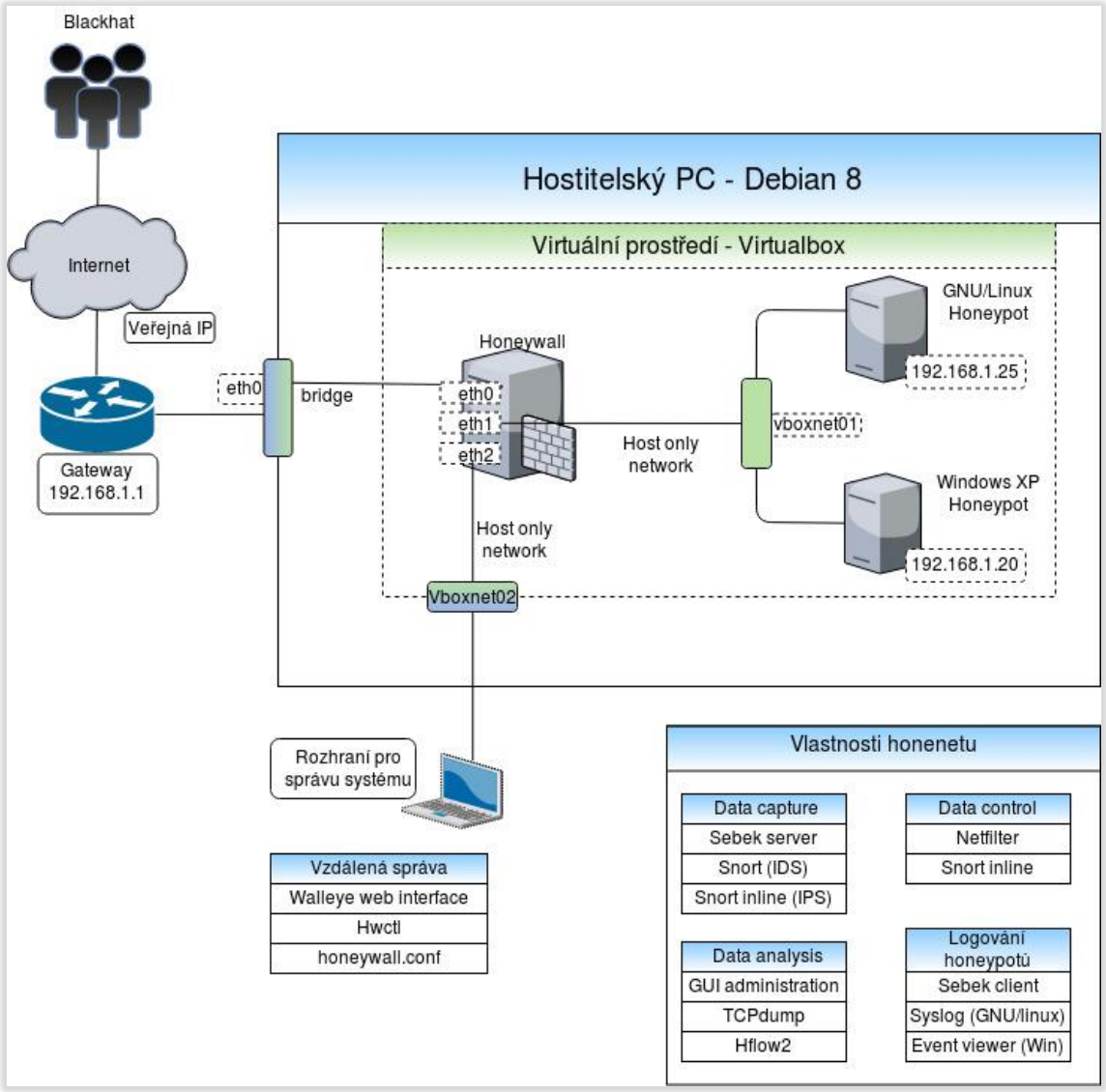
Obrázek 11 - Architektura plné virtualizace (48)

Honeypoty obecně lze implementovat jak na fyzických strojích, tak na virtuálních. Virtualizace se nemusí omezovat pouze na jeden honeypot, mohou být virtualizované celé honeynety. Virtualizace poskytne platformu, na které lze vybudovat celý honeynet za použití jednoho fyzického stroje, ze kterého bude celý systém spravován. Jsme omezeni pouze HW prostředky daného stroje, na kterém virtualizovaný systém implementujeme. V této práci byla využita architektura GEN III honeynetu za použití virtualizačního nástroje Virtualbox (46). Klíčovou komponentou systému je hostitelský počítač, na kterém je nainstalován Virtualbox software. Tento software umožnil virtualizaci systémů:

- Honeywall Roo 1.4
- Ubuntu server 7.10
- Windows XP SP2
- Debian 8

Honeywall Roo je hlavní komponentou virtuálního honeynetu – vytváří tři virtuální síťová rozhraní. Virtuální rozhraní eth0 je přemostěno do fyzického rozhraní eth0 na hostitelském počítači a zajišťuje konektivitu Honeywallu s internetem. Rozhraní eth1 je propojeno pomocí sítě pouze s hostem s virtuálním rozhraním honeypotů – vboxnet01. Rozhraní eth2 je propojeno s rozhraním vboxnet02 pomocí sítě pouze s hostem a zajišťuje tak konektivitu se systémem pro správu honeynetu. Na obrázku č.12 je zobrazena architektura navrženého honeynetu. Virtuální prostředí je zvýrazněno pomocí zelené barvy a jsou zde zobrazena

jednotlivá síťová rozhraní. Takto navržený honeynet je plně virtuální a je závislý na jednom fyzickém počítači. Pro vzdálenou správu celého systému lze využít rozhraní eth2.



Obrázek 12 - Architektura virtuálního prostředí honeynetu – vlastní zpracování

4.2 Implementace honeynetu

V této kapitole budou popsány jednotlivé nástroje, které byly při implementaci virtuálního honeynetu využity. Následující tabulka č.2 souhrnně zobrazuje jednotlivé nástroje s popisem a specifikacemi.

Software	**Popis**	**Specifikace**
GNU/Linux, Debian8, X86_64, 3.16.0	Hostitelský systém, fyzický stroj	4GB RAM Core2Duo 2,66Ghz RadeonHD 4550 200GB HDD 192.168.1.5
GNU/Linux, Debian8, X86_64, 3.16.0	Virtuální stroj, přístupový bod pro vzdálenou správu	1024MB RAM 40GB HDD Vboxnet02
Roo 1.4, CentOS based system	Honeywall, virtuální stroj	512MB RAM 40GB HDD Eth0, Eth1, Eth2
GNU/Linux,Ubuntus7.10, i386, 2.6.22	Honeypot, virtuální stroj	512MB RAM 20GB HDD 192.168.1.25 Vboxnet01
Windows XP, SP2, 32Bit	Honeypot, virtuální stroj	512MB RAM 20GB HDD 192.168.1.20 Vboxnet01
Oracle VM VirtualBox	Virtualizační SW, GPL v.2	VirtualBox 5.2 for Linux
Snort	Intrusion Detection System	Snort 2.6.1.5
Snort Inline	Intrusion Prevention System	Snort Inline 2.6.1.5
Sebek	Nástroj pro zachytávání aktivity na honeypotu, Server + klient	Sebek-server-3.0.2 Sebek-client-3.2.0b (GNU/Linux) Sebek-client-3.0.5 (Win32)
Walleye uživatelské rozhraní	Webové rozhraní pro správu a konfiguraci Honeywallu a pro analýzu dat	Walleye-1.2.11

Tabulka 2 - Použité nástroje – vlastní zpracování

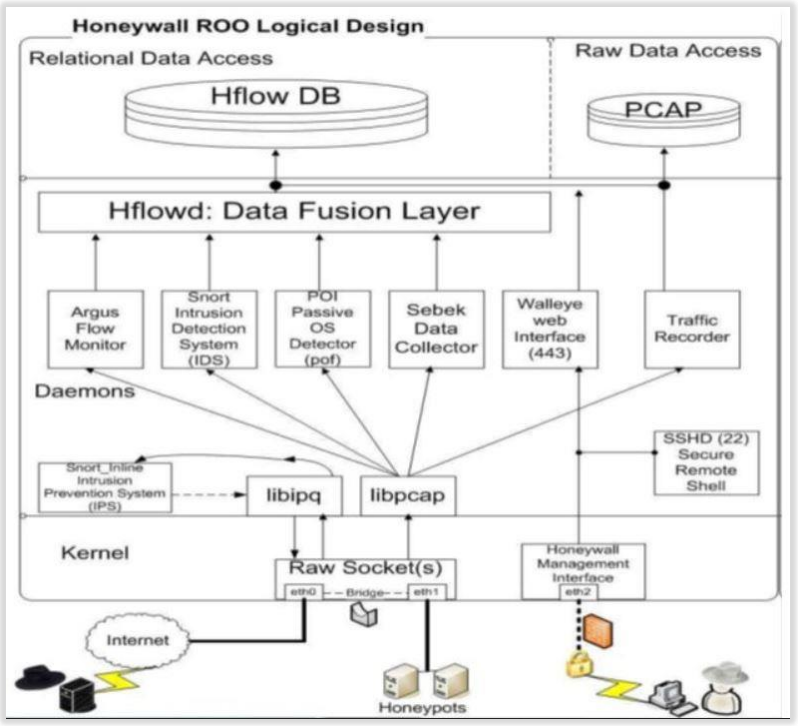
4.2.1 Honeywall Roo

Roo je ucelené řešení pro implementaci vysokointeraktivních honeypotů vytvořené společností The Honeynet Project (49). Jedná se o neziskovou organizaci, jejímž účelem je studium kybernetických útoků a vytváření open-source nástrojů, které slouží pro zlepšení počítačové bezpečnosti.

Jedním z nástrojů, který byl touto společností vyvinut, je i nástroj Roo Honeywall. Jedná se o hlavní komponentu v GENIII architektuře honeynetů a poskytuje tři hlavní funkcionality – Data Capture, Data Control a Data Analysis. Roo je šířen v podobě bootovatelného CDROM (stáhnutelného jako ISO obraz), pomocí kterého jsou nainstalovány veškeré nástroje a funkcionality, nezbytné pro vytvoření, správu a analýzu GENIII honeynetu. Jedná se o upravený a minimalizovaný operační systém založený na GNU/Linux distribuci CentOS. Tento systém obsahuje následující bezpečnostní nástroje:

- Sebek – nástroj, pro zachycení aktivity na honeypotech
- Tcpdump – nástroj, pro analýzu paketů
- Snort – Intrusion Detection System (IDS)
- Snort Inline – Intrusion Prevention System (IPS)
- Hflow2 – nástroj pro analýzu dat
- P0f – nástroj pro pasivní OS fingerprint
- Walleye – webové rozhraní pro konfiguraci a administraci Honeywallu a pro analýzu dat

Na následujícím obrázku č.13 je zobrazeno schéma, které vyjadřuje architekturu nástroje Roo. Jsou zde znázorněny jednotlivé vrstvy v architektuře a procesy, které stojí za průchozími pakety, které směřují do honeypotů a z honeypotů ven do internetu.



Obrázek 13 - Architektura nástroje Roo (41)

Instalace Honeywallu je plně automatická, po stažení poslední verze Roo 1.4 ISO obrazu (42) byl vytvořen nový virtuální stroj v prostředí aplikace Virtualbox a následně byla provedena instalace. Po dokončení instalace proběhne reboot systému a nově nainstalovaný systém Roo nabootuje. Jelikož se jedná o minimální systém, není zde obsažen X server, tudíž je systém ovládán pomocí terminálu a v případě konfigurace i pomocí jednoduchých dialogových menu. V prvním kroku je nutné se přihlásit do systému. V defaultním nastavení obsahuje Roo dva uživatelské účty: *roo* a *root*. Oba účty mají stejné defaultní heslo – *honey*. Po přihlášení pod uživatelem *roo* je možno pomocí *su* – přepnout uživatele na root. Prvotní přihlášení pod rootem není možné. Po prvotním přepnutí uživatele na root, v nenakonfigurovaném systému, se zobrazí dialogové menu, z kterého je možno provést konfiguraci systému. Dialogové menu je možné použít pouze pod uživatelem root – znovu vyvolání dialogového menu se provede příkazem *menu*. Pomocí dialogových menu formou interview (kdy se systém pomocí otázek ptá na informace, které jsou k nakonfigurování potřeba), se provede prvotní konfigurace systému – nastavení IP adres honeypotů, IP adresa brány, IP adresa Walleye, limity pro příchozí a odchozí spojení, nastavení snort inline, nastavení modulu Sebek atd. Další možností, jak systém nakonfigurovat je využití konfiguračního souboru v cestě */etc/honeywall.conf* – jedná se o klasický konfigurační soubor v textové podobě, k jeho úpravám můžeme využít některý z textových editorů,

například editor *nano* (50). Systém lze konfigurovat i pomocí webového rozhraní Walleye. Přes Walleye však nelze provést prvotní konfiguraci, neboť není ještě nakonfigurovaná IP adresa tohoto rozhraní. Walleye lze tedy využít při následných úpravách v konfiguraci systému. Poslední možností, jak lze konfigurovat systém, je pomocí utility hwct – HoneywallControl. Jedná se o utilitu, která je ovládaná skrze terminál a umožňuje měnit hodnoty nastavení Honeywallu, aktualizovat a zálohovat konfigurační soubor */etc/honeywall.conf*. Hwct je taktéž volána dialogovými menu a Walleye rozhraním, pokud je třeba provést změnu, nebo resetovat službu (43). Pokud je systém ovládán skrze SSH, je hwctl efektivní utilitou pro vzdálenou správu systému.

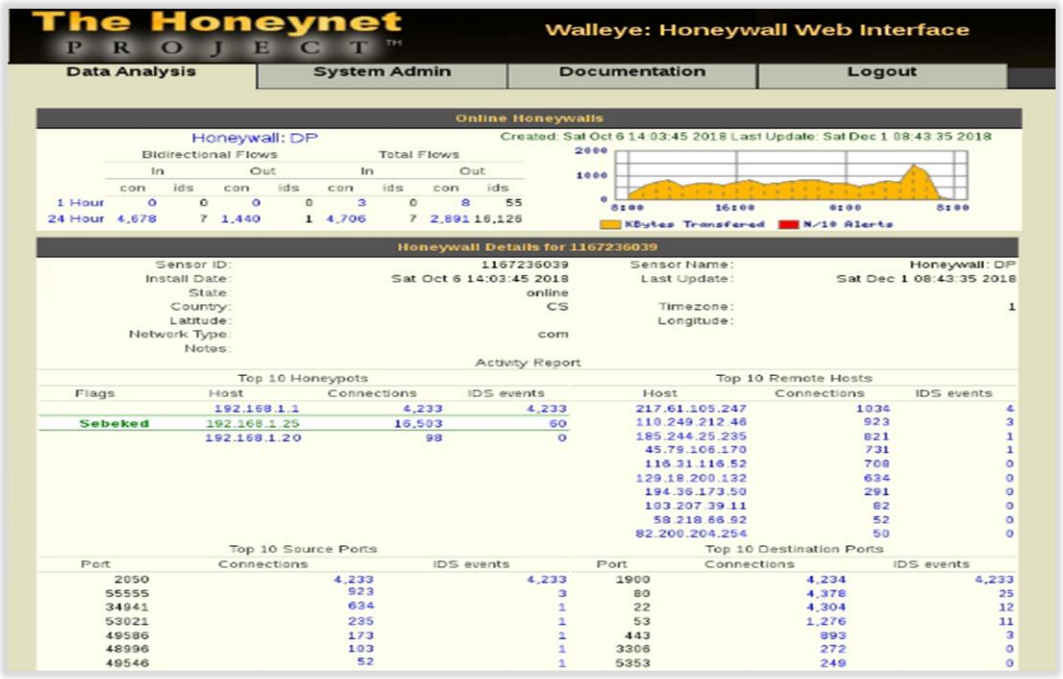
4.2.2 Walleye webové rozhraní

Webové rozhraní Walleye (Eye of Honeywall) je uživatelské grafické rozhraní pro vzdálenou správu Honeywallu a pro analýzu nasbíraných dat. Walleye podává přehledné informace o aktivitě v celém honeynetu. Aby bylo možné toto rozhraní používat, je nutno provést nastavení IP adresy, pod kterou se budeme k tomuto rozhraní připojovat. Pokud je vše správně nastaveno, je Walleye dostupné skrze adresu: https://námi-nastavená-IP. Pro vyšší bezpečnost je spojení zabezpečeno SSL certifikátem. Pro přihlášení je vyžadován login s heslem, tyto údaje jsou shodné s loginem a heslem k Roo Honeywall, tedy *roo* a *honey*. Po prvotním přihlášení je vyžádána změna hesla, nové heslo musí být dostatečně silné.

Walleye obsahuje dvě hlavní sekce – Data Analysis a System Admin. Správa systému je dostupná v sekci System Admin a je přímo propojena s konfiguračním souborem Honeywallu, případné změny v konfiguraci lze provádět v této sekci.

Sekce Data Analysis umožňuje analyzovat síťové toky v reálném čase, přehledně zobrazuje jednotlivá příchozí a odchozí spojení, zobrazuje IDS události, detaily jednotlivých spojení či zobrazení dat zachycených modulem Sebek. Walleye také umožňuje stáhnout detailní informace o jednotlivých paketech v podobě PCAP souborů, které mohou být dále podrobeny analýze. Na obrázku č.14 je zobrazen pohled na sekci Data Analysis, kde je vidět souhrn aktivity na Honeywallu během posledních 24 hodin. Graf zobrazuje množství přenesených kilobajtů v osmi hodinových intervalech. Dále je zde zobrazeno 10 nejaktivnějších IP adres, ze kterých bylo navazováno spojení, 10 nejčastějších zdrojových

portů a 10 nejčastějších cílových portů, na které byly útoky vedeny. Je zde také zobrazen počet celkových připojení na jednotlivé honeypoty a počet IDS událostí.



Obrázek 14 - GUI rozhraní Walleye – vlastní zpracování

4.2.3 Snort IDS a Snort inline IPS

Snort (51) je efektivní open-source Intrusion Detection System (IDS) a Intrusion Prevention System (IPS) se schopností analýzy síťového provozu a logování paketů na úrovni IP protokolu. Snort funguje na principech provádění detekce pomocí pravidel, politik a anomálií. Detekce pomocí pravidel funguje tím způsobem, že jsou síťové toky porovnávány se signaturami, které jsou uloženy v lokální databázi. Pokud se obsah daného rámce s těmito signaturami shoduje, je tento tok vyhodnocen jako útok, či podezřelá aktivita. Detekce založená na politikách umožňuje detekci v tom případě, pokud jsou správně nadefinované politiky a je stanoveno, jakým způsobem, kdo a jak může komunikovat uvnitř sítě. Pokud se vyskytne komunikace mimo tyto politiky, je tato komunikace detekována a vyhodnocena jako podezřelá. Detekce založená na anomáliích je implementována pomocí snort senzoru, který je nastaven tak, aby věděl, jaké činnosti a v jaké míře jsou uvnitř sítě prováděny. Pokud se chování uvnitř sítě bude tomuto nastavení vymykat, je vyhodnoceno jak podezřelé (52).

Snort může pracovat v následujících režimech:

- *Sniffer* – v tomto režimu jsou veškeré pakety procházející senzorem zobrazovány na výstupní monitor.
- *Logger* – v tomto režimu jsou veškeré pakety procházející senzorem ukládány do souboru pro následnou analýzu.
- *Network Intrusion Detection System* (NIDS) – v tomto režimu Snort pasivně analyzuje veškerou síťovou komunikaci procházející senzorem, a na základě definovaných pravidel a signatur, porovnávací metodou vyhodnocuje anomálie.
- *Inline režim* – tento režim zajistí, že pokud bude honeynet napaden, nebudou napadeny ostatní stroje v síti. Snort inline umožňuje modifikovat, odmítat a ignorovat pakety které mají signatury shodné se známými útoky. Především schopnost modifikace je pro honeynet důležitá, neboť umožňuje modifikovat obsah nebezpečných paketů a změnit povahu paketu ze škodlivého na neutrální.

Na následujícím příkladu je ukázáno pravidlo snort inline režimu. V tomto případě snort inline hledá pakety obsahující payload s DNS expoit, který se snaží získat přístup ke

```
alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg:"DNS EXPLOIT
named";flags: A+; content:"|CD80 E8D7 FFFFFF|/bin/sh";
replace:"|0000 E8D7 FFFFFF|/ben/sh";)
```

Snort inline pravidlo pro DNS exploit (39)

GNU/Linux bash shellu – /bin/sh na napadeném systému. Pokud je takovýto paket objeven, je jeho obsah změněn na /ben/.sh. Pomocí tohoto pravidla je efekt exploitu vyrušen bez toho, aby to útočník věděl. Walleye poskytuje přehled IDS událostí a umožňuje analýzu těchto paketů. Honeywall tyto pakety ukládá do PCAP souborů které lze stáhnout prostřednictvím uživatelského rozhraní Walleye, nebo je lze vyexportovat z databáze která je umístěna na Roo Honeywallu.

4.2.4 IPTables a Netfilter

Iptables je user-space nástroj kterým je ovládán engine pro filtraci paketů – Netfilter. Jedná se o nástroj, který je implementován v operačních systémech GNU/Linux od jádra verze 2.4 a slouží pro manipulaci s tabulkami Xtables a v nich umístěných řetězců složených

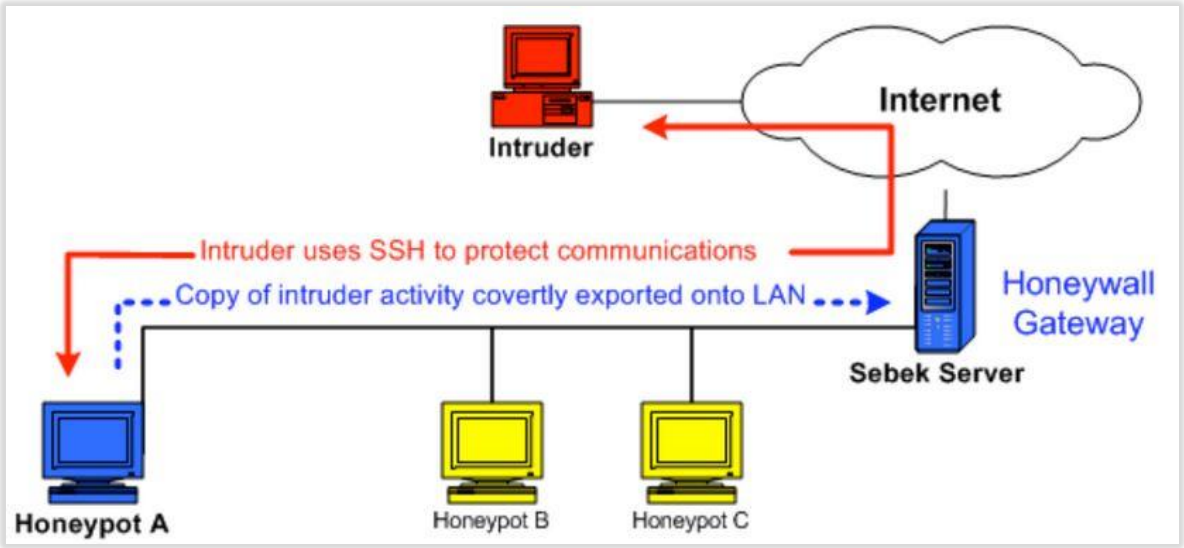
z pravidel. Tato pravidla slouží k ovlivňování průchodu paketů subsystémem jádra operačního systému – *TCP/IP stack*. Pomocí tohoto nástroje lze vytvářet pravidla, která průchod paketů ovlivňují. Defaultní iptable tabulka je nazvána *filter* a obsahuje tři řetězce pravidel: FORWARD, INPUT a OUTPUT. Každý paket, který prochází jádrem, musí projít jedním z těchto řetězců (48). Pravidla v řetězci FORWARD jsou určena pro pakety, které přicházejí na jedno síťové rozhraní a musí být přesměrována do jiného rozhraní. Řetězce INPUT a OUTPUT jsou pro pakety, které směřují nebo odcházejí z localhost. Kromě tabulky *filter*, jsou zde dále tabulky *nat* a *mangle*. Tabulka *nat* obsahuje řetězce pravidel pro kontrolu network adress translation a tabulka *mangle*, obsahuje pravidla, které modifikují nebo upozorňují na síťové pakety mimo kontext *nat* a *filter*.

Každé pravidlo z řetězce má určitý *target* neboli cíl, který určuje, co se má stát s paketem, který toto pravidlo splňuje. Tyto *targets,* které jsou v pravidlech dostupné, jsou následující: ACCEPT, DROP, REJECT, LOG, MIRROR, QUEUE, REDIRECT, RETURN a ULOG. Pokud tedy pravidlo obsahuje *target* ACCEPT, jsou všechny pakety splňující toto pravidlo povoleny, a nechá se jim volný průchod k jejich cíli. DROP a REJECT zahodí pakety s tím, že DROP je „tichý“ a REJECT zašle zpětnou chybovou hlášku pomocí ICMP protokolu. LOG zajišťuje zaznamenávání paketů, které splňují dané pravidlo a ULOG zaznamenávání paketů rozšiřuje. REDIRECT zajišťuje přesměrování paketů na jiný cíl. MIRROR vymění zdrojovou IP adresu za cílovou, než je paket dále odeslán a QUEUE zajišťuje přesměrování paketů do programů uživatele pomocí jaderného modulu (48).

V případě Roo Honeywall lze netfilter ovládat jak z grafického uživatelského rozhraní – Walleye tak z příkazové řádky systému Roo. Nejdůležitějším pravidlem Honeywallu je počet odchozích spojení za jednotku času. Správné nastavení tohoto pravidla zajistí bezpečnost vůči ostatním systémům v síti a zároveň umožní útočníkovi například stažení rootkitu, aniž by pojal podezření, že se nachází v prostředí honeypotu – může do jisté míry komunikovat s okolním světem. Zároveň, pokud je honeypot napaden nějakým typem malware, který má schopnost šířit sám sebe a infikovat tak další systémy, je po dosažení tohoto limitu spojení přerušeno. Dále můžeme nastavit blacklist, tedy z jakých zdrojových IP adres budou pokusy o připojení rovnou zahazovány a také můžeme nastavit povolení/zakázání určitých protokolů.

4.2.5 Sebek

Sebek (44) je nástroj pro zachycení dat, jehož hlavním účelem je zachytávat ta data, která nám umožní zpětnou rekonstrukci událostí, které se staly na honeypotu během jeho napadení. Architektura tohoto nástroje je zobrazena na obrázku č.15. Sebek tedy zachytává veškerou aktivitu, kterou útočník během útoku na honeypot produkuje, nejenom *keystrokes*, ale veškerá *sys_read* data. Sebek funguje na architektuře klient - server, kde klient je implementován jako *loadable kernel module* (LKM) v případě GNU/Linux OS, nebo jako jaderný ovladač v případě OS Windows. Klient dokáže zaznamenat veškerá data, ke kterým bylo přistupováno pomocí *read()* systémového volání. Toho je docíleno tím způsobem, že je funkce *read()* v tabulce systémových volání nahrazena novou funkcí. Tato nová funkce volá původní funkci, zkopíruje její obsah a zabalí ho do paketu. Data v podobě těchto paketů jsou následně exportována na server prostřednictvím UDP paketů v takové podobě, která je velmi obtížně detekovatelná z honeypotu. Každý paket obsahuje část informace o systémovém volání, které bylo učiněno, a o datech, ke kterým bylo pomocí tohoto systémového volání přistupováno. Každý paket obsahuje jeden Sebek záznam, který



Obrázek 15 - Sběr keystrokes z honeypotu pomocí modulu Sebek (44)

obsahuje popis procesu, který systémové volání učinil, čas tohoto volání a samotná data. Každý paket je vytvořen modulem Sebek, aniž by byl využit *TCP/IP stack*. Právě díky této funkcionalitě je pro uživatele nemožné tyto pakety blokovat pomocí IPTABLES či je zachytit pomocí snifferu. Aby nemohly být Sebek pakety z honeypotu A zjistitelné na honeypotu B, je součástí modulu Sebek i modifikovaná implementace nástroje Raw Socket

Interface. Díky tomuto nástroji jsou Sebek pakety putující po síti z jiných honeypotů ignorovány (44).

Serverová část modulu Sebek je tvořena třemi komponentami. První komponentou je *sbk_extract*. Tato komponenta zajišťuje zachycení Sebek paketů přímo ze síťového rozhraní, nebo jejich extrakci z tcpdump souboru a slouží k analýze Sebek paketů. Jakmile je Sebek paket zachycen a extrahován, je možné ho uložit do databáze, nebo zobrazit výstup v terminálu. Další komponentou je Perl script *sbk_ks_log.pl* který již extrahované pakety posílá na výstup STDOUT, díky tomuto scriptu je možné sledovat události na honeypotu v reálném čase přímo v příkazové řádce. Třetí komponenta je script *sebekd.pl*, který slouží pro nahrání již extrahovaných Sebek dat do externí mysql databáze pro zálohu získaných dat. Aby nebylo možné modul na honeypotu odhalit, je nutné jeho přítomnost zamaskovat. Toho je docíleno modulem Cleaner. Tento modul upravuje seznam nainstalovaných modulů v systému a modul Sebek z tohoto seznamu odstraňuje.

Modul Sebek byl nainstalován na obou virtuálních honeypotech a byl nastaven tak, aby zasílal data na serverovou část která je součástí Honeywallu. Architekturu modulu Sebek znázorňuje obrázek č.15.

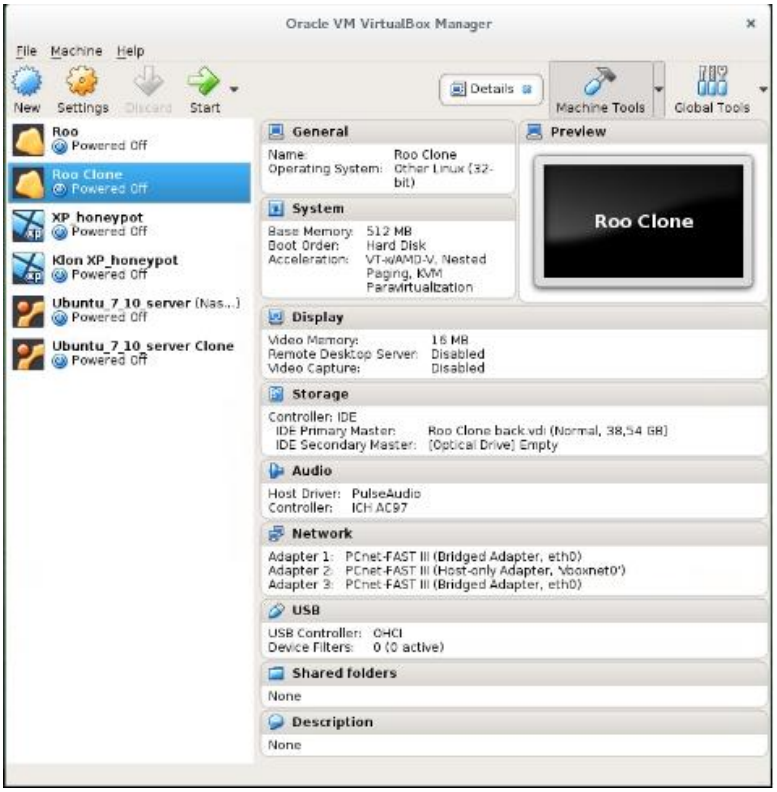
4.2.6 Virtualbox

Jako virtualizační nástroj byl použit Oracle VirtualBox 5.2 (46). Jedná se o open-source nástroj, který je dostupný jak pro systémy Windows, tak pro GNU/Linux a OS X. Toto řešení disponuje požadovanými funkcionalitami pro implementaci virtuálního honeynetu, především umožňuje souběžný běh několika virtuálních strojů a možnost vytvořit virtuální síťová rozhraní. VirtualBox byl nainstalován na GNU/Linux distribuci Debian 8 která slouží jako host OS. V prostředí VirtualBoxu byly vytvořeny čtyři virtuální stroje – Roo, Ubuntu_7_10_server, XP_honeypot a Debian8. Po instalaci a konfiguraci jednotlivých systémů byly vytvořeny jejich klony pro případnou obnovu systému v případě havárie.

VirtualBox umožňuje několik způsobů, jak zapojit virtuální síťové karty jednotlivých virtuálních strojů. Pro každý virtuální stroj může být využito více síťových rozhraní. Této funkcionality bylo využito při implementaci systému Roo. Tomuto systému byly vytvořeny tři síťové karty. První karta byla připojena pomocí síťového mostu k rozhraní eth0, druhá karta byla připojena pomocí sítě pouze s hostem k rozhraní vboxnet01 a třetí karta byla připojena pomocí sítě pouze s hostem k rozhraní vboxnet02. Honeypoty

Ubuntu_7_10_server a XP_honeypot byly připojeny k rozhraní vboxnet01 pomocí sítě pouze s hostem. Virtuální stroj Debian8 byl připojen k rozhraní vboxnet02 pomocí sítě pouze s hostem. Obrázek č.16 znázorňuje nainstalované virtuální systémy v prostředí nástroje VirtualBox. Při implementaci byly využity dva typy síťového připojení:

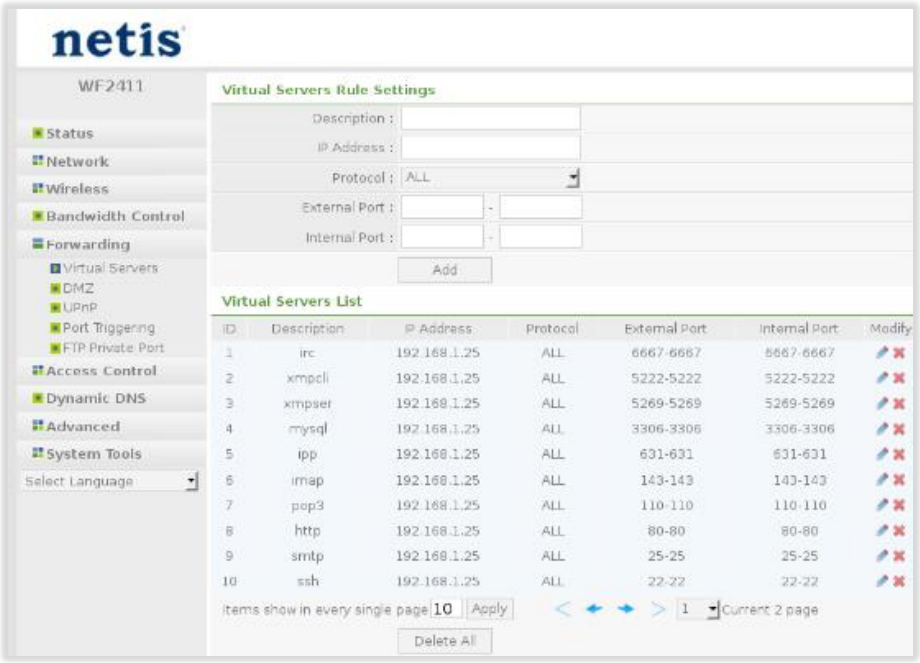
- *Síťový most* – Host OS zde hraji roli transparentního mostu mezi fyzickým síťovým rozhraním a virtuálním rozhraním. Virtuální stroj tak získá přímé propojení s fyzickým síťovým rozhraním.
- *Síť pouze s hostem* – V tomto typu je vytvořena síť mezi hostem a virtuálním strojem tím způsobem, že je komunikace možná pouze mezi těmito systémy a přístup k internetu z virtuálních strojů není možný. Pokud je v této síti více virtuálních strojů, mohou komunikovat mezi sebou.



Obrázek 16 - Virtuální rozhraní nástroje Virtualbox – vlastní zpracování

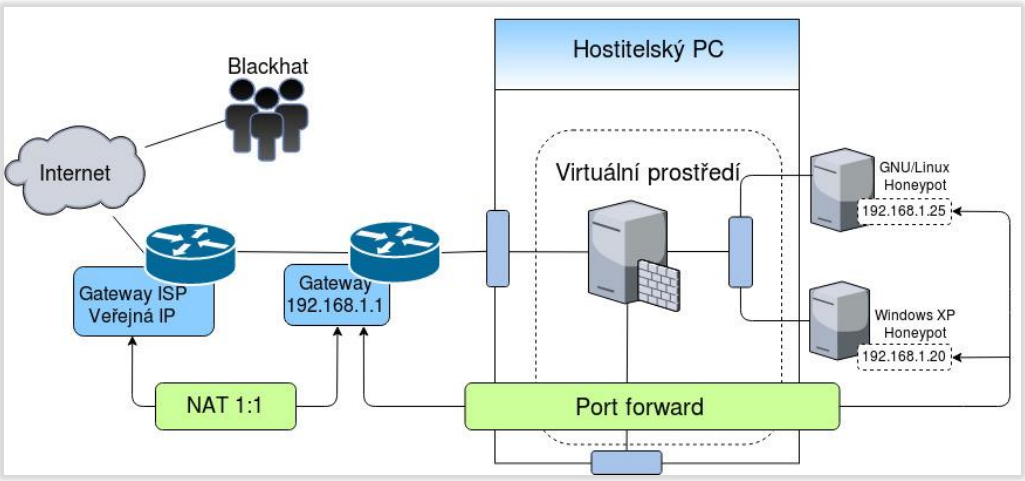
4.2.7 Síťování

Aby byly honeypoty dostupné z internetu, musí mít veřejnou IP adresu. Od poskytovatele byla poskytnuta veřejná IP adresa v NAT 1:1. Tímto způsobem bylo zajištěno, že síťový provoz směřující na tuto IP adresu byl přesměrován na vnitřní adresu brány –



Obrázek 17 - Port forward z gateway na honeypot – vlastní zpracování

192.168.1.1. Z této brány bylo nutné zajistit další přesměrování síťového provozu na konkrétní virtuální honeypoty. Na obrázku č.17 jsou zobrazena jednotlivá pravidla pro forwarding paketů na virtuální honeypoty. Tato pravidla byla nastavena na gateway vnitřní sítě. Jednotlivé pakety tedy přicházejí na síťové rozhraní (eth0) hostitelského systému, dále procházejí skrze Honeywall, který hraje roli síťového mostu a jsou směřovány k jednotlivým honeypotům. Vzhledem k tomu, že celý systém je závislý pouze na jedné veřejné IP adrese, nebylo možné zajistit současný běh obou honeypotů, neboť služby, které jsou honeypoty poskytovány je možno forwardovat buď na jeden nebo na druhý honeypot. Aby mohla být data sbírána z obou honeypotů, bylo nutné provoz celého honeynetu rozdělit do dvou částí. V každé části experimentu byl síťový provoz směřován na jiný honeypot. Na obrázku č.18 je zobrazen diagram, který zjednodušeně zobrazuje síťovou komunikaci mezi honeypoty a internetem. Z vnějšího pohledu se tedy honeypoty jeví jako fyzické stroje, které sídlí přímo na veřejné IP adrese.



Obrázek 18 - Síťování v experimentálním segmentu sítě – vlastní zpracování

4.2.8 Konfigurace honeypotů

Každý ze dvou honeypotů v experimentálním prostředí je reprezentován individuálním virtuálním strojem s nainstalovaným starším neaktualizovaným operačním systémem, který představuje snadný cíl pro potencionální útočníky. Vzhledem k tomu, že cílem je provést forenzní analýzu chování útočníka v napadeném systému, hlavní důraz je kladen na implementaci služby SSH a implementaci modulu Sebek, díky kterému bude možno zrekonstruovat jednotlivé kroky, které útočník po napadení systému učinil. Mimo služby SSH byly nainstalovány i další zranitelné služby tak, aby byl honeypot pro útočníky více lákavý. Pro primární honeypot byla zvolena GNU/Linux distribuce Ubuntu server 7.10 s verzí jádra 2.6. Na tomto honeypotu byly dále nainstalovány zranitelné služby jako: OpenSSH, HTTP, SMTP, POP3, IMAP, IRC a další. Pro sekundární honeypot byla zvolena instalace Windows XP SP2. Na tomto honeypotu byl nainstalován zastaralý a zranitelný balík služeb XAMPP 1.6.6 který obsahuje služby FTP, MySQL, HTTP a phpmyadmin. Všechny tyto služby obsahují známé zranitelnosti, díky kterým se honeypot stává snadným cílem potencionálních útočníků.

Pro snazší získání kontroly nad primárním honeypotem (Ubuntu 7.10 server) ze strany útočníků, byl po instalaci OpenSSH serveru vytvořen uživatel USER s heslem USER1. Po každém obnovení honeypotu ze snapshotu bylo toto heslo změněno.

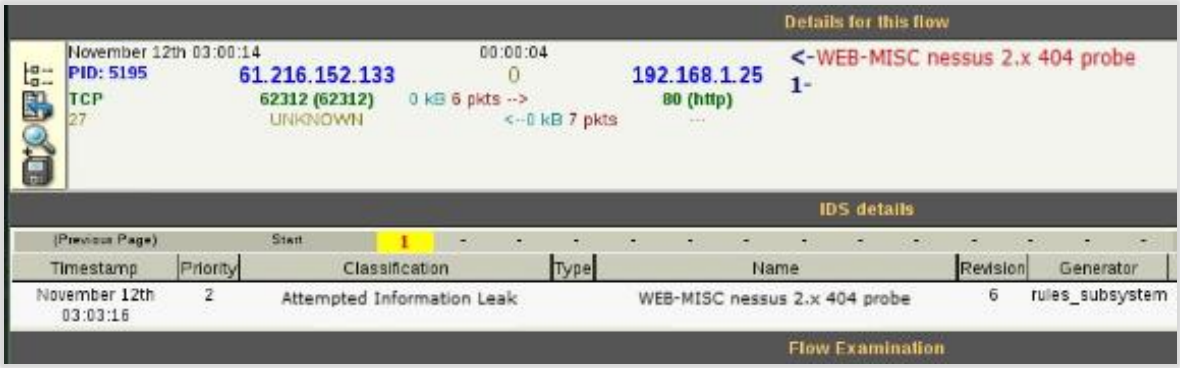
5 Výsledky a diskuse

Výsledkem práce je pohled na ucelený kybernetický útok, který byl veden ze strany útočníka a byl úspěšně zachycen na honeypotu. Aby mohl být takovýto útok zachycen, musel být nejprve navrhnut vysokointeraktivní honeynet, který by dokázal útočníka přilákat a dokázal by zachytit jednotlivé kroky, které útočník po napadení systému učinil. Klíčová komponenta tohoto honeynetu je jaderný modul Sebek, který dokáže zachytit jednotlivé *keystrokes,* tedy stisknutí kláves, které útočník vykonal v terminálu během útoku. Další klíčovou komponentou je implementace služby SSH na honeypotu. Tento síťový komunikační protokol zajišťuje vzdálený přístup k terminálu, pomocí kterého lze ovládat celý počítač. V kombinaci s modulem Sebek, lze tedy dosáhnout zachycení jednotlivých kroků a postupů, jaké útočník po napadení systému učinil. Tato část práce popisuje analýzu útoků, které byly vedeny na implementované honeypoty a které byly úspěšně zachyceny, důraz je kladen především na analýzu útoku hrubou silou na službu SSH. Kromě analýzy jednotlivých druhů útoků je zde zahrnuta i statistická a geografická analýza, která podává ucelený pohled na počet útoků za dané období, a na geografická data, která vypovídají o zemích, z kterých byly útoky vedeny.

5.1 Forenzní analýza

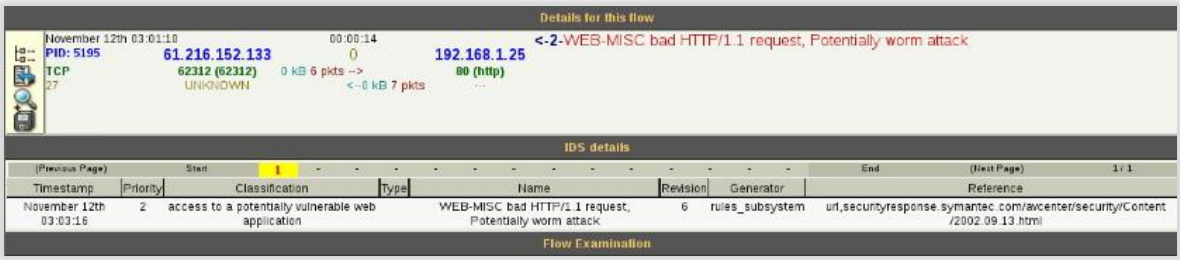
V oblasti počítačové bezpečnosti je pod pojmem forenzní analýza myšleno detailní vyšetřování a studování vzniklých bezpečnostních incidentů. Cílem tohoto vyšetřování je zjistit, jakým způsobem a pomocí jakých nástrojů byl veden útok vůči informačnímu systému. Kdy a jak daný útok započal a co útoku předcházelo a jaký měl útok dopad na chod a funkčnost informačního systému, jakož i na data, která tento informační systém obsahuje. V širším pojetí je pod pojmem forenzní analýza myšlen i sběr důkazů z elektronických zařízení, jenž slouží jako důkazní materiál u soudních procesů. Forenzní analýza zahrnuje široké množství vyšetřovacích metod a postupů, od analýzy síťových toků, analýzy systémových logů až po rekonstrukci smazaných dat na fyzických discích. Cílem této práce je forenzní analýza chování útočníka, k tomu budou využity především metody síťové forenzní analýzy, analýzy systémových logů, netfilter logů, snort inline paketů a sebek paketů. Aby mohla být provedena rekonstrukce útočníkovy aktivity na napadeném honeypotu, bylo třeba se zaměřit především na komunikační protokol SSH, který slouží k ovládání vzdáleného počítače.

Postup při vyšetřování útoků na honeypoty byl následující: prvním krokem při vyšetřování bezpečnostního incidentu bylo studium, jakým způsobem útočník zahájil svůj útok. K tomuto prvotnímu kroku bylo vždy využito GUI Walleye, neboť přehledně zobrazuje síťové toky a lze zjistit, kdy útok započal. Pokud se jedná o promyšlený útok, předchází mu scanování portů a zjištění zranitelností, které služby na těchto portech mají. Ke zjištění těchto informací lze využít nástroje jako například nmap nebo Nessus. Na obrázku č.19 je zachycen snort alert, který popisuje, jaký nástroj útočník ke scanu portu využil. V tomto případě se jedná o nástroj Nessus což je vulnerability scanner sloužící ke



Obrázek 19 - Snort alert - scan portu č.80 pomocí nástroje Nessus – vlastní zpracování

zjištění zranitelností dané služby. Útočník si tedy zjistil zranitelnosti služby HTTP na portu 80 a na základě těchto informací zahájil útok. Na následujícím obrázku č.20 je zobrazen snort alert, který upozorňuje na potencionální worm attack. Tento útok nastal během minuty po prvotním scanování portu ze stejné zdrojové IP adresy. Grafické rozhraní Walleye



Obrázek 20 - Snort alert - potentially worm attack – vlastní zpracování

přehledně zobrazuje informace o síťové komunikaci mezi honeypoty a potencionálními útočníky. Abychom však dokázali provést forenzní analýzu chování útočníka po napadení honeypotu, musíme využít další techniky, především techniku rekonstrukce jednotlivých příkazů, které útočník po napadení systému zadal do terminálu. Na základě této prvotní analýzy síťového provozu ve Walleye byly vyhodnoceny jednotlivé druhy útoků, a tyto

útoky byly následně podrobeny analýze logů dané služby, analýze paketů a analýze Sebek paketů.

5.1.1 Forenzní analýza zachyceného útoku

Při forenzní analýze chování útočníka byla pozornost zaměřena především na studium útoků vůči službě SSH. Tato služba umožňuje vzdálený přístup k terminálu daného počítače a autentizace uživatele v experimentálním prostředí byla implementována pomocí jména a hesla. Při útocích na služby zabezpečené tímto autentizačním mechanizmem se útočník snaží získat přístup k systému, zařízení či aplikaci pomocí odposlechnutí či prolomení (uhádnutí) kombinace loginu a hesla. Mezi tyto útoky patří i útoky na slabá místa v hashovacích algoritmech s využitím technik na derivaci hesel z takzvaných *hashů*. Existují typy cryptografických algoritmů, které využívají operační systémy a aplikace k vytvoření hashe. Mezi tyto algoritmy patří například algoritmus SHA. Při útoku na tyto hashe, se vychází z toho, že stejné heslo zašifrované stejným hashovacím algoritmem má pokaždé stejný výstup. Pokud je tento hash odposlechnut v síťové komunikaci, lze ho pomocí metod reverzního inženýrství prolomit a to tak, že použijeme stejný hashovací algoritmus na potencionální hesla a ty budeme pomocí tohoto algoritmu šifrovat. Pokud se výsledný hash bude shodovat s odposlechnutým, známe použité heslo (32). K prolomení hesel se využívají tyto útoky:

- *Slovníkový útok* – v tomto útoku se využívají soubory, které obsahují množství slov, frází a různých kombinací znaků v různých světových jazycích, které by mohli uživatelé používat jako svá hesla – takzvané slovníky (35). Útočník může také využít slovník s loginy, který obsahuje uživatelská jména. K prolomení hesla se využívají programy jako například THC Hydra (53). Při prolamování je vůči specifickému protokolu testované každé slovo ze slovníku hesel, popřípadě i ze slovníku loginů. Pokud slovníky obsahují daná hesla a loginy, dojde ke shodě a heslo je prolomeno. V případě hashovaných hesel je třeba slova ve slovníku zahashovat příslušným algoritmem a následně hledat shodu, nebo využít takzvané *rainbow tables* které obsahují hashovaná slova ze slovníků a umožňují útočníkovi rychle nalézt shodu (35).

- *Útok hrubou silou* – Tento útok proti heslově založené autentizaci vyžaduje využití takzvaných *setů* – tyto sety obsahují jakékoliv znaky (včetně speciálních znaků a čísel). Útok započne tím, že je použit první znak z tohoto setu a je testován jako heslo vůči autentizačnímu systému. Pokud nenastane shoda, zkouší se další znak ze setu. Takto jsou testovány vůči autentizaci veškeré možné kombinace znaků z daného setu. Tento útok může být velmi časově náročný – záleží na množství znaků v setu a na konečné délce hesla (35).

Během časového období, ve kterém byly honeypoty nasazeny, byly zaznamenány především slovníkové útoky vůči službě SSH. Na službu SSH bylo vedeno celkem 38532 pokusů o připojení. Z těchto všech útoků bylo pouze 18 útoků úspěšných – tedy došlo k prolomení hesla. Po prolomení hesla následoval již samotný útok, který se odehrál s určitým časovým odstupem a nastal jen v určitých případech. Z těchto 18 prolomení hesel do systému byl následný útok vedený ze strany hackera zaznamenán pouze v osmi případech. Po napadení honeypotu byl honeypot pozorován po dobu 48 hodin a pokud nebyl v tomto intervalu zaznamenán žádný útok, který by měl spojitost s prolomením hesla, honeypot byl obnoven ze snapshotu do původního stavu před napadením. Zaznamenané útoky měly vesměs stejné scénáře. Útočník si nejprve zjistil informace o systému a o přihlášených uživatelích, a následně vytvořil skrytou složku a pokusil se do ní stáhnout některý ze svých nástrojů, zametl po sobě stopy a poté se odhlásil. V některých případech se útočník pokusil získat přístup k účtu roota a v pěti případech útočník před odpojením se od systému změnil heslo uživatele. Ve třech případech se útočník nijak nesnažil smazat historii vykonaných příkazů. Z těchto tří případů byly ve dvou případech zaznamenány kroky, které nevedly k žádnému ohrožení systému – útočník si jen zjistil informace o systému a uživatelích a následně se odhlásil. Při analýze rekonstruovaných *keystrokes*, byla pozornost zaměřena i na časové intervaly, ve kterých byly jednotlivé příkazy zadány. Ve dvou případech byly příkazy zadávány téměř okamžitě – z toho lze vyvodit, že útočník k útoku využil některý automatizovaný nástroj.

Prvotní analýza byla provedena ve webovém GUI Walleye, kde se dalo přesně určit, kdy daný útok započal, jak dlouho trval a kdy skončil. Následující obrázek č.21 zachycuje

Obrázek 21 - Zahájení útoku na honeypot 192.168.1.25 – vlastní zpracování

zahájení útoku vedeného na honeypot s operačním systémem GNU/Linux s IP 192.168.1.25 a na službu SSH. Útok započal dne 19.11 2018 ve 4:09 hodin, trval 43minut a byl veden z IP 209.141.51.85. Po této analýze byla provedena analýza SSH logů, při které bylo zjištěno, že se danému útočníkovi podařilo heslo prolomit. Na obrázku č.22 je vidět část logfilu

```
Nov 19 04:52:44 ubuntuDP sshd[145285]: Failed password for invalid user mysql from 209.141.51.85 port 35420 sshd2
Nov 19 04:52:46 ubuntuDP sshd[145285]: pam_unix(ssh:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser=
rhost=209.141.51.85 user=root
Nov 19 04:52:50 ubuntuDP sshd[145267]: Failed password for invalid user mysql1 from 209.141.51.85 port 49352 sshd2
Nov 19 04:52:52 ubuntuDP sshd[145267]: pam_unix(ssh:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser=
rhost=209.141.51.85 user=root
Nov 19 04:52:56 ubuntuDP sshd[145278]: Failed password for invalid user test from 209.141.51.85 port 13809 sshd2
Nov 19 04:53:01 ubuntuDP sshd[145278]: pam_unix(ssh:auth): authentication failure; logname= uid=0 euid=0 tty=ssh ruser=
rhost=209.141.51.85 user=root
Nov 19 04:53:06 ubuntuDP sshd[145285]: Accepted password for user from 209.141.51.85 port 41429 sshd2
Nov 19 04:53:06 ubuntuDP sshd[145285]: pam_unix(ssh:session): session opend for user user by (uid=0)
Nov 19 04:53:06 ubuntuDP sshd[145285]: Received disconnect from 209.141.51.85: 11: Bye Bye
Nov 19 04:53:06 ubuntuDP sshd[145285]: pam_unix(ssh:session): session closed for user user
```

Obrázek 22 - SSH Logfile z napadeného honeypotu – vlastní zpracování

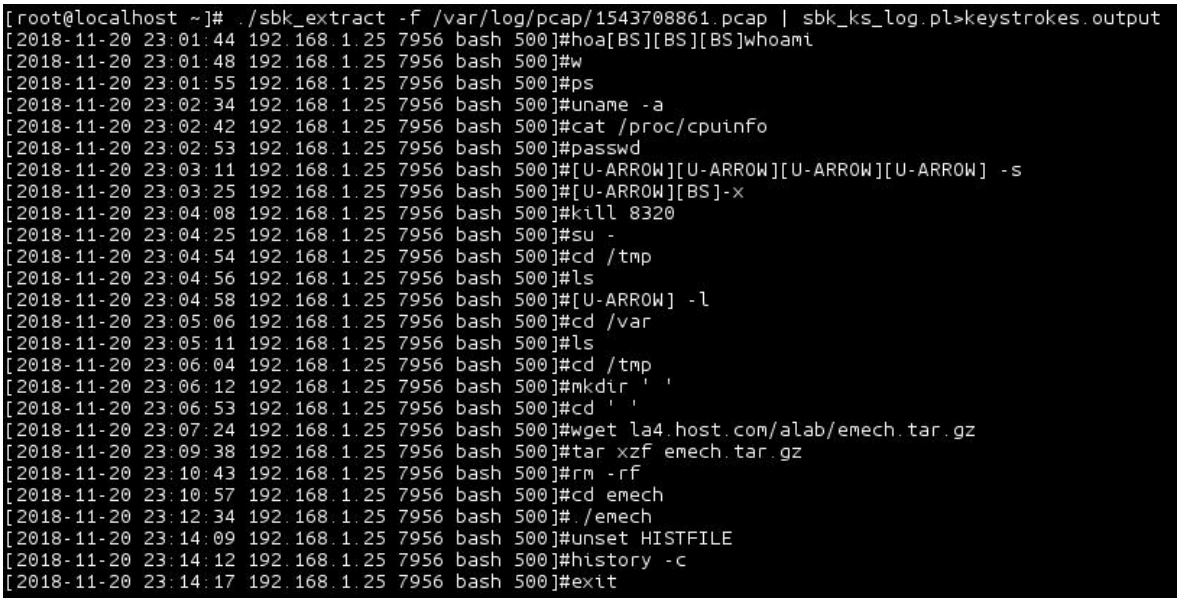
z napadeného honeypotu. Z logu je patrné, že se v čase 04:53:06 podařilo heslo prolomit a útočník získal přístup do systému. Okamžitě po prolomení hesla došlo k odhlášení. Z této události lze vyvodit, že útočník k prolomení hesla využil některý z automatizovaných nástrojů. Z SSH logu byly zjištěny uživatelská jména, která byla k útoku použita. V následující tabulce č.3 je zobrazeno deset nejvyužívanějších uživatelských jmen, která byla při útoku využita. Tyto loginy, neboli uživatelská jména jsou obsažena v textových souborech, kterých je využito při slovníkovém útoku. Z tabulky je patrné, že nejvíce zkoušeným uživatelským jménem při tomto konkrétním útoku bylo jméno *root*. Toto jméno označuje ve světě GNU/Linux systémů uživatele s nejvyššími právy.

Přihlašovací jméno	Počet použití
root	84
mysql	36
test	28
user	23
admin	19
mysql	17
guest	17
info	14
www	13
database	8

Tabulka 3 - Seznam nejčastěji použitých loginů během útoku – vlastní zpracování

Útočník se následně odmlčel a po dobu 40 hodin nebyly na honeypotu pozorovány žádné události, které by měly spojitost s daným útokem. Ze stejné IP adresy se útočník znovu připojil dne 20.11. 2018 ve 23 hodin – již se známými přihlašovacími údaji. Všechny kroky, které útočník po přihlášení učinil, byly úspěšně zachyceny modulem Sebek a odeslány na komponentu Sebek-server kde byly uloženy v podobě PCAP souboru. Na základě těchto zjištění byla následně provedena rekonstrukce Sebek PCAP souboru pomocí komponent *sbk_extract* a *sbk_ks_log.pl*. Výstup je zobrazen na obrázku č.23. Z obrázku je patrné, že modul Sebek dokáže zaznamenat pouze příkazy, které zadal útočník, avšak nezobrazuje vrácené hodnoty těchto příkazů. Pokud útočník použije během zadávání příkazu klávesy mimo znaky klávesnice, jsou tyto příkazy zobrazovány v hranatých závorkách – například šipka nahoru má označení [U-ARROW]. Každý řádek zobrazený pomocí nástroje *sbk_ks_log.pl* má následující tvar:

- [TIMESTAMP IP_ADDRESS PID COMMAND UID] Text
- TIMESTAMP – datum a čas zadání příkazu
- IP_ADDRESS – IP adresa honeypotu
- PID – je ID procesu
- COMMAND – prvních 10 znaků názvu příkazu
- UID – uživatelské ID vlastníka procesu



Obrázek 23 - Rekonstrukce Sebek paketu zobrazující jednotlivé keystrokes – vlastní zpracování

Útočník začal s tím, že si zjistil pod jakým uživatelem je přihlášený a zjistil si informace o uživatelích systému a běžících procesech. Dále si zjistil informace o systému a nechal si vypsat informace o procesoru. Následně změnil heslo uživatele a prozkoumal běžící procesy, načež jeden z procesů ukončil. Poté se pokusil přepnout na uživatele root, což se mu dle stále stejného UID nepovedlo. Poté vytvořil skrytou složku ' ' v adresáři *tmp* a do této složky stáhnul nástroj *emech* který následně rozbalil a spustil. Před odpojením od systému ještě smazal historii zadaných příkazů a následně se odpojil.

Při vyšetřování nástroje emech bylo zjištěno, že se jedná o IRC bota, který slouží ke vzdálenému ovládání počítače. Následně byla provedena analýza síťových toků a bylo zjištěno, že honeypot udržuje spojení se vzdáleným C&C serverem. Honeypot byl tedy úspěšně kompromitován a stal se z něho takzvaný *zombie* v síti botnet. Botnet je síť napadených systémů které jsou ovládány vlastníkem botnetu – botmasterem. K ovládání počítačů v síti botnet se využívá protokol IRC který je primárně určen k textové komunikaci v reálném čase. IRC sítě jsou vybudovány na architektuře klient – server, kdy uživatelé odesílají a přijímají zprávy pomocí IRC klienta instalovaného na jejich počítači. Klient odesílá zprávy na server a ten zprávu přepošle příjemci. Standardně protokol IRC využívá při komunikaci port 6667/TCP a porty v rozsahu 6660 - 7000. Zprávy, které jsou posílány skrze IRC, nejsou nijak šifrovány a lze je snadno dekódovat ze zachycených paketů. Při navazování spojení mezi uživatelem a serverem je v IRC protokolu definováno několik speciálních příkazů. Tyto příkazy neslouží ke komunikaci mezi uživateli, ale k ověření uživatele vůči serveru:

- *Pass message* – zpráva obsahující heslo. Tato zpráva je odeslána v případě, pokud je vyžadováno připojení k serveru
- *Nick message* – pomocí tohoto příkazu je nastaven nový nickname uživatele, popřípadě změna stávajícího
- *User message* – pomocí tohoto příkazu je specifikován username, hostname a opravdové jméno uživatele

Po připojení uživatele k serveru je možné pomocí příkazu *Join* připojení k určitému kanálu. Veškeré zprávy odeslané v tomto kanálu, jsou viditelné všemi uživateli, kteří jsou taktéž připojeni k tomuto kanálu. Příkaz PING/PONG slouží k ověření, zda je daný klient stále připojen k serveru – server odešle příkaz ping a čeká na odpověď pong. Z analýzy paketů bylo zjištěno, že automatizovaný IRC bot vytvořil nového uživatele: *x6337* který byl následně připojen do sítě: *ass IRC network*. Tato síť obsahovala 1952 uživatelů. Spojení bylo udržováno a ověřováno pomocí příkazů PING/PONG. V tomto stavu tedy honeypot čekal na příkaz od botmastera, který mohl zaslat příkaz k určitému útoku. Na obrázku č.24 je část paketu, který zobrazuje komunikaci PING/PONG mezi honeypotem – 192.168.1.25 a C&C serverem 146.185.171.227. Dle IP adresy se tento server nachází v Amsterdamu.

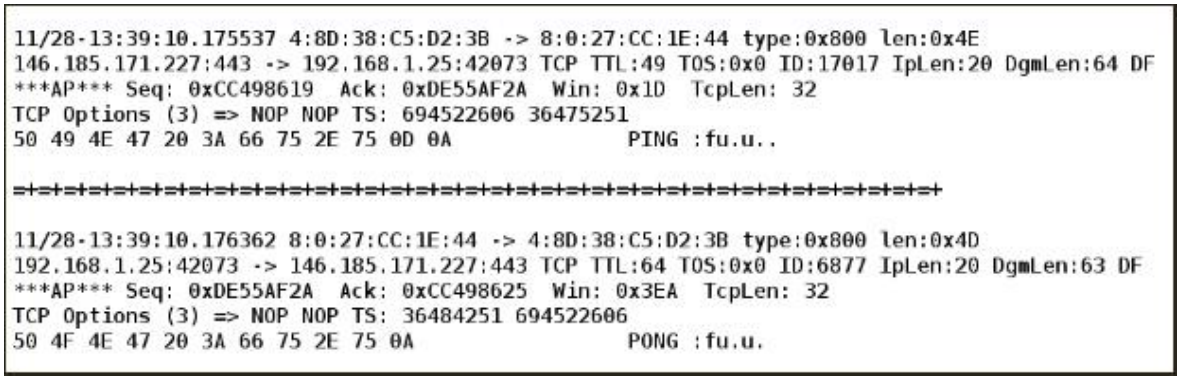
```
11/22-07:47:49.092397 4:8D:38:C5:D2:3B -> 8:0:27:CC:1E:44 type:0x800 len:0x4E
146.185.171.227:6665 -> 192.168.1.25:46074 TCP TTL:49 TOS:0x0 ID:17358 IpLen:20 DgmLen:64 DF
***AP*** Seq: 0x581FF911  Ack: 0x92A097E5  Win: 0x1D  TcpLen: 32
TCP Options (3) => NOP NOP TS: 559060096 49190162
50 49 4E 47 20 3A 66 75 2E 75 0D 0A              PING :fu.u..

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

11/22-07:47:49.093328 8:0:27:CC:1E:44 -> 4:8D:38:C5:D2:3B type:0x800 len:0x4D
192.168.1.25:46074 -> 146.185.171.227:6665 TCP TTL:64 TOS:0x0 ID:11746 IpLen:20 DgmLen:63 DF
***AP*** Seq: 0x92A097E5  Ack: 0x581FF91D  Win: 0x3EA  TcpLen: 32
TCP Options (3) => NOP NOP TS: 49191252 559060096
50 4F 4E 47 20 3A 66 75 2E 75 0A                 PONG :fu.u.

=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Obrázek 24 - Komunikace mezi C&C serverem a honeypotem pomocí portu 6665 – vlastní zpracování

Po tomto zjištění byl honeypot odstaven a byl obnoven ze snapshotu tak, aby nemohl být zneužit k útokům na další systémy. Při analýze IRC paketů nebyl zjištěn žádný příkaz, který by byl odeslán na honeypot a který by sloužil ke spuštění útoku. Po dobu analýzy útoku tedy honeypot vyčkával na příkazy od útočníka. Je vysoce pravděpodobné, že by k nějakému útoku mohlo dojít, pokud by honeypot nebyl odstaven. Počítače, které jsou napadeny

některou formou IRCbota a jsou připojeny do botnetu, se velmi často zneužívají k útokům typu DDoS či k rozesílání spamu.

Odhalení komunikace mezi honeypotem a C&C serverem bylo v tomto případě snadné, neboť IRCbot využíval ke komunikaci port 6665. Poté, co byl honeypot odstaven a znovu obnoven ze snapshotu, byl během následujících šesti dnů zaznamenán další útok vedený ze stejné IP adresy, který měl shodný průběh, avšak útočník využil maskování komunikace mezi honeypotem a C&C serverem tím způsobem, že IRCbot využíval ke komunikaci port 443. Tento port je standardně používán protokolem HTTPS a při běžné kontrole síťové komunikace nevzbuzuje podezření, že se jedná o komunikaci mezi IRCbotem a C&C serverem. Na obrázku č.25 je část paketu zobrazující komunikaci typu PING/PONG mezi honeypotem 192.168.1.25 a C&C serverem 146.185.171.227. V tomto případě je však komunikace vedena pomocí portu 443. Samotný průběh útoku byl velmi podobný útoku, který byl zaznamenán o 6 dní dříve, tedy 22.11. 2018. Útočník se po prolomení hesla na 24 hodin odmlčel a následně se připojil na honeypot, v tomto případě z jiné IP adresy. Zjistil si informace o uživatelích systému, zjistil si informace o systému a pomocí příkazu *wget* stáhl IRCbota do skryté složky. Následně bota rozbalil a spustil, poté smazal historii zadaných

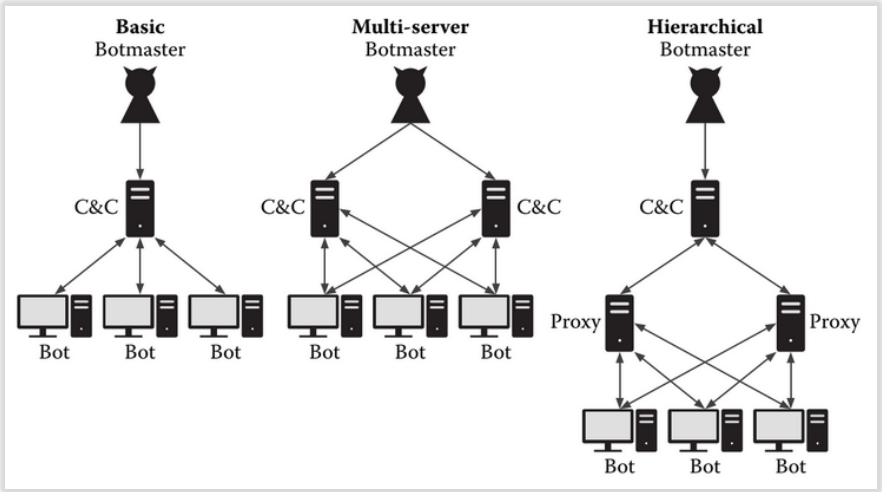


Obrázek 25 - Komunikace mezi C&C serverem a honeypotem pomocí portu 443 – vlastní zpracování

příkazů a odhlásil se. Po následné analýze a zjištění, že honeypot udržuje komunikaci typu PING/PONG se vzdáleným serverem, byl honeypot odstaven ještě předtím, než byl zneužit k dalšímu útoku.

Z této forenzní analýzy lze vyvodit, jaké měl útočník k provedení tohoto útoku motivy. Dle kroků, které po napadení honeypotu učinil, lze říci, že se s největší pravděpodobností jedná o botmastera – tedy o člena blackhat komunity, který je vlastníkem jednoho nebo i více botnetů. Tito botmasteři se snaží infikovat další systémy, neboť čím větší počet uzlů daný botnet obsahuje, tím je botnet silnější z hlediska útoků na další systémy. Botnet se

zpravidla využívá při DDoS útocích či k rozesílání spamu (32). Architektura botnetu může být dvojího typu. Prvním typem je takzvaná centralizovaná architektura, která je charakteristická tím, že C&C server má centralizovanou lokaci. Druhým typem je architektura decentralizovaná, která nemá centrální C&C server který by ovládal jednotlivé *boty*, ale využívá k zasílání příkazů P2P komunikaci. Vzhledem k tomu, že infikovaný honeypot komunikoval s C&C serverem napřímo, lze vyvodit, že se jednalo o botnet centralizovaného typu. Centralizované botnety se dále dělí do tří typů. Prvním typem je takzvaný základní botnet, který využívá pouze jeden C&C server, druhý typ je takzvaný multi-serverový botnet, který k ovládání *botů* používá více C&C serverů a třetím typem je hierarchický botnet, který využívá proxy, mezi C&C serverem a jednotlivými *boty*. Obrázek č.26 zobrazuje jednotlivé architektury centralizovaných botnetů. Během forenzní analýzy byla odhalena komunikace mezi honeypotem a C&C serverem, která odpovídala základnímu typu botnetu – komunikace probíhla pouze mezi honeypotem a jedním C&C serverem.



Obrázek 26 - Architektury centralizovaných botnetů (35)

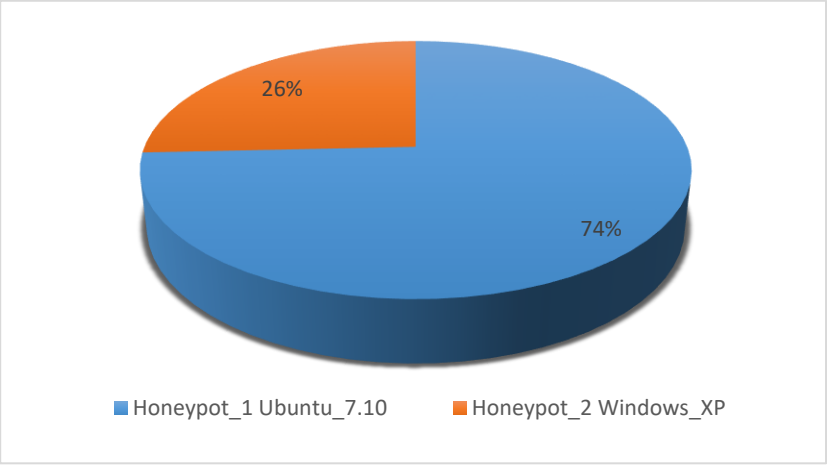
5.2 Statistická analýza

Tato část práce zobrazuje souhrnné statistické údaje z období mezi listopadem a prosincem 2018, kdy byly honeypoty spuštěny. Paralelní provoz honeypotů nebyl možný vzhledem k tomu, že celý systém byl závislý na jedné veřejné IP adrese. Provoz byl tedy rozdělen na dvě části. V první části byl spuštěn honeypot_1 se systémem Ubuntu 7.10 server a IP adresou 192.168 1.25, tento honeypot byl spuštěný po dobu 30 dní. Po uplynutí této doby byl honeypot odstaven a byl nasazen honeypot_2 se systémem Windows XP a IP adresou 192.168.20, který byl spuštěn taktéž po dobu 30 dní. Tabulka č.4 zobrazuje počty pokusů o navázání spojení s jednotlivými honeypoty. Vzhledem k povaze chování

Honeypot	**Počet pokusů o spojení**
Honeypot_1, Ubuntu 7.10	42 265
Honeypot_2, Windows XP	14 632
Celkem	**56 897**

Tabulka 4 - Počet pokusů o připojení na jednotlivé honeypoty – vlastní zpracování

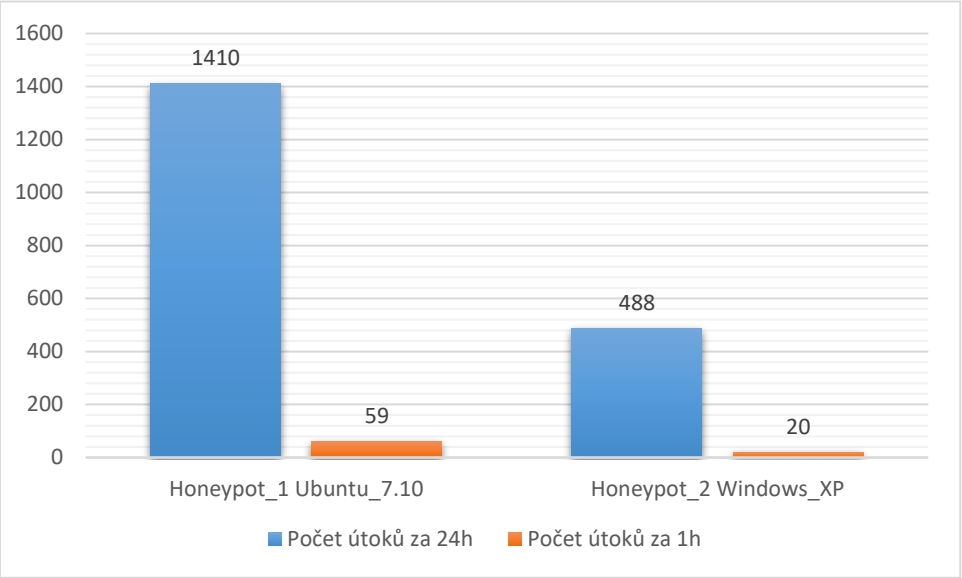
honeypotu lze tedy říci, že se jedná o počty potencionálních útoků. Na honeypot_1 bylo vedeno téměř třikrát více útoků než na honeypot_2. Dohromady bylo během experimentu, který trval 60 dnů zaznamenáno 56897 potencionálních útoků. Na obrázku č.27 je zobrazen graf, který vyjadřuje množství útoků na jednotlivé honeypoty v procentech. Celých 74% ze všech útoků bylo mířeno na honeypot s operačním systémem Ubuntu 7.10. Tento systém byl



Obrázek 27 – Procentuální vyjádření množství útoků na jednotlivé honeypoty – vlastní zpracování

během experimentu mnohem více lákavým cílem nežli honeypot s operačním systémem

Windows XP. Na obrázku č.28 je znázorněn graf, který vypovídá o průměrném počtu útoků na jednotlivé honeypoty během stanoveného časového období. Z grafu je patrné, že



Obrázek 28 - Průměrný počet útoků na honeypoty během stanoveného časového úseku – vlastní zpracování

průměrný počet útoků, který byl během 24 hodin veden na honeypot s operačním systémem Ubuntu 7.10, byl 1410 a průměrný počet útoků za 1 hodinu byl 59. Na honeypot s operačním systémem Windows XP bylo v průměru každých 24 hodin vedeno 488 útoků a každou hodinu 20 útoků.

Pohledem do tabulky č.5 můžeme zjistit, jaké množství potencionálních útoků bylo na honeypoty vedeno dle typu komunikačního protokolu. Z tabulky jasně vyplývá, že nejvíce

Protokol	**Počet pokusů o připojení**	**Procentuální vyjádření**
TCP	51813	91%
UDP	931	2%
ICMP	4253	7%
Celkem	**56897**	**100%**

Tabulka 5 - Počty připojení dle protokolu – vlastní zpracování

útočníky využívaný protokol je protokol TCP. Během experimentu byl tento protokol využit při 91% všech útoků, které byly na honeypoty vedeny. Tato převaha je způsobena tím, že množství služeb a aplikací používá pro komunikaci právě tento protokol v porovnání s protokolem UDP. Protokol TCP – *transmission control protocol* – je protokol z transportní vrstvy OSI modelu, díky kterému mohou aplikace mezi sebou navázat spojení. Tento

protokol garantuje spolehlivé doručení a doručení ve správném pořadí jednotlivých datagramů. Naproti tomu, protokol UDP – *user datagram protocol* – spolehlivost a správnost spojení negarantuje. Protokol ICMP – *internet control message protocol* – je využívám operačními systémy pro ověření, zda je požadovaný počítač či router dostupný, nebo zda je dostupná určitá služba.

V tabulce č.6 jsou zobrazeny počty připojení na jednotlivé porty. Jak je vidět, nejvíce napadaným portem byl port 22, na kterém běží služba SSH. Útoky vedeny vůči tomuto portu byly vesměs slovníkové útoky, které se snažily získat neautorizovaný přístup ke vzdálenému honeypotu. Dalším velmi často napadaným portem byl port č.80 na kterém běží služba HTTP. Útoky na tento port byly vyhodnoceny modulem Snort jako pokusy o přístup k potencionálně zranitelné webové aplikaci, dále jako potencionální *worm-attack* a dalším typem vyhodnoceného útoku byl také pokus o únik informací. Třetím napadaným portem v pořadí byl port č.3306 na kterém běží služba MySQL. Útoky vůči tomuto portu byly vyhodnoceny modulem Snort jako *MYSQL client authentication bypass attempt.*

Služba	**Port**	**Počet pokusů o připojení**
SSH	22	38532
HTTP	80	7312
MySQL	3306	5824
IRC	6667	3331
FTP	21	868
POP3	110	556
SMTP	25	374
IMAP	143	51
IPP	631	23
XMP	5269	23

Tabulka 6 - Počet pokusů o připojení na jednotlivé služby/porty – vlastní zpracování

Díky logování událostí a především díky logování autentizace služby SSH do logfilu /var/log/*auth.log* v systému Ubuntu 7.10 server, bylo možné provést analýzu přihlašovacích jmen, která byla při slovníkových útocích vůči této službě zkoušena. Z celkového počtu 38532 útoků na službu SSH, bylo zaznamenáno 27328 slovníkových útoků, které pocházely z 31 různých IP adres. V tabulce č.7 jsou zobrazena uživatelské jména, která byla během těchto útoků nejčastěji využívána jako přihlašovací jméno do systému.

Uživatelské jméno	Počet použití při útocích na SSH
root	8635
test	4325
admin	4021
oracle	1532
user	943
mysql	651
guest	423
qwert	413
bin	364
web	213

Tabulka 7 - Deset nejvíce používaných uživatelských jmen při útocích na SSH – vlastní zpracování

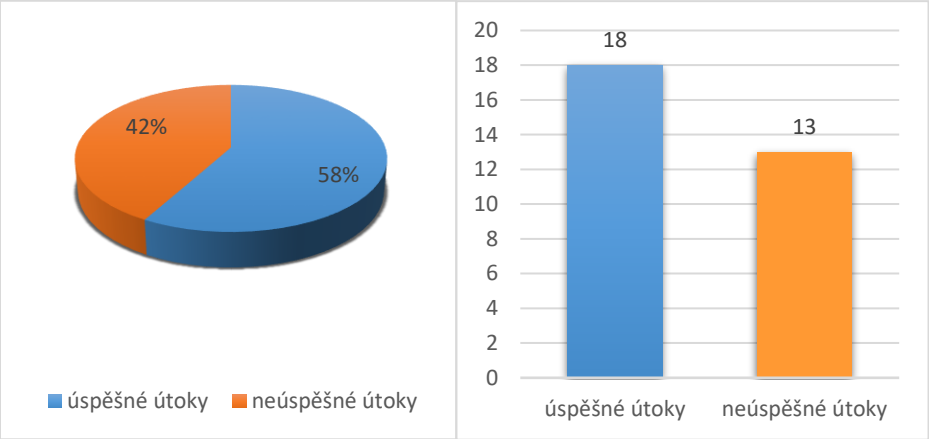
V tabulce č.8 jsou zobrazeny statistické údaje z těchto slovníkových útoků. Dohromady bylo zaznamenáno 27328 pokusů o zadání uživatelského jména a hesla. Tyto útoky pocházely z 31 různých IP adres. Slovníkových útoků bylo tedy 31, přičemž průměrná délka jednoho útoku byla 43minut a průměrný počet vyzkoušených kombinací (přihlašovací jméno + heslo) bylo 882 v průběhu jednoho útoku.

Počet slovníkových útoků	Průměrná doba útoku (min)	Průměrný počet pokusů při jednom útoku
31	43	882

Tabulka 8 - Statistické údaje o slovníkových útocích na SSH – vlastní zpracování

Z těchto 31 slovníkových útoků bylo zaznamenáno 18 úspěšných útoků – tedy došlo k prolomení autentizace a autorizace a útočník získal plný přístup do systému skrze vzdálený

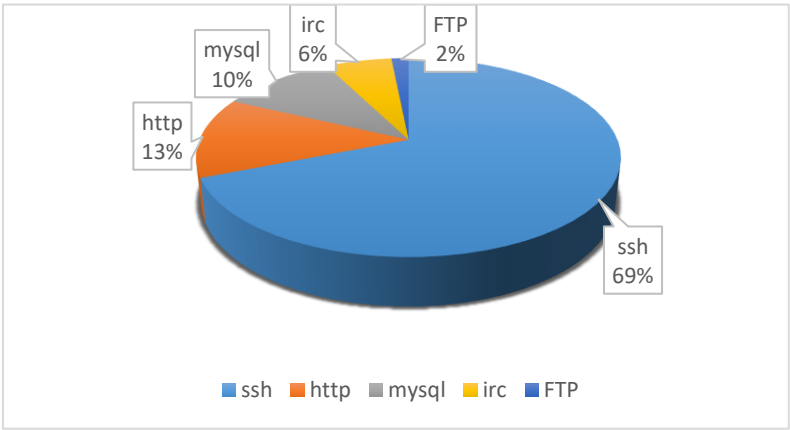
terminál – SSH. Na následujícím obrázku č.29 jsou zobrazeny grafy, které vyjadřují procentuální a absolutní počty úspěšných i neúspěšných útoků.



Obrázek 29 - Úspěšné a neúspěšné útoky vůči SSH – vlastní zpracování

Z obrázku je patrné, že v 58% slovníkových útoků vůči službě SSH byli útočníci úspěšní, a podařilo se jim získat plný přístup ke vzdálenému systému – honeypotu s operačním systémem Ubuntu 7.10.

Na obrázku č.30 je znázorněn graf, který procentuálně zobrazuje pět služeb, které byly nejvíce napadány. Jak je vidět, útoky na službu SSH tvoří 69% ze všech celkových útoků na oba honeypoty. Služba SSH je nejvíce lákavá ze strany útočníků především proto, že při prolomení přihlašovacích údajů získá útočník přístup k systému, což mu dává mnoho možností, jak daný systém zneužít ve svůj prospěch. Útoky vůči této službě se podařilo úspěšně zachytit, a mohla být tak provedena forenzní analýza chování útočníka. Tato analýza



Obrázek 30 - Procentuální vyjádření nejvíce napadaných služeb – vlastní zpracování

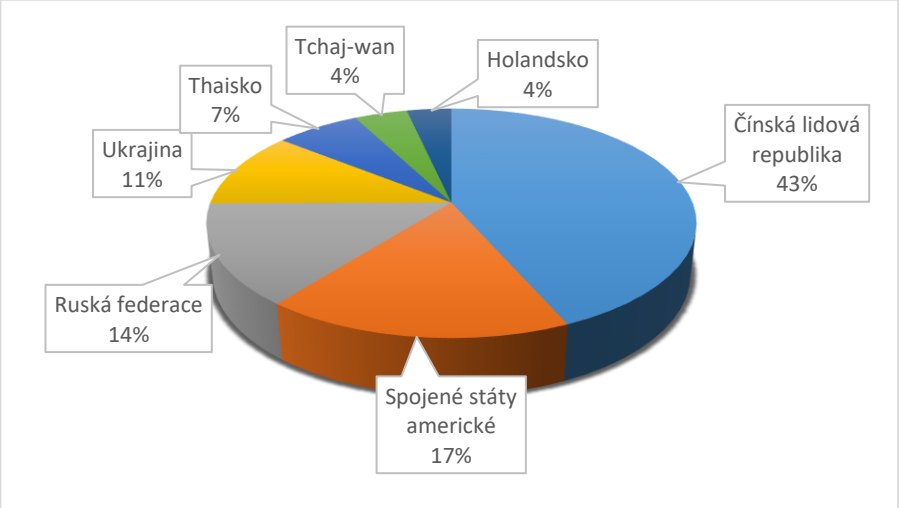
je popsána v kapitole 4.3, kde je analýza demonstrována právě na jednom z mnoha útoků vedených vůči službě SSH.

Po dobu, co byly honeypoty nasazeny, byly pokusy o navázání spojení s nimi zaznamenány z IP adres, které pocházely ze zemí celého světa. V následující části jsou představeny ty IP adresy, z kterých bylo zaznamenáno největší množství pokusů o připojení. Během experimentu bylo zaznamenáno množství útoků, které byly opakovaně vedeny z konkrétních IP adres. I přes to, že tyto IP adresy známe a známe i jejich geografickou polohu, nelze dle těchto údajů objektivně posoudit, zdali byl útok veden skutečně z dané země. Důvodem této nejednoznačnosti je fakt, že útočníci se při provádění takovýchto útoků na informační systémy snaží zůstat v anonymitě a proto využívají technik, jako je například přístup k internetu přes *proxy*. Tato proxy se může nacházet na území jiného státu, než ve kterém se nachází útočník, a tak stopy povedou pouze k této proxy. Nesmíme také opomenout fakt, že útok mohl být veden z nějaké veřejné sítě, a proto stopy povedou pouze k veřejné IP adrese této sítě, ale k IP adrese skutečného útočníka se již nedostaneme. Díky této analýze však můžeme zjistit, jací internetoví poskytovatelé mají dané IP adresy ve správě, a lze je tak upozornit, že se za jejich IP adresami schovávají útočníci z řad blackhat komunity.

IP adresa	Počet pokusů o připojení	Země, z které IP pochází
209.141.51.85	3552	Spojené státy americké
58.218.92.26	2946	Čínská lidová republika
116.31.116.47	2741	Čínská lidová republika
5.188.210.12	2705	Ruská federace
193.201.224.199	1863	Ukrajina
116.31.116.49	1359	Čínská lidová republika
171.96.126.127	742	Thaisko
123.249.0.132	521	Čínská lidová republika
61.216.152.133	498	Tchaj-wan
37.139.5.191	423	Holandsko

Tabulka 9 - Deset nejaktivnějších IP adres a jejich geografická lokace – vlastní zpracování

Z tabulky č.9 je patrné, že nejvíce pokusů o připojení, které byly vedeny z jedné IP adresy, pocházely z IP adresy, která je lokalizovaná na území Spojených států amerických, konkrétně ze státu Nevada a města Las Vegas. Tato IP adresa je registrována u poskytovatele FranTech solutions. Z této IP adresy byl také zaznamenán úspěšný útok vůči službě SSH který vedl k tomu, že byl honeypot úspěšně kompromitován a byl na něm nainstalován IRCbot. Na obrázku č.31 je zobrazeno procentuální zastoupení jednotlivých zemí, z kterých byly útoky vedeny. Z grafu je patrné, že nejvíce útoků – celých 44% z nejaktivnějších IP adres bylo vedeno z Čínské lidové republiky. IP adresa 58.218.92.26 pochází z regionu Jiangsu a města Xuzhou a je registrována u poskytovatele Chinanet. IP adresy 116.31.116.47 a 116.31.116.49 pochází z regionu Guangdong a města Foshan. Tyto IP adresy jsou registrovány taktéž u poskytovatele Chinanet. IP adresa 123.249.0.132 pochází z regionu Guangdong města Guangzhou a je taktéž registrována u Chinanet. 20% útoků bylo vedeno ze Spojených států amerických z Las Vegas v Nevadě, 16% útoků bylo vedeno z IP 5.188.210.12, která se nachází na území Ruské federace, konkrétně z města Saint Petersburg a je registrována u poskytovatele Petersburg Internet, 11% útoků bylo vedeno z IP 193.201.224.199, která se nachází na území Ukrajiny ve městě Kiev a je registrována u poskytovatele PE Tetyana Mysyk, 4% útoků bylo vedeno z IP adresy 171.96.126.127, která se nachází na území Thaiska, v hlavním městě Bangkok a je registrována u poskytovatele True Internet Corporation, 3% útoků bylo vedeno z IP adresy 61.216.152.133 která se



Obrázek 31 - Procentuální vyjádření geografické lokace 10 nejaktivnějších IP adres – vlastní zpracování

nachází na území Tchaj-wanu z města Taipeia, a je registrována u poskytovatele Data Communication Business Group, 2% útoků bylo vedeno z IP adresy 37.139.5.191 která se

nachází na území Holandska z města Amsterdam, a je registrována u poskytovatele DigitalOcean.

5.3 Diskuse

Tato diplomová práce poskytuje podrobný pohled na problematiku počítačové bezpečnosti, konkrétně na bezpečnost komunikace v prostředí počítačových sítí. Pomocí nástroje Honeywall byl vytvořen systém s vysokointeraktivními honeypoty, který byl nasazen v reálném provozu. Během tohoto experimentálního provozu byly zaznamenány desítky tisíc potencionálních útoků, které byly vedeny z míst po celém světě a byly vedeny vůči službám, které jsou běžně dostupné na veřejných IP adresách – například služby jako HTTP, HTTPS, MySQL, POP3, SMTP a jiné. Největší množství útoků bylo vedeno vůči službě SSH, jednalo se především o slovníkové útoky. Z těchto slovníkových útoků bylo 58% úspěšných – došlo k prolomení autentizace a útočník získal přístup do systému. Díky navrženému systému se tyto útoky podařilo zachytit a následně velmi podrobně analyzovat. Získali jsme tak ucelený pohled do mysli útočníků z řad blackhat komunity, odhalili jsme jejich motivy, nástroje a cíle útoků.

Takto navržený systém lze využít například při studiu nových nástrojů a postupů, které blackhat komunita při svých útocích využívá. Ze zachycených útoků tak lze získat nástroje, které ještě nejsou zdokumentovány, a není známá jejich funkcionalita. Pokud bude navržený systém disponovat neodolatelnými návnadami pro blackhat komunitu, a systém bude spuštěn po dostatečně dlouhou dobu, získáme velké množství škodlivých nástrojů v podobě *rootkit, exploit, backdoor*, a jiných druhů *malware,* z nichž určité množství bude v podobě takzvaných *zero-day* hrozeb, proti kterým doposud neexistuje žádná ochrana, jelikož jsou neznámé. Na základě následné analýzy těchto *zero-day* nástrojů, získáme cenné informace o tom, jak daný konkrétní *malware* funguje a vydáme bezpečnostní opatření – například v podobě signatur pro IDS a IPS systémy. Díky těmto bezpečnostním opatřením přispějeme k celkové bezpečnosti v prostředí počítačových systémů a z daných *zero-day* zranitelností se stanou již známé zranitelnosti, vůči kterým existuje ochrana.

Systém lze také využít v celkové obraně síťové infrastruktury organizace. Pokud bude systém zapojen v prostředí vnitřní sítě, bude velmi snadné díky povaze systému zachytit malware šířící se po síti či útočníka, který provádí scan počítačové sítě a připravuje útok. Systém dokáže zachytit podezřelou síťovou komunikaci bez toho, aniž by logoval veškerý

síťový provoz. Z tohoto důvodu je pro bezpečnostního administrátora snadné odhalit nebezpečnou aktivitu, aniž by musel procházet obrovské množství logů, které poskytují jiné bezpečnostní prvky jako například NetFlow sonda. Díky této funkcionalitě je tento systém vhodným nástrojem pro zajištění celkové bezpečnosti v prostředí počítačových sítí jednotlivých organizací.

Data, která byla během experimentálního provozu zaznamenána, mají svojí hodnotu především v tom smyslu, že díky nim lze vyhodnotit, jaké hrozby v prostředí celosvětové veřejné počítačové sítě – Internetu číhají a jaké nástroje a metody útočníci při svých průnicích do systémů využívají. Hrozby, kterým počítačové systémy čelí, pocházejí opravdu z koutů celého světa a tyto hrozby jsou mířeny na každou veřejně dostupnou IP adresu a každý port, respektive službu, která je na této IP adrese poskytována. I přes to, že je kybernetická bezpečnost stále více diskutovaným tématem, pořád se najdou administrátoři, kteří nedbají základních pravidel pro zabezpečení systémů a volí slabá, nedostatečně dlouhá a málo složitá hesla, která mohou být již známá a mohou být obsahem slovníků, které útočníci při svých útocích používají. Nejslabším článkem v zabezpečení systémů tak stále zůstává člověk a jeho laxnost a nepozornost, kterou blackhat komunita zneužívá ke svému prospěchu. Z nasbíraných dat je patrné, že množství útoků na služby je opravdu vysoké a útočníci spoléhají na to, že narazí na špatně zabezpečené služby, vůči kterým bude jejich útok úspěšný. Jedním ze základních obranných prvků každého systému je dostatečně silné neprolomitelné heslo. Administrátoři by měli volit taková hesla, která nebudou založena na slovnících, budou složitá a dostatečně dlouhá. Pokud se na nasbíraná data a statistiky podívá jakýkoli administrátor, měl by se minimálně zamyslet nad tím, jaká hesla on sám při zabezpečování systémů používá, a měl by přehodnotit to, jakým způsobem na zabezpečení systémů nahlíží. Pokud i on je ze skupiny administrátorů, kteří neberou bezpečnost příliš vážně, možná díky těmto nasbíraným datům svůj pohled na věc přehodnotí a bude obhospodařovávané systémy lépe zabezpečovat.

Hodnota provedené forenzní analýzy zachyceného útoku spočívá především v tom, že na základě této konkrétní analýzy lze vytvořit bezpečnostní opatření, například pomocí již zmíněných signatur do IDS a IPS systémů. Bezpečnostní opatření, která budou na základě této analýzy vytvořena, jsou do jisté míry ovlivněna tím, pro jaké prostředí budou opatření vytvořena a pomocí jakých bezpečnostních prvků budou implementována. Tato bezpečnostní pravidla jsou již individuální a každý bezpečnostní administrátor je bude implementovat odlišným způsobem v závislosti na tom, jakým způsobem bude s útokem

tohoto typu nakládat. Závěrem lze říci, že na základě této forenzní analýzy lze vytvořit bezpečnostní opatření, která dokáží odhalit komunikaci mezi C&C serverem a napadeným systémem a dokáží na tuto komunikaci upozornit a následně jí blokovat. Tím pádem nebude možné daný systém dále zneužívat pro útoky typu DDoS či rozesílání SPAMU. Lze také doporučit opatření, která znesnadní či přímo znemožní útočníkům danou službu prolomit. Například použitím již zmiňovaných silných hesel, nebo využití certifikátů, pomocí kterých bude zajišťována autentizace vůči službě SSH.

6 Závěr

Hlavním cílem této práce byla aplikace forenzní analýzy chování útočníka, který pronikl do napadeného systému. Aby mohla být tato analýza provedena, musely být zaznamenány jednotlivé kroky, které útočník po průniku do systému učinil. K tomu, aby mohly být tyto kroky zaznamenány, byl vytvořen systém s vysokointeraktivními honeypoty v odděleném segmentu domácí sítě, který byl obstarán veřejnou IP adresou.

V práci je popsán implementovaný systém, v kterém hraje klíčovou roli nástroj Honeywall, což je nástroj, který je založen na operačním systému GNU/Linux distribuce CentOS a vystupuje jako síťový most mezi jednotlivými honeypoty a gateway. Veškerý síťový provoz tedy prochází skrze tento nástroj, který obsahuje řadu dílčích nástrojů jako Snort IDS, IPS, NetFilter a další, pomocí kterých lze zachytávat jednotlivé pokusy o připojení, lze vytvářet signatury pro IDS a IPS, nastavovat firewall pravidla pro daná spojení a v neposlední řadě lze zaznamenávat jednotlivé pakety a následně je analyzovat. Druhým neméně důležitým nástrojem je jaderný modul Sebek, který slouží k monitorování a zaznamenávání jednotlivých *keystrokes,* které útočník v napadeném systému učinil. Díky tomuto modulu bylo možné provést rekonstrukci jednotlivých kroků, které útočník po napadení systému učinil.

Provoz implementovaného systému byl rozdělen do dvou fází, a to z toho důvodu, že systém byl závislý na jedné veřejné IP adrese a implementované vysokointeraktnivní honeypoty byly dva. V první fázi byl tedy provoz po dobu 30 dní směřován na honeypot s distribucí operačního systému GNU/Linux – Ubuntu 7.10 server a v druhé fázi, byl provoz směřován na honeypot s operačním systémem Windows XP SP2 taktéž po dobu 30 dnů. Na těchto honeypotech byly nainstalovány zastaralé verze služeb, které obsahují známe zranitelnosti, a mohly se tak stát snadným terčem útoků.

Za dobu, co byly honeypoty spuštěny v experimentálním provozu, bylo zaznamenáno dohromady 56 897 pokusů o navázání spojení z IP adres z celého světa. Nejvíce aktivní IP adresa – tedy adresa, z které bylo zaznamenáno největší množství útoků byla adresa 209.141.51.85, která se nachází na území Spojených států amerických ve státě Nevada. Nejvíce pokusů o navázání spojení bylo zaznamenáno na službu SSH – 69% ze všech pokusů o navázání spojení, tedy přesně 38532 potencionálních útoků. Útoky na tuto službu byly především slovníkové útoky, kdy se útočník snaží uhádnout přihlašovací jméno a heslo pomocí takzvaných slovníků, které obsahují množství různých loginů a hesel přičemž automatizovaný nástroj zkouší dosazovat jednotlivé kombinace z těchto slovníků do

přihlašovacího pole. Díky logování služby SSH na úrovni operačního systému bylo zjištěno, jaká uživatelská jména útočníci při svých útocích používali. Nejvíce využívaným přihlašovacím jménem bylo jméno *root,* které ve světě operačních systémů GNU/Linux označuje uživatele s nejvyššími právy, dále pak jména *test* a *admin.* K prolomení hesla došlo v 18 případech a díky modulu Sebek bylo možné provést rekonstrukci *keystrokes.*

Forenzní analýza útoků vůči službě SSH odhalila jednotlivé kroky, které útočníci po napadení systému učinili. Forenzní analýza je demonstrována na jednom takovém útoku, při kterém se útočník po prolomení hesla na 40 hodin odmlčel a po této době zahájil útok – po přihlášení se do systému si zjistil základní informace o uživatelích systému, o běžících procesech a o systému jako takovém. Následně změnil heslo uživatele a vytvořil skrytou složku, do které stáhl nástroj *emech,* tento nástroj posléze spustil. Poté smazal historii zadaných příkazů a od systému se odhlásil. Při následné analýza nástroje *emech* bylo zjištěno, že se jedná o IRCbota který udržoval spojení se vzdáleným C&C serverem a vyčkával na další instrukce od útočníka – z honeypotu se tak stal *bot* v síti *botnet* a mohl být útočníkem dále využíván při útocích na jiné systémy.

Implementovaný systém i přes některé své nedostatky dokáže úspěšně monitorovat aktivitu blackhat komunity a může být v této podobě nasazen do organizací, které se zaměřují na výzkum blackhat komunity, výzkum nástrojů a postupů, která tato komunita k prolamování systému využívá, jak maskuje své kroky a k čemu napadené systémy využívá. Díky schopnostem systému zachytit i takzvané *zero-day* zranitelnosti se tak systém stává velmi cenným zdrojem poznání, s jehož použitím lze zvyšovat celkovou bezpečnost informačních systémů a lze vytvářet bezpečnostní opatření vůči hrozbám, které nebyly doposud objeveny. Systém může být nasazen i v komerčních organizacích či v orgánech státní správy jako další rozšiřující prvek celkového zabezpečení. Díky tomu, že se jedná o nástroje, které jsou šířeny pod GPL licencí, může být tato varianta zajímavým řešením.

7 Seznam použitých zdrojů

7.1 Seznam literatury

1. Internet usage statistisc. *InternetWorldStats.* [Online] 30. Červen 2018. [Citace: 15. Únor 2019.] https://www.internetworldstats.com/stats.htm.

2. Cyber crime: attacks experienced by companies worldwide 2017. *statista.* [Online] srpen 2017. [Citace: 13. červenec 2018.] https://www.statista.com/statistics/474937/cyber-crime-attacks-experienced-by-global-companies/.

3. Cyber definitoins. *ccdcoe.* [Online] [Citace: 18. Srpen 2018.] https://ccdcoe.org/cyber-definitions.html.

4. Schaeffer, Richard C. *National Information Assurance Glossary. CNSS.* [Online] 26. Březen 2010. [Citace: 26. Srpen 2018.] https://www.hsdl.org/?view&did=7447.

5. Healey, Jason. *A Fierce Domain: Conflict in Cyberspace, 1986 to 2012.* Vienna : Cyber Conflict Studies Association, 2013. 098932740X.

6. Jirovský, Václav. *Kybernetická kriminalita, nejen o hackingu, crackingu, virech a trójských koních bez tajemství.* Praha : Grada Publishing a.s, 2007. 978-80-27-1561-2.

7. Lin, Tom C. W. Financial Weapons of War. [Online] 14. Duben 2016. [Citace: 26. Srpen 2018.] https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2765010.

8. Difference Between Active and Passive Attacks. *techdiffrences.* [Online] 2. Únor 2018. [Citace: 31. Srpen 2018.] https://techdifferences.com/difference-between-active-and-passive-attacks.html.

9. Mann, Ian. *Hacking the Human, Social Engineering Techniques and Security Countermeasures.* Hampshire : Gower Publishing Limited, 2008. 978-0-566-08773-8.

10. Kevin, Mitnick. *CSEPS Course Workbook.* místo neznámé : Mitnick Security Publishing, 2004.

11. Watson, David, Holz, Thorsten a Mueller, Sven. Know your Enemy: Phishing. *The Honeynet Project.* [Online] 16. Srpen 2008. [Citace: 2. září 2018.] https://www.honeynet.org/papers/phishing.

12. What is Spear Phishing? *kasperskylab.* [Online] [Citace: 4. Září 2018.]

13. Vishing. *The free Dictionary.* [Online] [Citace: 4. Září 2018.] https://encyclopedia2.thefreedictionary.com/Vhishing.

14. Social Engineering, the USB Way. *Darkreading.* [Online] 6. Červenec 2006. [Citace: 5. Září 2018.] https://www.darkreading.com/attacks-breaches/social-engineering-the-usb-way/d/d-id/1128081.

15. Rouse, Margaret. malware (malicious software). *SearchSecurity.* [Online] Říjen 2016. [Citace: 8. Září 2018.] https://searchsecurity.techtarget.com/definition/malware.

16. Nash, Troy. An Undirected Attack Against Critical Infrastructure . [Online] Září 2005. [Citace:8.Září2018.]https://ics-cert.us-cert.gov/sites/default/files/recommended_practices/CaseStudy-002.pdf.

17. Judge, Kevin. What is a Computer Virus? *comodo antivirus.* [Online] 3. Srpen 2018. [Citace: 9. Září 2018.] https://antivirus.comodo.com/blog/computer-safety/what-is-virus-and-its-definition/.

18. Hák, Igor. Moderní počítačové viry. *Viry.* [Online] 2005. [Citace: 10. Září 2018.] https://www.fce.vutbr.cz/aiu/vojkuvka.m/u3v/vyuka/Kniha-o-virech.pdf.

19. Stallings, Wiliam a Brown, Lawrie. *Computer Security: Principles and Practice.* Boston : Pearson, 2017. 978-0134794105.

20. Ludwig, Mark. *The giant black book of computer viruses.* Scotts Valley : CreateSpace Independent Publishing Platform, 2009. 978-1441407122.

21. Rouse, Margaret. computer worm. *SearchSecurity.* [Online] Červen 2017. [Citace: 15. Září 2018.] https://searchsecurity.techtarget.com/definition/worm.

22. What is a computer worm, and how does it work? *Norton.* [Online] [Citace: 15. Září 2018.] https://us.norton.com/internetsecurity-malware-what-is-a-computer-worm.html.

23. Rouse, Margaret. Trojan horse (computing). *SearchSecurity.* [Online] Leden 2018. [Citace: 15. Září 2018.] https://searchsecurity.techtarget.com/definition/Trojan-horse.

24. Rouse, Margaret. ransomware. *SearchSecurity.* [Online] Září 2017. [Citace: 16. Září 2018.] https://searchsecurity.techtarget.com/definition/ransomware.

25. ransomware. *US-CERT.* [Online] 11. Červenec 2016. [Citace: 16. Září 2018.] https://www.us-cert.gov/security-publications/Ransomware.

26. *Cryptovirology: extortion-based security threats and countermeasures.* Young, Adam a Yung, Moti. Oakland : Proceedings 1996 IEEE Symposium on Security and Privacy, 1996. 0-8186-7417-2 .

27. What is Spyware? - Definition. *Kaspersky.* [Online] [Citace: 16. Září 2018.] https://www.kaspersky.com/resource-center/threats/spyware.

28. Erbschloe, Michael. *Trojan, Worms, and Spyware: A Computer Security Professional's Guide to Malicious Code.* Oxford : Butterworth-Heinemann, 2004. 978-0750678483.

29. Spyware. *TechTerms.* [Online] [Citace: 17. Září 2018.] https://techterms.com/definition/spyware.

30. Rouse, Margaret. adware. *SearchSecurity.* [Online] Duben 2017. [Citace: 17. Září 2018.] https://searchsecurity.techtarget.com/definition/adware.

31. What is AdWare. *Bank Valut cybersecurity.* [Online] [Citace: 16. Září 2018.] https://www.bankvaultonline.com/knowledge-base/definition-of-the-day/definition-adware/.

32. Aitel, Dave a Young, Susan. *The hacker´s handbook: The strategy behind breaking into and defending networks.* New York : Auerbach publications, 2004. 978-0849308888.

33. Microsoft's Software is Malware. *GNU Operating System.* [Online] 3. Květen 2017. [Citace:25. Únor 2019.] https://www.gnu.org/proprietary/malware-microsoft.en.html#back-doors.

34. Snyder, Bill. Snowden: The NSA planted backdoors in Cisco products . *InfoWorld.* [Online] Květen 2014. [Citace: 3. Březen 2019.] https://www.infoworld.com/article/2608141/snowden--the-nsa-planted-backdoors-in-cisco-products.html.

35. Graham, James, Olson, Ryan a Howard, Richard. *Cyber Security Essential.* London : Routledge, 2010. 978-1439851234.

36. Cole, Eric. *Hackers Beware.* místo neznámé : New Riders Publishing, 2001. 0-7357-1009-0.

37. Spitzner, Lance. *Honeypots: Tracking Hackers.* Boston : Addison Wesley, 2002. 978-0321108951.

38. Provos, Niels a Holz, Thorsten. *Virtual Honeypots: From Botnet Tracking to Intrusion Detection.* Boston : Addison-Wesley Professional, 2007. 0-321-33632-1.

39. Intrusion detection system (IDS). *Searchsecurity.* [Online] Leden 2018. [Citace: 13. Prosinec 2018.] https://searchsecurity.techtarget.com/definition/intrusion-detection-system.

40. Spitzner, Lance. The Value of Honeypots, Part One: Definitions and Values of Honeypots. [Online] 9. Říjen 2001. [Citace: 12. Prosinec 2018.]

https://www.symantec.com/connect/articles/value-honeypots-part-one-definitions-and-values-honeypots.

41. Project, Honeynet. Know Your Enemy: Honeynets. [Online] Honeynet Project, 31. Květen 2006. [Citace: 6. Prosinec 2018.] http://old.honeynet.org/papers/honeynet/.

42. Welcome to the Honeywall project site. [Online] The Honeynet Project. [Citace: 10. Říjen 2018.] https://projects.honeynet.org/honeywall/.

43. Roo CDROM User´s Manual - 6. Maintaining. *The Honeynet Project.* [Online] 25. Květen 2007. [Citace:13.Prosinec 2018.] http://old.honeynet.org/tools/cdrom/roo/manual/6-maintain.html.

44. Project, The Honeynet. Know Your Enemy: Sebek - A kernel based data capture tool. *The Honeynet Project.* [Online] The Honeynet Project, Listopad 2003. [Citace: 28. Prosinec 2018.] old.honeynet.org/papers/sebek.pdf.

45. Hussain, Sadequl. How To View and Configure Linux Logs on Ubuntu and Centos . *DigitalOcean.* [Online] 2013. [Citace: 20. Únor 2019.] https://www.digitalocean.com/community/tutorials/how-to-view-and-configure-linux-logs-on-ubuntu-and-centos.

46. VirtualBox. [Online] [Citace: 2. Říjen 2018.] https://www.virtualbox.org/.

47. What is virtualization? *opensource.com.* [Online] [Citace: 16. Prosinec 2018.] https://opensource.com/resources/virtualization.

48. Nemeth, Evi, Snyder, Garth a Hein R. Trent, Whaley, Ben. *Unix and linux system administration handbook.* New Jersy : Prentice Hall, 2010. 978-0131480056.

49. About The Honeynet Project. *The Honeynet Project.* [Online] [Citace: 22. Prosinec 2018.] https://www.honeynet.org/about.

50. Allegretta, Chris. The GNU nano homepage. *nano-editor.* [Online] [Citace: 22. Prosinec 2018.] https://www.nano-editor.org/who.php.

51. *snort.org.* [Online] [Citace: 23. Prosinec 2018.] https://www.snort.org/.

52. Watkins, Michael a Kevin, Wallce. *CCNA Security Official Exam Certification Guide.* Indianapolis : Cisco Press, 2008. 978-1587202209.

53. THC Hydra. *Sectools.* [Online] [Citace: 9. Únor 2019.] https://sectools.org/tool/hydra/.

54. Cyber definitoins. *ccdcoe.* [Online] [Citace: 18. Srpen 2018.] https://ccdcoe.org/cyber-definitions.html.

55. Nmap free security scanner. *nmap.org.* [Online] [Citace: 9. únor 2019.] https://nmap.org/.

7.2 Seznam obrázků

7.3 Seznam tabulek