Univerzita Komenského v Bratislave

Fakulta matematiky, fyziky a informatiky

Analýza rodín internetových robotov

bakalárska práca

2015

Jozef Brandys

Univerzita Komenského v Bratislave

Fakulta matematiky, fyziky a informatiky

Analýza rodín internetových robotov

bakalárska práca

Študijný program:	Informatika
Študijný odbor:	2508 Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	Mgr. Martin Jurčík

Bratislava, 2015

Jozef Brandys

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Jozef Brandys
Študijný program: informatika (Jednoodborové štúdium, bakalársky I. st., denná forma)
Študijný odbor: 9.2.1. informatika
Typ záverečnej práce: bakalárska
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Analýza rodín internetových robotov
Analysis of Internet Bot Families

Cieľ: Cieľom bakalárskej práce je analyzovať vývoj rodín internetových robotov (botov), analyzovať použité techniky a technológie. Práca bude poskytovať ucelený pohľad na danú problematiku z pohľadu informačnej bezpečnosti. Práca by sa mala hlavne zamerať na rodiny bankových trójskych koní, ich dopad na napadnutú infraštruktúru.

Vedúci: Mgr. Martin Jurčík
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: doc. RNDr. Daniel Olejár, PhD.

Dátum zadania: 28.10.2014

Dátum schválenia: 28.10.2014 doc. RNDr. Daniel Olejár, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Abstrakt

Botnety, ktoré sú siete kompromitovaných zariadení, sa stali veľkou hrozbou internetu. Práca je sondou narastajúcej hrozby, popisujúca aktuálne poznatky a trendy týkajúce sa botnetov. Venuje sa popísaniu botnetov pomocou ukážok ich životného cyklu, komunikačných sieti a motivácií útočníkov. Taktiež popisuje možnosti ich analýzy a detekcie, ktorých cieľom je botnet zneškodniť. Špeciálna pozornosť sa v práci dostáva bankovým botnetom.

Kľúčové slová: botnet, bot, bankové botnety, malwér, škodlivý kód

Abstract

Botnets, which are networks consisting of compromised devices, have become a serious threat to the Internet. Their growing threat has become theme of this bachelor thesis, which summarizes actual knowledge about botnets, including latest trends. It describes botnets by examples of life cycle, communication channels and motivation of attackers. It also describes the possibilities of analysis and detection of botnets, all of which target is to disrupt botnets. Special attention in this thesis is given to banking botnets.

Keywords: botnet, bot, banking botnets, malware, malicious code

Obsah

Úvod

Jedným zo závažných problémov dnešnej doby je aj šírenie botnetov a ich hrozba pre užívateľov. Botnety sú siete kompromitovaných zariadení, ktoré sa stali veľkou hrozbou internetu. Tieto siete boli vytvorené za účelom nelegálnych aktivít, dokonca ohrozujú fungovanie súkromných aj verejných služieb v mnohých krajinách na celom svete. Odhliadnuc od toho, že sa jedná o celosvetový problém, cieľom bolo sumarizovať aktuálne poznatky a trendy v tejto oblasti. Cieľom bolo zároveň podrobnejšie popísať bankové botnety a hrozby, ktoré spôsobujú finančnému sektoru. Výsledkom našej snahy je táto bakalárska práca, ktorú sme rozčlenili do 5 hlavných kapitol.

V prvých troch kapitolách sme sa venovali botnetom všeobecne. Prvá kapitola popisuje motiváciu ľudí, ktorí ovládajú botnety - botherderov, a formy útokov na ktoré sa botnety používajú. Ďalej sme popísali technológie, ktoré botnet využíva počas svojho životného cyklu a ako jeho životný cyklus vyzerá. Spomenuli sme tiež najdôležitejšiu časť botnetov - komunikačný model, ktorý botnety využívajú.

V druhej kapitole sme popísali možnosti analýzy botnetov, ktoré využívajú výskumníci. Najprv sme sa zamerali na spôsoby detekcie prítomnosti botnetu v systéme, alebo v počítačovej sieti. Neskôr sme popísali metódy na analýzu vzorky, ktorú sa nám podarilo získať. A nakoniec spôsoby analýzy celého botnetu, jeho hierarchie a enumeráciu jednotlivých botov. V rámci kapitoly sme tiež popísali technológie, ktorými sa botnet snaží zabrániť detekcii, tzv. obfuskačné metódy.

Témou tretej kapitoly sú metódy na zabránenie šírenia botnetov. Naj-

skôr sme si spomenuli ako predísť infekcii a zabrániť infik;ovaniu systému. V prípadoch, keď sa už botnet rozšíril, sú spomenuté aj možnosti jeho zneškodnenia. Kapitolu sme ukončili možnosťami prevencie útokov, ktorých úlohou je skomplikovať dosiahnutie cieľov botherderov.

V štvrtej kapitole sme sa zaoberali bankovými botnetmi. Začali sme bližším popísaním útokov, ktoré sa používajú a ďalej sme si popísali niektoré botnety, ktoré sa vyskytli na internete za posledné obdobie.

Piata kapitola je zhrnutím predpovedí na najbližšie roky, aké technológie budú botnety využívať a ako budú dosahovať svoje ciele.

Cieľom bakalárskej práce bolo analyzovať hrozbu botnetov a priniesť ucelený prehľad na problematiku. Taktiež sa mala zamerať na bankové botnety.

Kapitola 1

Botnety

Slovo botnet sa skladá z dvoch častí - bot a net. Bot je skratka od slova robot a net je od network(sieť). Botnety sú siete tvorené zariadeniami, ktoré sú infikované malwérom (škodlivý softvér), jednotlivé systémy sa nazývajú boti alebo zombie systémy. Ten kto má botnet pod kontrolou a dokáže ho ovládať sa nazýva botmaster alebo botherder. Hlavnou charakteristikou botnetou je možnosť vzdialenej komunikácie na ovládanie botov[56], ale niekedy túto vlastnosť zámerne nemajú.

V tejto kapitole budeme približovať botnet ako taký, najskôr uvedieme motiváciu, ktorá stojí za botnetmi a ich útokmi. Nadviažeme na to a vysvetlíme si ako sa niektoré typy útokov vykonávajú a následne si ukážeme typický životný cyklus bota a jeho priebeh. Nakoniec si popíšeme rôzne modely komunikačných sietí, ktoré botnet používajú.

1.1 Motivácia botherdera

Motivácia botnetu závisí hlavne na cieľoch jeho botherdera. Prvotnou motiváciou bolo ukázanie zručností programátora, ktorý sa chcel pochváliť svojím botnetom a poukázať na zraniteľnosti. Čoskoro sa, ale motivácia zmenila a botnety sa využívajú pre osobné či politické účely.

Bežným cieľom je získavanie financií. Profit pomocou botnetu sa dá získať

mnohými spôsobmi. Prvým je získavanie osobných údajov z napadnutých systémov a ich následný predaj na čiernom trhu. Najbežnejšou komoditou sú v tomto prípade prihlasovacie údaje do internet bankingov, čísla kreditných kariet. Ďalším je využívanie systému na posielanie spamu, ktorý predávajú tretím stranám, ktorý ho môžu využívať na phishing čím ukradnú údaje a získajú z nich peniaze [85].

Špeciálne sa na získavanie financií pomocou botnetov zameriavajú bankové botnety, ktoré budú neskôr témou samostatnej kapitoly, ich cieľom sú bankové inštitúcie a špeciálne ich internetové bankovníctvo.

Útočník, ktorý získava peniaze za navštívenie webovej alebo za kliknutie reklamy, môže využiť napadnutý počítač na vytváranie klamlivých navštívení stránky [14, 34], alebo ju môže priamo zobrazovať používateľovi napr. modifikáciou webových stránok, inštalovaním reklamných rozšírení do prehliadačov.

Existuje aj mnoho ďalších finančných motívov na použitie botnetov. Napríklad sledovanie hráčových kariet pri hraní pokru [36] alebo sa botnety tiež využívali na ťaženie crypto meny Bitcoin [60, 70].

Botnety sa tiež stali súčasťou mnohých elektronického útokov napríklad elektronického špehovania. Cieľom môžu byť nielen komerčné spoločnosti, ale aj národné organizácie či vojenská infraštruktúra. Narozdiel od bankových botnetov sa používajú špecializované botnety s obmedzeným množstvom cieľov a starostlivo si vyberajú počítače na ktoré sa idú rozširovať aby zbytočne nevzbudzovali pozornosť. Dôvody na útoky bývajú často aj politického charakteru. Objavili sa aj prípady, kde sa predpokladá že za botnetmi stoja vládne agentúry [85].

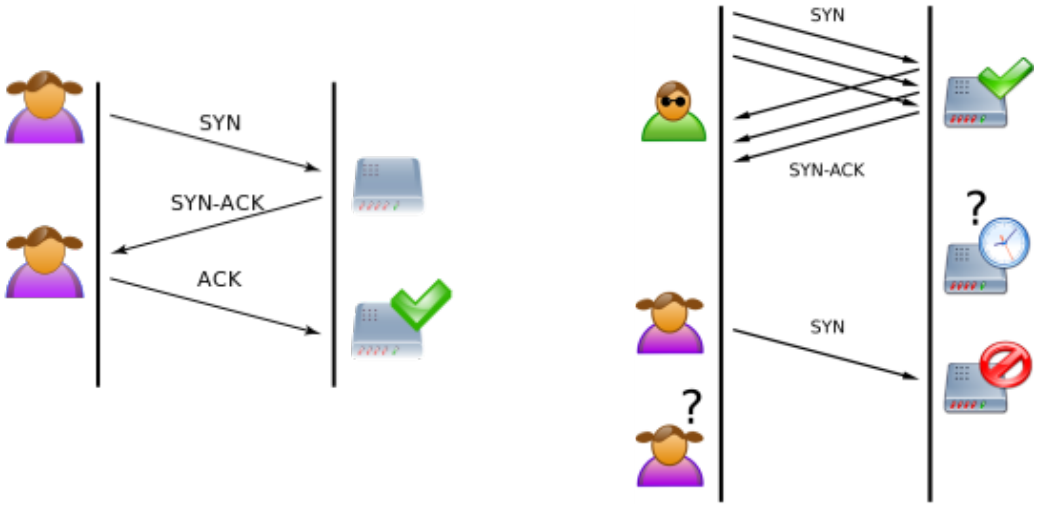
Jednotlivci alebo rôzne skupiny využívajú na istý druh protestu, čo je istý druh haktivizmu. Haktivizmus je využívanie cyber útokov, kvôli politickému alebo sociálnemu dôvodu. Veľmi známy je prípad útoku na národné organizácie v Estónsku z roku 2007, čo bola odpoveď na premiestnenie sochy sovietskeho vojaka v Talinne . Pri hacktivizme sa najčastejšie používajú botnety na DDoS útok na web stránky cieľa [88].

1.2 Formy útokov

Formy útokov, ktoré budeme popisovať v nasledujúcej časti úzko súvisia s už popísanou motiváciou botherderov. Pre každý cieľ existuje niekoľko útokov, ktoré pomáhajú útočníkovy dosiahnuť svoj cieľ[85, 60, 54].

- **DDoS útok** - je pokus o dočasné alebo trvalé prerušenie služby pripojenej k internetu[89]. Narozdiel od DOS útoku, kde útočí jeden systém, pri DDoS útoku sa zneužíva dva a viac systémov. Získal si veľmi veľkú popularitu a v roku 2014 bolo priemerne zistených až 28 útokov za hodinu [17].

  Jednou z možných metód, ako realizovať útok, zahrňuje vyťaženie cieleného systému komunikačnými požiadavkami tak, aby nedokázal odpovedať na komunikáciu s reálnymi používateľmi služby. Docieľuje sa to napríklad núteným reštartom systému, vyťažením komunikačných liniek alebo iných zdrojov. V minulosti sa zväčša využívala na útoky 4. vrstva OSI modelu (transportná služba)[86]. Najznámejší je TCP SYN flood útok, kedy sa využíva 3 paketové nadväzovanie TCP spojenia, pri ktorom sa pošle iba prvý paket a nečaká sa na odpoveď. Server čaká na ďalšie pakety a kým vyprší časový limit spojenia, zaberá systémové zdroje. Ukážku môžeme vidieť na obrázku 1.1 Potom prišli útoky na 7. vrstve (aplikačnej). Príkladom môže byť Http Post DDos útok, pri ktorom útočník pošle http požiadavku na server a v hlavičke protokolu nastaví "Content-length" (dĺžka obsahu) na vhodne zvolené väčšie číslo, následne posiela obsah tela http požiadavky pomalou rýchlosťou, aby na čo najdlhší čas využíval prostriedky serveru a udržal spojenie. Na aplikačnej vrstve môžu existovať aj špecializované útoky, využívajúce špecifické informácie o konkrétnych bežiacich službách, napríklad chyby typu buffer overflow, výpočtovo náročné požiadavky a mnohé ďalšie. Výhodou distribuovanému útoku oproti nedistribuovanému nie je iba väčšie množstvo zdrojov, ktorými môže zaťažiť server, ale aj ťažšie blokovanie väčšieho množstva systémov a anonymizovanie pôvodcu



Obr. 1.1: Ukážka najprv normálneho spojenia a potom TCP SYN flood útoku(Zdroj: wikipedia.com)

útoku, prípadne odpútanie pozornosti od iného paralelného útoku.

- **SPAM a Phishing** - sieť botov sa používa na posielanie nevyžiadanej pošty (spamu), pričom podobne ako pri spomínaných DDoS útokoch, útočník skrýva svoju identitu a využívanie väčšieho množstva botov zabraňuje jednoduchému blokovaniu správ. Vlastník pošle botom predlohu správy a k nej niekoľko pravidiel, ako sa správa môže modifikovať a prispôsobovať konkrétnemu prijímateľovi tak, aby bolo pre filtre čo najťažšie identifikovať či sa skutočne jedná o spam, alebo nie [53].

  Spam je jedným z možných druhov šírenia botnetu a infikovaním ďalších systémov. Využitie spamu zvyčajne závisí na sociálnom inžinierstve a neinformovanosti používateľov o nástrahách, ktoré na nich čelia pri používaní internetu [69, 68].

- **Krádež osobných údajov** - malwér máva často prístup k systému ako bežná aplikácia, čo mu umožňuje takmer bez obmedzení zbierať množstvo údajov. Vlastník botnetu môže zaslaním príkazu stiahnuť súbory z infikovaného počítača, sledovať jeho komunikáciu alebo aj zaznamená-

vať stlačenia kláves (keylogging). Môže napríklad kradnúť prihlasovacie mená k internet banking-u, plateným účtom alebo čísla kreditných kariet, ktoré sa neskôr pokúsi predať na čiernom trhu. Bežnou súčasťou botou je extrahovanie prihlasovacích údajov z manažérov hesiel, ftp klientov, e-mailových klientov, či aj krádež súkromných kľúčou, ktoré sa používajú na šifrovanie komunikácie.

Častým spôsobom pri kradnutí údajov je používanie tzv. man in the browser útoku, keď bot dokáže modifikovať zobrazované stránky, čítať z nich údaje aby dosiahol svoj cieľ. Veľmi často sa využívajú napríklad finačnými botnetmi. Útočníci takisto implementujú do botov aj možnosť pre vzdialené pripojenie (backconnect) a na pozadí napríklad pomocou VNC, ovládajú počitač cez grafické rozhranie.

- **Ostatné** - Botnety sa snažia využiť svoju prítomnosť aj na infikáciu počítača ďalšími škodlivými kódmi. Na rôznych skrytých fórach sa dá kúpiť služba pay-per-install, ktorou môžete spustiť vami zvolený kód na už napadnutých systémoch. [11].

1.3 Životný cyklus

Životný cyklus, ktorý si spomenieme, sa skladá z niekoľkých častí. Začína sa konfiguráciou botov, ktorý sa budú ďalej šíriť. Druhou je infikovanie počítača, čo je prvotný úspech bota. Neskôr, aby sa stal súčasťou botnetu, musí byť schopný s ním komunikovať, čím sa dostane do tretej fázy. Po svojej inicializácií a pripojení sa k botnetu nastáva hlavná fáza, keď bot vykonáva všetku svoju činnosť a komunikuje s botherderom. Aj keď to nie je úplne bežná súčasť cyklu, spomenieme si aj poslednú fázu, keď z rôznych dôvodov ukončí svoju činnosť.

Pred samotnou existenciou konkrétneho botnetu si spomenieme ešte zopár činností, ktoré sa dejú ešte pred tým ako sa botnet začne rozširovať. Prvá činnosť je implementácia samotného botnetu. Programátori, ktorý vytvoria potrebné kódy na botnet (C&C server, bot, nástroj na infikovanie) buď pre-

vádzkujú botnet sami alebo ho môžu predávať a získavať tak finančné odmeny od botherderov.

1.3.1 Konfigurácia bota

Keď má niekto prístup k kódom botnetu ešte pred tým, ako ho uvoľní na internet, nastaví konfiguráciu. Súčasťou konfigurácie môžu byť parametre rôzneho druhu, napríklad informácie o C&C servery, šifrovacie kľúče, dĺžka intervalu medzi jednotlivými pripojeniami na C&C server, dĺžka čakania pred prvým pripojením, rôzne skripty, zoznam údajov, ktoré nám ma poslať, stránky, ktoré má sledovať, moduly ktoré ma používať,.... Keď je vytvorená konfigurácia pripraví sa bot. Mnohé botnety majú ako svoju súčasť generátor botov, ktorý dostanú ako vstup konfiguračný súbor, moduly a výstupom je binárny kód bota vhodný na šírenie.

1.3.2 Infikovanie počítača

Útočníci sú pri infikovaní veľakrát veľmi vynaliezaví a využívajú mnohé druhy techník. Škodlivý softvér, ktorí následné pripojí počítač to vybranej siete kompromitovaných zariadení, teda ho zapojí do botnetu sa môže do počítača dostať so spoluprácou používateľa tzv. sociálnym inžinierstvom, pomocou neopravených zraniteľností, zlej konfigurácií firewallu alebo iného softwéru, slabých hesiel, alebo využitím inej slabiny. Teraz si popíšeme niektoré z najčastejšie využívaných spôsobov infikovania a inštalácie bota do počítača. Krátkymi ukážkami naznačíme, ako sa v minulosti využívali v reálnych prípadoch [28].

- **Sociálne inžinierstvo**

  Ide o metódu manipulácie používateľov za účelom prevedenie akcie, ktorej cieľom je získať dôverné informácie alebo prístup do informačného systému [90]. Útočník môže rozposielať správu, ktorá sa javí ako dôveryhodná, a nabáda tak používateľa, aby otvoril škodlivý súbor,

navštívil webovú stránku, nainštaloval si nový softvér. Problém nastáva vtedy, keď sa používateľ nechá zmanipulovať a otvorí napríklad infikovaný súbor (pdf-dokument, textový dokument, prezentáciu, ...) či odkaz, ktorý využíva zraniteľnosť v prehliadačoch, alebo to je v skutočnosti iba binárny kód s ikonou charakeristickou pre daný typ súboru [69]. Pri infikovaní pomocou webových stránok sa využívajú zraniteľnosti v internetových prehliadačoch, zásuvných moduloch a podobne. Škodlivý kód sa môže objaviť aj na zdanlivo bezpečných stránkach pomocou Cross-site scripting (XSS) útoku, kedy útočník spustí svoj kód pomocou chýb pri tvorbe webu.

Útočník môže nalákať užívateľa ja na inštaláciu trójskeho koňa. Pri tomto útoku si používateľ chce nainštalovať nový nástroj, ktorý môže síce naozaj robiť to, čo od neho očakával, ale navyše sa spustí aj škodlivý kód ktorý v sebe obsahuje bota. Mnohé útoky sociálneho inžinierstva útoky majú výhodu, že obchádzajú filtre a dokážu sa dostať aj na stanice, ktoré sa môžu sa skrývať za firewallmi alebo NAT routerom.

- **Prípojné zariadenia**

  Infiltrovať zariadenie sa dá aj pomocou rôznych dátových nosičov, USB gadgetov a ďalších prídavných zariadení, ktoré si používateľ pripojí do počítača. Spomínali sme infiltráciu pomocou USB, ale výskumníkom sa podarilo identifikovať aj iný malwér, ktorý sa šíril pomocou USB kľúčov napr. Conficker a Stuxnet. Prvý menovaný využíval na svoje šírenie automatické spúšťanie pri vložení usb a dalo sa brániť vypnutím tejto funkcie. Stuxnet využíval 0-day exploit (chyba, ktorá ešte nebola verejne známa) na svoje šírenie a dokázal sa šíriť aj bez funkčného internetového pripojenia[79].

  Útok môže prebiehať napríklad cez automatické infikovanie vložených USB, ktoré ďalej šíria hrozbu. Iný spôsob je za pomoci sociálneho inžinierstva, keď útočník trikom zmanipuluje svoju obeť, napríklad aj v obchode pomocou USB nabíjačky na elektronické cigarety, ktoré sa lacno predávajú na internete [83], alebo aj “náhodným” USB kľúčom,

ktoré nájdete na parkovisku [80].

- **Využívania chýb v sieťových službách**

  Na sieti sa bežne nachádzajú rôzne služby. Napríklad DNS servery poskytujú možnosť zistiť IP adresu k doménovej adrese, rôzne služby na zdieľanie súborov, webové servery, proxy servery a mnohé ďalšie. Niektoré z týchto služieb bývajú štandardne nainštalované v systéme a spúšťajú sa bez ďalšej konfigurácie. Chyby v takýchto programoch môžu dovoliť útočníkovi spustiť svoj vlastný kód, získať citlivé dáta [46, 62]. Služby, ktoré bývajú automaticky spúšťané sú o to nebezpečnejšie, že postihujú veľké množstvo systémov, a preto bývajú zvyčajne útočníkmi preferované. Problém spôsobujú taktiež slabo zabezpečené služby, keď používajú nezmenené heslá.

- **Drive-by-Download útok**

  Automatické sťahovanie škodlivého kódu do systému (napríklad pri navštívení stránky, otvorení e-mailu) bez vedomia používateľa a mnohokrát aj bez žiadneho náznaku a interakcie s používateľom [56]. Narozdiel od využívania chýb v sieťových službách, kde sa pri útoku posielajú dáta smerom k systému od útočníka, používateľ sám iniciuje pripojenie a stiahne si kód k sebe. Preto je pre firewally komplikované zabrániť takémuto útoku. Rozlišujeme 2 spôsoby útoku:

  – Zneužitie API - Ak jedno API ponúka možnosť stiahnuť si súbor a druhé spustiť ľubovoľný súbor, ich kombinácia vedie k možnosti drive-by-download útoku. Rozšírené používanie rôznych pluginov dáva útočníkovi veľa možných kombinácií na realizovanie možného útorku.

  – Využitie chýb v prehliadači - podobne ako pri útokoch cez sieť, útočník sa pokúsi využiť chybu v softvéry (prehliadač alebo rôzne jeho rozšírenia) a spustiť svoj vlastný kód, k úspečnému využitiu zraniteľností mnohokrát pomáhajú skripty bežiace na používateľovom systéme, napr. javascript

Aby vôbec mohol útok začať, musí používateľ najprv navštíviť škodlivú stránku. Útočník môže využiť vyššie popísané metódy sociálneho inžinierstva, pričom posiela spam s linkami na škodlivé stránky alebo môže využiť pharming útok, keď sa snaží presmerovať normálnu stránku na svoju škodlivú (napríklad modifikáciou host súboru, napadnutím DNS serveru). Napríklad ransomware Cryptowall sa šíril pomocou spam a RIG exploit kit-u.[22] Na maximalizovanie úspešnosti napadnutia používateľa sa útočník snaží zvýšiť množstvo stránok, na ktorých je prítomný škodlivý kód. Toto môže dosiahnuť napadnutím web servera a infikovaním stránky. Ďalšou možnosťou ako zvýšiť šance na úspech, je umelé zvyšovanie hodnotenia vo vyhľadávačoch či možnosť nepriamo infikovať stránku pomocou vkladania reklám z externých zdrojov, škodlivými rozšíreniami [28]. Príkladom môže byť vloženie škodlivého na stránky svetových denníkov cez reklamnú agentúru [51].

- **Slabé heslá**

  Jednoduchým spôsobom útoku je uhádnutie hesla a získanie prístupu k systému. Botnet Conficker používal na infiltráciu slovníkový útok[32]. Problémom sú takisto routre, ktoré mnoho používateľov neprekonfiguruje a zostávajú zraniteľné na skúšanie defaultných hesiel. Príkladom môže byť botnet Lizard, ktorý infikuje takmer výlučne domáce routre, ale našli sa infikované aj komerčné routre napríklad na univerzitách a vo firmách [33].

- **Pay-per-install**

  Nakoniec si spomenieme odlišný prístup. Pri tomto druhu infikovaní sa využije iný malwér, ktorý už predtým infikoval systém. Mnoho malwéru dokáže spúšťať ľubovoľné programy a túto službu ponúkajú na predaj. Útočník si jednoducho zaplatí za inštaláciu malwéru. Hlavnou výhodou je flexibilita, keď tvorca botnetu sa môže sústrediť na funkcionalitu a infekciu systému za neho vyrieši niekto iný.

1.3.3 Inicializácia bota

Potom ako bol spustený škodlivý kód botu musí zabezpečiť komunikačný kanál s botnet sieťou a vykonať príkazy na zamedzenie jeho detekcii v systéme. Bot sa snaží identifikovať antivírusové programy a zabrániť im zobrazeniu varovných hlášok o možnej infiltrácií. Tiež sa snaží skryť svoju prítomnosť v systéme tým, že vymaže súbory a záznamy, aby predišiel detekcii z vniknutia do systému. Taktiež sa snaží skryť pomocou rôznych techník, ktoré neskôr popíšeme v kapitole 2.3.

V inicializačnej fáze bot zistí informácie o práve napadnutom systéme a vyhodnotí jeho možnosti, na základe toho mu udelí ďalšie úlohy poprípade sa ho nepokúsi infikovať. Môže tiež detekovať či sa nenachádza v simulovanom prostredí, ktoré sa snažia využiť výskumníci na jeho pozorovanie a v kladnom prípade sa môže rozhodnúť ich oklamať, napríklad generovaním požiadaviek na pripojenie sa na falošné C&C servery a tím zmiasť stopy o skutočnej infraštruktúre.

1.3.4 Vykonávanie činnosti

Najdôležitejšia fáza života botu je, keď začne vykonávať úlohy, pre ktoré bol určený. V tejto fáze bot počúva na komunikačnom kanály a vykonáva všetky príkazy, ktoré mu botherder zašle. Najpoužívanejšie úlohy sme už popísali v kapitole 1.2 a na ich vykonávanie, si môže bot stiahnuť z internetu rôzne moduly, ktoré sú potrebné na vykonanie inštrukcie. Takto sa môže bot aktualizovať, vykonávať nové druhy útokov, alebo aj využívať nové zraniteľnosti na ďalšiu propagáciu. V tejto fáze takisto posiela na C&C servery výsledky príkazov a botherder môže získavať prehľad o svojom botnete.

1.3.5 Ukončenie činnosti

Ukončenie činnosti je väčšinou na príkaz botherdera, keď existuje podozrenie odhalenia bota alebo keď už botherder nemá žiadne ďalšie úmysly s infikovaným počítačom. Niektoré boty v rámci tejto fázy za sebou zametajú

stopy vymazaním všetkých svojich súčasti, nastavením registrov na pôvodné hodnoty, vymazanie systémových záznamov a podobne.

Životný cyklus, ktorý sme práve popísali iba zovšeobecňuje správanie botnetov, no skutočné správanie sa väčšinou aspoň mierne líši, napríklad môžu zahrňovať fázu, kedy čakajú na svoju aktiváciu, aby nepriťahovali na seba pozornosť, nemusia mať implementovanú fázu na ukončenie činnosti alebo môžu mať vlastné špeciálne fázy.

1.4 Komunikačný model botnet sieti

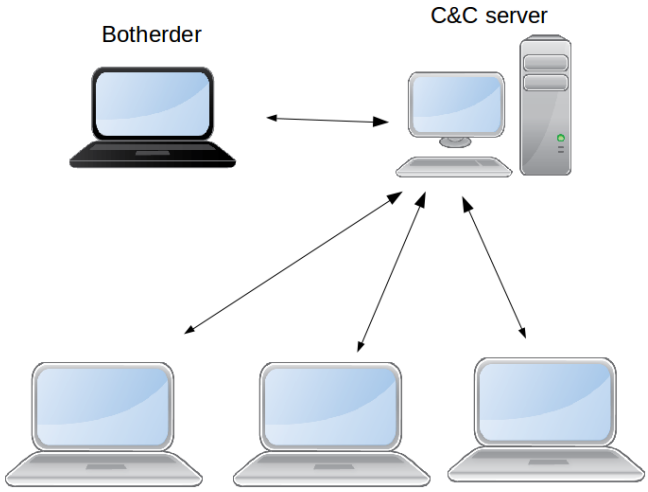
Pri infikovaní počítačov musí útočník zabezpečiť tvorbu komunikačnej siete, aby ju neskôr mohol použiť na zasielanie príkazov. Kontrolný mechanizmus botnetov sa nazýva aj command and control infraštruktúra (C&C infrastructure).

Dobrá infraštruktúra musí vyriešiť mnoho problémov a v tejto sekcii sa pokúsime porovnať rôzne , a takisto zistíme, aké majú výhody a nevýhody. Za mnoho rokov vývoja sa zaznamenalo niekoľko rôznych modelov a útočníci sa snažili vytvoriť spoľahlivejšie modeli, ktoré by čo najviac sťažili kompromitovanie siete a detekciu jednotlivých botov.

1.4.1 Centralizovaný C&C model

Centralizovaný model je najpoužívanejší model vôbec, práve vďaka najjednoduchšej implementácií. V centralizovanom modeli sa boty pripájajú na jediný C&C server, ku ktorému má útočník prístup, aby mohol prostredníctvom neho zadávať príkazy (ukážku vidíme na obr. 1.2). Najväčším nedostatkom je, že obsahuje iba jediný bod, ktorý môže byť jednoducho detekovaný a tím by znefunkčnil celú sieť, ako sa to už v mnohých prípadoch stalo. Tento problém sa môže vyriešiť niekoľkými spôsobmi:

- **Fast flux**



Obr. 1.2: Centralizovaný C&C model

Na pripojenie sa využije sieť DNS serverov. Pri pripájaní boty získavajú DNS záznamy, ktoré majú zvyčajne krátku dobu platnosti a obsahujú záznamy o niekoľkých stokách (či tisícoch) IP adries. Vďaka krátkej dobe a rýchlej zmeny záznamov sa budú IP adresy rýchlo meniť. Tieto adresy zvyčajne patria už napadnutým počítačom, ktoré sa správajú ako proxy servery na ceste k C&C serveru. [45]

- **Domain generation algorithm (DGA)**

  Jednotlivé boty generujú množinu DNS adries, na ktorých sa môže ukrývať C&C server, väčšinou sa jedná o generátory, ktoré ako počiatočný stav používajú aktuálny dátum. Vzhľadom na to, že útočník pozná konkrétny algoritmus na generovanie domén, môže si ich dopredu rezervovať a nastavovať DNS záznamy tak, aby zabránil ich kompromitovaniu. Medzi hlavné výhody patrí zvýšená odolnosť voči firewallom, udržiavanie C&C infraštruktúry je o niečo jednoduchšie, keď môže byť aktívna iba v určité časové úseky (napríklad keď chce botherder zadať nový príkaz) a registrovanie vygenerovanej adresy tesne pred jej aktivitou sťažuje používanie rôznych metód na blokovanie prístupu [20]. Ako príklad môžeme uviesť botnet Conficker, ktorý zvykol generovať

približne 50000 domén denne, z ktorých si náhodne vybral 500 a tie sa pokúšal kontaktovať. Ešte podotkneme, že posledné dve metódy sa niekedy zvyknú kombinovať, ako to bolo aj v prípade Gameover Zeus [55].

- **Používanie anonymizačných sietí**

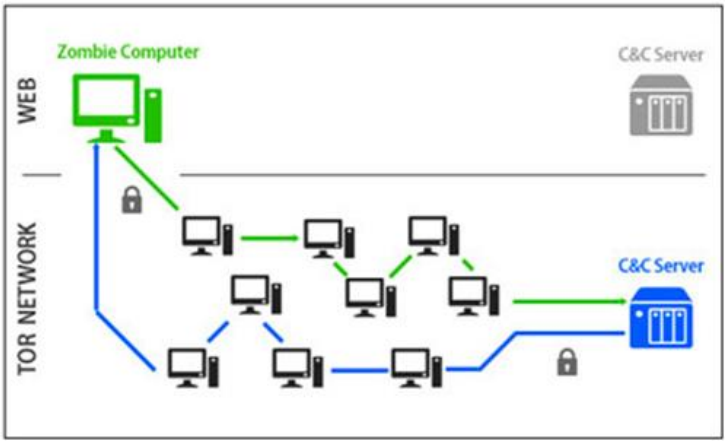
  Komunikácia s C&C serverom prebieha cez siete napríklad Tor alebo I2P (The invisible project), ktoré presmerúvavajú a šifrujú komunikáciu (príklad môžeme vidieť na obrázku 1.3), tak aby sa nedal vystopovať skutočný cieľ správ a ani ich obsah. Výhodami sú:

  – Komunikácia vyzerá ako klasická komunikácia využívanej anonymizačnej siete, a preto je ťažšie ju filtrovať a detekovať v nej správy patriace botnetu.
  – Ťažká lokalizácia C&C servera.
  – Využívanie kryptografie vylučuje možnosť prevzatia kontroly na adresou
  – Botherder môže presúvať svoj fyzický server jednoduchým znova použitím tých istých šifrovacích kľúčov.

  Medzi nevýhody patrí hlavne zložitosť komunikácie, ktorá spôsobuje problémy pri implementácií botnetu alebo veľká dĺžka odozvy pri komunikácií, či väčšie využívanie systémových prostriedkov pri používaní kryptografie. Príkladom môžu byť 64-bitová verzia botnetu Zeus, ktorá využívala sieť Tor a bankový trojan Dyre, ktorý využíva spomínanú sieť I2P [71].

- **Sociálne siete ako Cloud C&C server**

  Botherderi si zaregistrujú na online sociálnych sieťach svoj vlastný účet a neskôr postujú nové statusy, príspevky, komentáre, v ktorých sa nachádzajú zašifrované príkazy. Pri pripojení nového bota do siete, sa bot jednoducho pripojí a stiahne si aktuálne príkazy, ktoré následne rozšifruje. Podobne botherderi využívali cloudové služby, pomocou ktorých šírili príkazy. Ich výhodou je odolnosť voči rôznym útokom. Ak by



Obr. 1.3: Komunikácia cez sieť Tor (Zdroj: infosecinstitute.com)

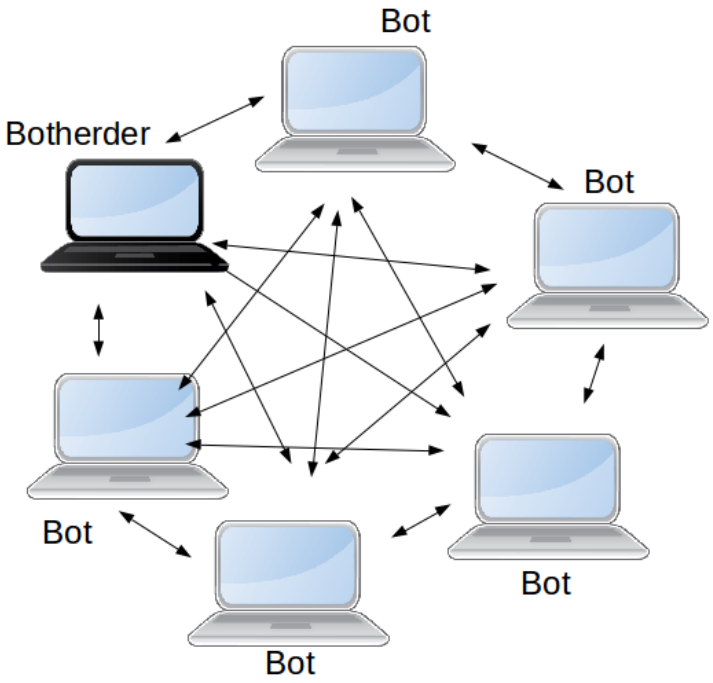
bol účet zablokovaný, tak sa jednoducho vytvorí nový (zoznam môže byť generovaný alebo aj prítomný v botovi). Taktiež sa veľmi ťažko blokuje prístup, keďže sociálne siete nemôžu byť zablokované firewallmi, kvôli využívaniu skutočnými používateľmi. Boli zaznamenané pokusy, kedy sa používali falošné účty na Facebooku, Twitteri(ukážku môžeme vidieť na obrázku 1.4) či služba Google Groups, Dropbox alebo Reddit. [3, 47, 50]

Obr. 1.4: Príkaz od C&C serveru na Twitteri (Zdroj: arbornetworks.com)

Aby sa predišlo aj dočasným výpadkom siete, väčšinou sa používa viacero C&C serverov. Táto metóda zahrňuje aj tzv. load balancing, keď sa potrebný

výkon rozdelí medzi viacero serverov. Mierne zložitejšou verziou centralizovaného C&C modelu je hierarchický model, kde jednotlivé C&C servery tvoria stromovú štruktúru, na vrchole ktorej je botherder.

1.4.2 Peer to peer C&C model

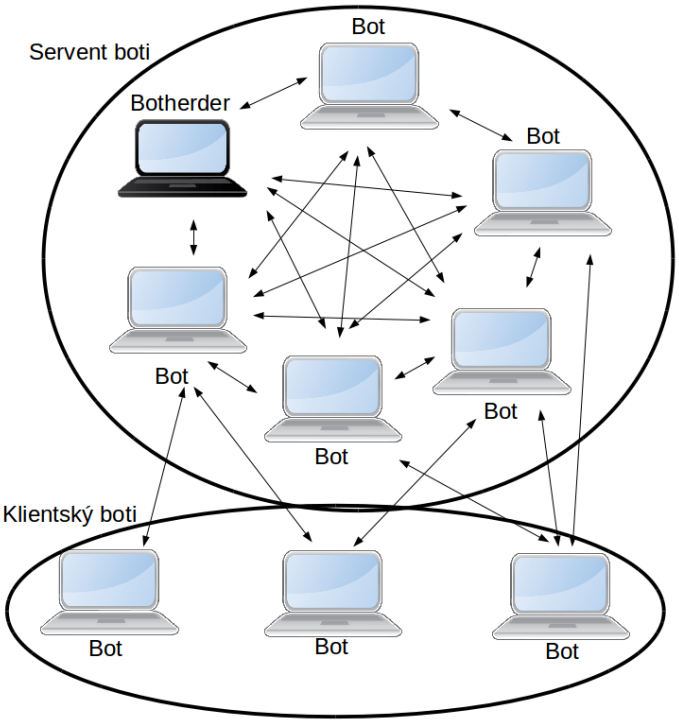


Obr. 1.5: Peer to peer C&C model

Hlavnou výhodou peer to peer systému je jeho decentralizovanosť, čo sa nakoniec ukáže aj ako jeho nevýhoda. V tomto modeli každý uzol (peer) vystupuje aj ako server, aj ako klient. Jednotlivé uzly sú napadnuté počítače. Dokonca aj samotný botherder sa pripája ako obyčajný uzol, pričom v sieti rozposiela podpísané správy (môžeme vidieť na obr 1.5). Keďže je decentralizovaný, netrpí podobnými problémami ako centralizovaný model a navyše ani kompromitovanie jednej stanice neznamená zánik celej siete. Na

druhej strane jeho implementácia je omnoho zložitejšia, čo spomaľuje jeho rozšírenie medzi botnetmi, takisto môže byť niekedy latencia správ pomerne vysoká a nie je ani zaručené doručenie správy. Ďalšou nevýhodou je ich rozšíriteľnosť a možné rozdrobenie do menších častí, keď jednotlivé uzly vytvoria nezávislé skupiny, ktoré medzi sebou nedokážu komunikovať. Na kompromitovanie takejto siete sa často využíva fakt, že žiaden uzol nemá informácie o celej sieti. Vtedy útočník pripojí mnoho falošných uzlov do siete tak, aby tvorili jej značnú časť a neskôr ich všetky, odpojí pričom sa snaží nechať sieť čo najviac fragmentovanú.

1.4.3 Hybridný C&C model



Obr. 1.6: Hybridný C&C model

Písali sme o niekoľkých modeloch, z ktorých každý má svoje výhody a

nevýhody, ktoré sú v značnom kontraste. Preto sa prirodzenou cestou vývoja vytvoril hybridný model (ukážka na obr. 1.6), ktorý spája čo najviac výhod z predchádzajúcich modelov. Tím Ping Wanga navrhol model, v ktorom by sa sa skupiny botov rozdelili na dve časti [87]. Prvá obsahuje tzv. servent boty, ktoré zohrávajú podobnú úlohu ako C&C servery v centralizovanom modeli, ale taktiež aj úlohu klienta. Tieto boty mávajú statickú, nesúkromnú IP adresu, ktorá je prístupná z internetu. Druhá skupina botov sa označuje ako klientske boty, ktoré majú napr. dynamickú IP adresu, IP adresu nedostupnú z internetu, pretože sa nachádza za firewallmi, alebo je ich IP adresa súkromná. Každý bot má zoznam servent botov (peer list), na ktoré sa môže pripájať a keďže majú statické IP adresy, takáto sieť je stabilná. Aby nedošlo k odhaleniu celej siete pri kompromitovaní jedného botu, každý bot obsahuje iba obmedzený počet záznamov .

Je zrejme, že hybridný model je ekvivalentný C&C modelu, kde servent boty reprezentujú C&C servery, pričom sú navzájom prepájané, čo je zase typická vlastnosť Peer to peer modelu. Takáto sieť je odolnejšie proti snahám o jej zneškodnenie a dokonca je ťažšia aj jej detekcia, keď jej komunikácia je typicky šifrovaná. Zároveň je ale jednoduchšia na správu pre vlastníka botnetu vďaka stabilnejším spojeniam.

1.4.4 Náhodný C&C model

Náhodný model je iba teoretickým modelom, ktorý sa ešte nezaznamenal pri žiadnom známom botnete. Jeho výhodou je jednoduchosť, keďže žiadny bot nepozná viac ako jeden ďalší bot v sieti. Botherder pri posielaní nového príkazu najprv príkaz zašifruje, a potom sa snaží v sieti nájsť živé boty, ktorým by mohol predať správu. Jeho hlavná výhoda je, že odhalenie jedného botu nekompromituje celú sieť. Nájsť všetkých botov v sieti, ale väčšinou trvá veľmi dlho a dokonca ani nie je zaručené, že správa sa doručí, napríklad ak je bot pri skúmaní jeho adresy nedostupný.

Kapitola 2

Analýza

V tejto kapitole si popíšeme jednotlivé techniky, aké môže používať výskumník, keď chce analyzovať botnet. Pred samotnou analýzou musí, ale nasledovať detekcia, ktorá bude popísaná v prvej časti. Nasledovať bude opis analýzi vzorky bota. Proti analýze botnety aktívne bojujú a vymýšľajú rôzne techniky ako jej zabrániť. Nazývajú sa obfuskačné techniky a budú témou ďalšej časti.

Sledovanie celého botnetu je zaujímavá časť, ktorá nám umožní vidieť aj do pozadia fungovania botnetu a preto jej venujeme predposlednú časť tejto kapitoly. Poslednou témou tejto kapitoly bude počítanie veľkosti botnetov.

2.1 Detekcia

Predtým než začneme čokoľvek analyzovať, potrebujeme nájsť spôsob, ako to nájsť. K tomu nám slúžia techniky, ktoré si popíšeme v nasledujúcej kapitole. Detekcia malwéru je náročný proces, či už kvôli minimalizovaniu falošných detekcií, ale aj výzve objaviť nové druhy malwéru, ktoré využívajú najnovšie poznatky o ich detekcii a snažia sa vytvoriť metódy na jej obídenie. Detekcia malwéru a vyhýbanie sa detekcii znamenajú neustály súboj tvorcami malwéru a tvorcami detekčných riešení.

Boty na systémoch zanechávajú mnoho stôp, či už pri infekcii, hľadaní zra-

niteľností. Na ich nájdenie môžeme sledovať médium (väčšinou sieť), ktorým sa snažia šíriť. Po úspešnej infekcii sa snažia robiť škodlivú činnosť, ktorá je tiež potencionálnym zdrojom informácií na detekciu. Ďalšou spoločnou charakteristikou je komunikácia s C&C serverom a sledovaním komunikačných kanálov môžeme hľadať špecifické vlastnosti takejto komunikácie.

Znaky, ktoré nám pomáhajú odhaliť infekciu sa nazývajú indikátory kompromitácie (IOC - Indicator of compromise). Najčastejším zdrojom tíchto znakov je analýza botnetu. Medzi indikátory kompromitácie patria napríklad odlačky súborov, IP adresy, používané mutexy, DNS adresy a mnohé ďalšie.

2.1.1 Analýza záznamov

Pri každej infekcii bot musí pristupovať k systémovým zdrojom, ktoré číta alebo mení. Napríklad keď sa snaží zabezpečiť, aby sa spustil aj po reštarte, zanechá stopy, napríklad vo forme zmenených/vytvorených súborov, či registrov. Moderné počítače si udržiavajú informácie o činnosti programov v logovacích súboroch, ktoré môžu byť zdrojom na analýzu. Vo všeobecnosti je ťažké oddeliť normálnu a škodlivú činnosť, a preto sa vytvorili špecializované nástroje nazývané tiež host-based intrusion detection systems, ktoré monitorujú logy na hľadanie stôp malwéru. Výhodou je, že touto metódou dokážeme detekovať aj neznámy malwér, ale kedže sledujeme stopy až po úspešnej infekcii, môže sa pokúsiť zahladiť za sebou stopy a zabrániť tak detekcii.

Okrem bežných logov v systéme, môžeme sledovať aj logy iných aplikácií. Napríklad firewall nám môže dať vedieť o pokusoch na pripojenie na blokované porty, keď sa botnet pokúsi pripojiť na C&C server alebo pri pokusoch o infekcii ďalších aplikácií cez sieť [85].

2.1.2 Antivírus

Cieľom antivírusových programove je identifikovať malwér, pričom sa väčšinou zameriava na konkrétne aplikácie, ktoré jednotlivo skúma. Spomenieme si niekoľko druhov techník používaných modernými antivírusmi[95]:

- **Hľadanie odtlačkov**

  Pri hľadaní odtlačkov sa zameriava na statické ohodnotenie skúmanej vzorky. Zvyčajne sa porovnáva kryptografický odtlačok celého súboru alebo iba niektorých jeho sekcií. Napriek tomu, že táto technológia bola súčasťou antivírusových riešení od začiatku, postupne sa vytráca jej účinnosť. Prvým dôvodom je používanie obfuskačných metód (kap. 2.3) a druhým je, že porovnávaním odtlačkov dokážeme identifikovať iba tie, ktoré už máme v databáze, zatiaľ čo neznámy malwér nie.

- **Používanie heuristík** je vylepšenie predchádzajúcej metódy. V binárnom kóde sa hľadajú podozrivé inštrukcie, kombinácie funkcií bez hľadania konkrétneho vzoru, s ktorým by sme to porovnávali. Antivírus môže tiež emulovať program a sledovať jeho správanie a stav pamäti. Problémom je, že identifikovaním podozrivých vlastností by sme mohli ľahko označiť dobrý program za zlý, preto sa využíva ohodnotenie na základe nájdených vlastností a ak prekročí prahovú hranicu, označí sa za malwér, čím sa zníži počet falošných identifikácií. Moderné antivírusy zvyknú analyzovať vzorku aj v cloude, čo má výhodu zníženia nároku na systém a zároveň kombináciou výsledkov z ostatných klientov antivírusu sa môžu dosiahnuť lepšie výsledky. Problémom analýzi v cloude sú citlivé informácie, kedže sa posielajú mimo siete užívateľa.

- **Pozorovanie správania** pri vykonávaní programu a hľadaní podozrivého správania, napríklad rozbaľovanie škodlivého kódu, zmena súborov, zaznamenávanie stlačených kláves, ... Pozorovaním môžeme identifikovať aj doteraz neznámy program. Podobne ako pri používaní heuristík, jednotlivé činnosti nemusia vzbudiť veľké podozrenie, ale ich kombinácia áno. Táto technika z časti zasahuje do techniky Analýza

záznamov.

2.1.3 Analýza sieťového toku

Bot počas svojho životného cyklu nezanecháva stopy iba na systéme, ale časom sa pripája na C&C server, či infikuje ďalšie systémy cez sieť. Analýza môže prebiehať buď analýzou jednotlivých packetov, keď môžeme pomocou rôznych znakov identifikovať podozrivé packety a nahlásiť nájdenie škodlivej činnosti. Problém nastáva, ak by bot posielal jednotlivé správy rozkúskované v malých packetov, keď by sa hľadaný znak neidentifikoval, lebo by sme ho nevyskladali. Druhým spôsobom je prirodzene využívať analýzu sieťových tokov, ich rekonštrukciou dostaneme celú komunikáciu medzi C&C serverom a botom. Tu nastáva problém, ak sa komunikácia šifruje, nedokážeme ju analyzovať. Podobne ako pri predchádzajúcich prípadoch porovnávame packety a ich vlastnosti iba z vopred vypočítaných znalostí o škodlivej komunikácií, takže dokážeme identifikovať iba už známe infekcie.

Pri analýze môžeme hľadať aj znaky, ktoré sa vymykajú normálnemu správaniu. Prvou fázou takejto analýzy je sledovanie normálneho toku, keď sme si istí že sieť nie je napadnutá. Následným pozorovaním siete, môžeme porovnať jej stavy s tými pri normálnej prevádzke a hľadať v nej anomálie, ktoré by dokazovali existenciu bota v sieti. Táto metóda môže síce odhaliť aj neznáme druhy malwéru, má ale aj pomerne veľa falošných hlásení, napríklad pri inštalácií nového softvéru, zmenu serverov, vnútornej infraštruktúry a podobne.

Analýza paketov aj sieťového toku si vyžaduje nemalé systémové prostriedky a nasadenie týchto technológii do väčších sieti by bolo finančne náročné. Môžeme sa teda pri analýze zaoberať iba vlastnosťami paketov a nie ich obsahom. Pri tzv. netflow analýze sa využívajú informácie ako zdrojová a cieľová IP adresa, čísla portov, protokolu, veľkosť paketu. Takto dokážeme spracovať omnoho väčšie množstvo paketov. Často sa ešte používa vzorkovanie, keď sa vyberie iba časť týchto dát, ktorá sa skutočne pošle na analýzu. Napriek tomu, že sme stratili množstvo informácie, čím sme stratili možnosť

všetko detekovať, ostalo jej dostatok na to, aby sme v nej identifikovali stopy škodlivej činnosti (skenovanie portov, časté získavanie informácií z DNS serverov, veľa odmietnutých spojení, distribúcia spamu, ...) [85, 21].

2.1.4 Honeypoty

Honeypoty sú zariadenia, ktoré poskytujú rôzne služby, ktoré môžu byť potenciálne napadnuté útočníkom. Tieto zariadenia ich, ale neponúkajú pre bežných používateľov, ale sú rôznymi spôsobmi podvrhované útočníkom. Vzhľadom na to že, jediný účel je byť napadnutý, každá komunikácia môže byť považovaná za škodlivú, a tak majú oproti ostatným metódam nižšie množstvo falošných hlasení. Zároveň sú monitorované a pri útoku môže byť získaných mnoho informácií.

Keďže sa pri detekcií útokov nehľadajú žiadne konkrétne črty, ako to bolo pri predchádzajúcich metódach, sú úspešné aj pri hľadaní nových druhov malwéru. Pri úspešnom napadnutí môžeme pomocou honeypotov získať informácie o spôsobe napadnutia, pozorovať kód a jeho ďalšiu interakciu so systémom alebo sieťou a keďže pri napadnutí sa stiahne aj malwér získame tím aj jeho vzorku.

Takisto sa môžu jednotlivé honeypoty zapojiť do siete nazývanej honeynet a môžeme pozorovať aktívne šírenie malwéru v našom kontrolovanom prostredí [40]. Honeypoty obsahujúce plne fungujúci operačný systém, ktorý komunikuje s útočníkom sa nazývajú vysoko interaktívne, tie sa ďalej delia na klientské a serverové.

Serverové väčšinou pasívne čakajú na útok, čím vzniká nízke množstvo falošných alarmov, keď väčšina komunikácie je škodlivá, čo je ich hlavnou výhodou.

Klientské, narozdiel od serverových, aktívne vyhľadávajú hrozby. Simulovaním klientských aplikácií, napríklad webových prehliadačov, pripájanie sa na rôzne servery, zdieľanie súborov v sieti a pod.. Môžu navštevovať potenciónálne nebezpečné webové dokumenty, otvárať neznáme dokumenty, spúšťať a inštalovať nebezpečné aplikácie a mnoho ďalších vecí. Tieto aktivity zvy-

šujú šancu napadnutia počítača, avšak vytvárajú množstvo udalostí, ktoré treba odfiltrovať od tých tvorených škodlivým kódom. Zároveň nám monitorovanie honeypotov poskytuje detailné informácie o útoku, prístup k vzorke, adresy serverov, s ktorými sa komunikuje, informácie o vykonaných zmenách v systéme a potencionálne aj ďalšie informácie o správaní sa malwéru. Po napadnutí a ukončení pozorovania je potrebné honeypot resetovať do pôvodného, aby sme pri ďalšom útoku dostali informácie výlučne pre tento útok.

Pri používaní reálnych kópií operačných systémov, môžeme testovať iba jednu konkrétnu statickú konfiguráciu, verziu softwéru. Preto sa vytvorili takzvané nízko interaktívne honeypoty, ktoré iba simulujú rôzne služby s rôznymi simulovanými zraniteľnosťami. Vzhľadom na to, že sú iba simulované, útočník nezíska prístup k počítaču, ale informácie môžu byť použité na extrakciu škodlivého kódu a nemusíme honeypot resetovať na jeho vyčistenie z predchádzajúceho útoku. Týmto spôsobom dostaneme iba obmedzený pohľad na hrozbu, ktorá napadla honeypot. Výhodou je naopak voľnosť pri simulovaní, keď môžeme rýchlo meniť správanie a prispôsobiť správanie prichádzajúcej požiadavke vybraním správnej verzie, aby sme získali čo najviac možných informácií.

2.1.5 Spamovacia pasca

Ako sme už vyššie písali, jeden z najrozšírenejších spôsobov využitia botnetu je posielanie spamu. Botnety v týchto prípadoch zvyčajne dostanú tepmplate správy, pravidlá na jeho modifikáciu, aby sa predišlo automatickej detekcii a taktiež zoznam e-mailových adries, ktoré sa zvyčajne získavajú z ľubovoľných webových stránok. Takzvané web cravleri alebo aj zberače aktívne prehľadávajú internet, aby z neho extrahovali e-mailové adresy. Tento proces môže byť napadnutý, keď sú mu podvrhované e-mailové adresy zriadené na tento účel, ktoré sú bežnému používateľovi neviditeľné, pretože bývajú prekrývané iným textom, vypísané farbou pozadia alebo schované iným spôsobom vo webovom dokumente. Tieto e-mailové adresy sa nazývajú spamovacie pasce (angl. Spam traps). Spamovacími pascami sa môžu stať aj kedysi regulárne

e-mailové adresy už zrušených účtov. Adresy môžu byť tiež generované dynamicky a pre každého náštevníka sa vytvorí jednoznačná adresa, táto informácia môže pomôcť pri identifikácií útočníka a odhalenie adresy počítača, z ktorého prebiehal zber údajov[59, 1].

Prichádzajúci spam môže pomôcť pri detekcii už existujúceho malwéru viacerými spôsobmi. Prvým je už spomínané vystopovanie adresy web crawlera. Ďalším je získanie adresy už napadnutého počítača, z ktorého bol poslaný konkrétny mail. Využíva sa na to fakt, že každý server, ktorý prijíma alebo preposiela e-mail, pridáva do hlavičky e-mailu IP adresu systému, z ktorého mu prišla správa.Takto sa môže podariť vystopovať odosielateľ správy, ktorý je s vysokou pravdepodobnosťou už napadnutý malwérom. Okrem tohoto, ďalším spôsobom, ako môžu spamovacie pasce pomôcť pri detekcii malwéru je získanie vzorky, ktorá sa môže nachádzať v e-mailovej správe prebiehajúcej kampane na šírenie malwéru.

Ako môžeme vidieť spamovacie pasce nám môžu pomôcť pri detekcií už známych, ale aj nových botnetov a dokonca sa dostať k ich vzorke.

2.2 Analýza botnetu

Potom ako sa podarí detekovať malwér, ďalším krokom je jeho analýza, aby sme zistili čo najviac informácií o jeho činnosti. Tieto informácie sú dôležité pre odhadnutie veľkosti hrozby, ktorú môžu jednotlivý boti napáchať, získanie rôznych špecifických znakov, ktoré môžu byť použité na detekciu prítomnosti malwéru v iných systémoch alebo aj zistenie úrovne rafinovanosti útočníka, ktorý ho navrhol. Takisto sa môže na základe analýzy navrhnúť postup na odstránenie malwéru z systému, zabrániť jeho ďalšiemu šíreniu a získať mnoho ďalších užitočných informácií. Pod pojmom analýza rozumieme proces skúšania jednotlivých častí malwéru, jeho správania aby sme ho študovali. Otázky, na ktorých odpovede sa snažíme pri analýze prísť, môžeme zatriediť do dvoch kategórií. Prvou sú otázky všeobecného charakteru zaoberajúce sa hrozbou a jej dopadmi na sieť, korporácie, užívateľov. Druhou

kategóriou sú otázky o konkrétnych technológiách používané botnetom [52]. Medzi najbežnejšie otázky z prvej kategórie patria:

1. Čo je cieľom malwéru ?
2. Kto za ním stojí a aký silný je útočník ?
3. Ako sa dá odstrániť ?
4. Čo sa mu podarilo ukradnúť, napáchať ?
5. Ako dlho sa nachádza v systéme ?
6. Dokáže sa nezávisle rozširovať ?
7. Ako ho detekovať na iných systémoch ?
8. Ako zabrániť opakovanému infikovaniu systému ?

Odpovede na tieto otázky sa zvyčajne dajú získať správnou kombináciu a syntézou odpovedí na otázky technického charakteru, ako sú napríklad:

1. Ako malwér infikoval systém ?
2. Aké sú sieťové indikátori, ktoré odhalia prítomnosť malwéru.
3. Ako identifikovať prítomnosť v systéme ?
4. Je infikovanie trvalé ? Aké mechanizmy pri tom používa ?
5. Kedy bol malwér napísaný, skompilovaný a nainštalovaný ?
6. Je závislý na inom dobre známom nástroji ?
7. V akom jazyku bol napísaný ?
8. Je malwér spackovaný ? Aký nástroj sa použil ?
9. Používa anti-debugovacie techniky ?
10. Nachádza sa v ňom kód na získanie správcovských oprávnení ?

Pri analýze máme najčastejšie prístup iba k binárnej podobe, ktorú môžeme rekonštruovať a zisťovať jej funkcie. Tento proces sa nazýva reverzné inžinierstvo. Táto technika sa delí na dynamickú, ktorá ho skúma pri jeho spustení a statická, ktorá analýzu obsahu vzorky bez toho aby ju spúšťala. Obe techniky majú svoje výhody a nevýhody a navzájom sa dopĺňajú. V nasledujúcich častiach sa vám ich pokúsime priblížiť.

2.2.1 Dynamická analýza

Dynamická analýza sa používa na pozorovanie skúmanej vzorky malwéru za jeho behu. Spúšťaním vzorky v kontrolovanom a pozorovateľnom prostredí sa dajú získať mnohé cenné informácie. Ako prvé môžeme pri analýze pozorovať rôzne zmeny v systéme, môžu to byť, napríklad zmenené systémove registre, vytváranie alebo zmena súborov, vytvorenie nových procesov. Toto môžeme dosiahnuť buď sledovaním všetkých zmien, alebo porovnaním pôvodného stavu a nášho pozorovaného. Sledovanie zmien väčšinou prebieha zaznamenávaním systémových volaní. Informácie o zmenách v systéme nám môžu pomôcť zistiť, ako si malwér zabezpečuje spustenie aj po reštarte počítača, alebo po vypnutí niektorého z jeho procesov. Takisto môžu pomôcť pri detekcii a vytváraní postupov na jeho kompletné odstránenie zo systému.

Ďalší prostriedok ktorí môžeme monitorovať je sieťová komunikácia. Môžeme sledovať, ako sa snaží rozširovať do ďalších systémov, akú aktivitu vykonáva v sieti a keďže sa pravdepodobne časom bude snažiť pripojiť k C&C serveru, dokážeme pozorovaním získať detailné informácie o C&C infraštruktúre (napr. či sa pripája na pevnú IP adresu, alebo ju zisťuje pomocou iných prostriedkov napr. DNS, či využíva sieť Tor alebo iné). Ak komunikácia nie je šifrovaná, môže byť aj tá podrobená ďalšej analýze. Taktiež môžeme sledovať posielanie spamu, aké služby využíva na infekciu ďalších systémov, aké príkazy dostáva od vlastníka botnetu a či sa zúčastňuje prebiehajúceho DDoS útoku a mnohé iné.

Dynamická analýza môže prebiehať viacerými spôsobmi. Jedným z nich je používanie debuggera a druhým je spúšťanie programu v kontrolovanom prostredí tzv. sandboxe.

- **Debugger** je program, ktorý dovoľuje spúšťať program krok po kroku, aby človek ktorý ho analyzuje pochopil ako funguje. Debuggeri môžu dovoliť meniť hodnoty v pamäti, registroch či meniť zdrojový kód, aby sa kód ľahšie analyzoval, a tiež na obídenie anti-debugovacieho kódu. Využitím debuggera dokážeme získať mnohé cenné informácie o malwéru, ale si to vyžaduje množstvo času a úsilia. Kód zvyčajne obsahuje

množstvo anti-debugovacích častí, ktoré majú skomplikovať jeho analýzu. Môže obsahovať detekciu prítomnosti debuggera, či už zavolaním systémových funkcií, meraním času, alebo využívaním iných odlišnosti prostredia debuggera od reálneho systému. Takisto sa kód z kompilátora môže nahradiť jeho významovo ekvivalentným, ale zato omnoho zložitejším, či dlhším kódom. Zvyčajne takýto kód obsahuje čo najviac podmienených skokov, aby výsledná logika bola čo najkomplikovanejšia.

- **Sandbox** je prostredie, ktoré má iba obmedzený prístup k dôležitým prostriedkom - napríklad systémových súborom, sieti, hardwérovým zariadeniam a podobne. Všetok prístup k týmto prostriedkom býva monitorovaní a niekedy aj emulovaný.

  Namiesto emulovania iba určitých prostriedkov môžeme simulovať priamo celé prostredie pomocou virtuálnych zariadení, ktoré obsahujú nástroje na analýzu. Aj v tomto prípade podobne ako v prechádzajúcich sa prostredie líši od skutočného prostredia, čo môže malwér identifikovať a zmeniť svoje správanie. Ďalším jednoduchým prostriedkom na vyhnutie sa analýze je, napríklad čakanie istú dobu predtým, než začne niečo ďalšie vykonávať, či vykonávanie škodlivej činnosti iba v určitých časových intervaloch. Keďže pri dynamickej analýze sú prostriedky monitorované, emulované alebo inak modifikované, malwér sa vykonáva pomalšie ako zvyčajne a presným meraním času môže zistiť, či je spustený v kontrolovanom prostredí. Niektoré prostredia preto umelo spomaľujú ubehnutý čas, v tom prípade ale môže napríklad nastať timeout pri sieťovej komunikácií. Takisto, ak ho špecialista upraví, môže si pravidelne kontrolovať svoj kontrolný súčet a detekovať zmenu. Existujú avšak nástroje, ktoré dokážu vytvárať tzv. shadow kópiu pamäte, ktorú poskytnú programu, keď sa pokúsi o čítanie a výpočet kontrolné súčtu.

  Všetky tieto techniky, nám ale poskytnú obraz iba o určitom časovom úseku, v ktorom bol program analyzovaný, aké príkazy malwér

prijal či prostriedkoch ku ktorým mal prístup (napr. prítomnosť súboru, konkrétny systém a pod.). Preto nám dynamická analýza pomôže odhaliť iba určitú časť funkcionality [25, 27].

Posledným problémom dynamickej analýzy je, že spúšťaním kódu sa môžeme podieľať na škodlivej činnosti, ktorá porušuje zákon.

2.2.2 Statická analýza

Statická analýza na rozdiel od dynamickej analýzy nespúšťa malwér. Zvyčajne sa skúma iba binárna reprezentácia kódu. Jedným z prvých krokov môže byť kontrola malwéru antivírusovými programami, pri tom nám môžu pomôcť online služby, ako napríklad službu `https://www.virustotal.com/`, ktorá ho skontroluje rôznymi programami naraz [28]. Mnoho detailov o malwéri sa dá získať skúmaním rôznych konštánt, ktoré sa nachádzajú v binárnej podobe zakódované vo vzorke. Môžu to byť napríklad IP adresy, dómenové mená C&C serverov, čísla portov, na ktorých malwér počúva, cesty k rôznym súborom, mená meniacich registrov, heslá k rôznym službám, kľúče využívané pri šifrovaní. Taktiež môže pomôcť analýza využívaných API funkcií systému alebo rôznych knižníc. Mnoho dát sa nachádza aj v hlavičke súboru, môže sa tam nachádzať čas kompilácie, ikona programu, číslo verzie, ďalšie reťazce a importované či exportované funkcie.

Pri statickej analýze binárneho kódu, sa kód prevedie do podoby čitateľnejšej pre analyzátora, zvyčajne to bude strojový kód. Takýto kód neobsahuje ako klasické programovacie jazyky mnemotechnické pomôcky pri názvoch funkcií a premenných. Kód sa zvyčajne delí na dielce, ktorého základnou jednotkou je skupina inštrukcií, ktoré sa spustia za sebou bez toho, aby sa mohol zmeniť beh programu. Analyzovaním spojení medzi týmito blokmi, môžeme získať predstavu o chode programu. Ich analyzovaním môžeme napríklad zistiť, ako prebieha komunikácia medzi C&C serverom, aké zmeny sa vykonávajú v systému, aké informácie hľadá, či aké kryptografické konštrukcie používa.

Podobne ako pri dynamickej analýze, aj pri statickej sa útočník pokúša znemožniť alebo čo najviac sťažiť prácu analyzátora. Využívajú sa na to tzv. obfuscation (zatemňovacie) techniky. Prvou môže byť použitie packera, ktorý zašifruje, skomprimuje alebo inak modifikuje program, aby skryl kód či obsah pamäte. Môžu sa tiež využívať virtuálne stroje, ktoré spúšťaj kód, ktorý je ťažšie analyzovateľným. Nato aby útočník zmiatol analyzátora, kód sa nahradí jeho sémantický ekvivalentným kódom. Táto technika sa podobá na jednosmernosť hashovacích funkcií, keď je jednoduché vytvoriť iný ekvivalentný kód, ale je ťažké nájsť jeho pôvodný obraz. Jednoduchým trikom je napríklad využitie faktu, že inštrukcie na niektorých architektúrach sú premenlivej dĺžky, ak sa vložia do kódu zbytočné časti, môžu zmiasť analyzátora, ktoré správne neurčia začiatok inštrukcie a väčšina kódu sa bude zdať ako náhodná zmes inštrukcií. Podrobnejšie sme si tieto metódy na zabránenie analýzy spomenuli v kapitole 2.3.

Ako posledný spôsob zabráneniu statickej analýze spomenieme modulovateľnosť malwérov, keď kód nemôže byť analyzovaný jednoducho, pretože nie je priamou súčasťou, ale sa stiahne z externého dátového priestoru podľa potreby. Poprípade môže byť síce prítomný v šifrovanej podobe, ale kľúč na odšifrovanie býva súčasťou príkazu na použitie modulu.

Autori malwéru zvyknú postupne vyvíjať novšie verzie, pričom znova a znova využívajú rovnaký kód a niekedy taktiež využívajú knižnice, nástroje ktoré im uľahčujú prácu. Dôkladná statická analýza býva aj s dobrými nástrojmi časovo náročná, a preto je vhodné, podobne ako to robia autori malwéru, využiť znalosti z predchádzajúcich analýz a tým si ušetriť námahu. Súčasťou statickej analýzy môže byť hľadanie spoločných častí kódu či už medzi rôznymi vzorkami malwéru, alebo medzi skúmaným malwérom a dostupnými opensource projektami. Príkladom môže byť RE-clone framework [16], ktorý sa pokúša identifikovať ich a poskytnúť o tom informáciu. Ukážkou môže byť analýza botnetu Citadel, keď sa tímu Rahamian [31] podarilo nájsť spojenie medzi viacerými open-source knižnicami a botnetom Zeus.

V spomínanej analýze môžeme tiež vidieť použitie nástroja RE-source

[30], ktorý pomocou analýzy kódu priradí funkciám tagy (označenia), napríklad: antidebugovacie techniky, injektovanie kódu do procesu, injektovanie DLL knižníc, výmena procesu, zmena registrov atď. Tieto tagy môžu byť použité na hľadanie zaujímavých častí, na porovnávanie rôznych verzií rovnakého malwéru. Dôležitým krokom pri použití posledných dvoch nástrojov, je predpríprava analyzovaného kódu. Dôvodom sú už spomínané obfuskačné metódy najmä packery a zašifrovanie, ktoré znemožňujú správne fungovanie nástrojom.

2.3 Obfuskácia

Obfuskácia je metóda, ktorej cieľom je vytvoriť nezrozumiteľný kód, ktorý je veľmi ťažké analyzovať, či detekovať. Proces obfuskácie kódu je transformácia, ktorá mení podobu kódu, ale zachováva jeho vnútorné vlastnosti [6].

V tejto časti si najprv spomenieme, niekoľko druhov obfuskovaných programov a aj najbežnejšie metódy na obfuskáciu.

2.3.1 Typy obfuskovaného malwéru

V tejto kapitole si spomenieme niekoľko základných typov obfuskovaného malwéru [94, 6, 73, 15, 75].

- **Šifrovaný malwér** (Encrypted malware)

  Prvý spôsob obfuskovania programu pomocou šifrovania na obídenie detekcie pomocou odtlačku. Malwér je najskôr zašifrovaný, pri spustení sa najprv spustí rutina na odšifrovanie malwéru, ktorá po skončení predá riadenie programu už dešifrovanému malwéru. Hlavným problémom je, že rutina na šifrovanie ostáva stále rovnaká, a tak aj napriek meneniu šifrovacích kľúčov, čím sa mení zašifrovaná časť malwéru, môže byť identifikovaný pomocou identifikácie kódu na šifrovanie.

- **Oligomorfný malwér** (Oligomorphic malware)

Podobne ako pri prvom prípade sa používa šifrovanie, ale rozdiel je v tom že autori šifrovaného malwéru pridali technológie na zmenu šifrovacích rutín, a tak sa mohli meniť medzi rôznymi verziami. Problém bol v malom počte rôznych mutácií, a tak sa stále dali ľahko identifikovať.

- **Polymorfný malwér** (Polimorphic malware)

  Ďalšia verzia, ktorá sa snažila vyriešiť problém s malým množstvom mutácií. Narozdiel od oligomorfného malwéru dokáže malwér pomocou techník popísaných v ďalšej sekcii vytvoriť nespočítateľné množstvo rôzneho kódu na šifrovanie, ktoré vždy inak vyzerá, ale správa sa rovnako. Výsledkom je malwér, ktorý nikdy nie je rovnaký a kód neobsahuje žiadne konštatné časti, pomocou, ktorých by mohol byť dekekovaný. Ale predsa je tu problém a ten nastáva hneď ako sa v pamäti objaví nešifrovaný kód, ktorý už nie je chránený proti detekcii. Antivírusové programy, spúšťajú malwér v chránenom prostredí (sandboxe) a úspešne prekonávajú problémy triky polymorfného malwéru.

- **Metamorfný malwér** (Metamorphic malware)

  Autory malwéru vyriešili aj problém s konštantným telom malwéru. Metaforný malwér modifikuje nielen dešifrovaciu rutiny, ale aj telo programu. Takto sa po dešifrovaní využíva vždy iná verzia, ktorá sa mení iba vnútorne, ale jej funkcionalita sa nemení. Súčasťou býva aj kód, ktorý dokáže automaticky vytvárať nové verzie, ktoré sa ďalej šíria. Takto modifikovaný malwér, je ťažko identifikovateľný a detekovateľný antivírusovými systémami.

- **Virtuálne stroje** (Virtual machines)

  Jedným z problémov je tiež skrývanie funkcionality, ktorú poskytuje kód. Kedže kód musí byť pred spustením nahraný do pamäte, nastáva moment kedy je analyzovateľný a môže nám dať isté informácie o malwéry napriek použitým obfuskáciám. Emulovaním inštrukcií, môžeme lepšie skryť kód pred analyzátorom.

  Pri používaní virtuálnych strojov je kód prepísaný do špeciálnej sady inštrukcií. Aj keď je analýza takéhoto kódu možná, trvá príliš dlho

a aj keď sa analyzátorovi podarí pochopiť fungovanie kódu, malwér už môže byť o niekoľko generácií ďalej a používať inú sadu inštrukcií na inom virtuálnom procesore.

2.3.2 Techniky

Na ukážku si spomenieme niekoľko jednoduchých metód na obfuskáciu kódu. Niektoré sa používajú už pri návrhu programu, iné sa pridávajú do programu používaním špeciálnych toolov [15, 94, 6].

- **Zámena inštrukcií** vymieňa inštrukcie za iné inštrukcie s rovnakým výsledkom.
- **Permutácia rutín** náhodne vymieňa časti kódu, ktoré sú od seba nezávislé.
- **Zmena registrov/premenných** v kóde je metóda, ktorá mení vymieňa pozície jednotlivých premenných v kóde, či mení registre, ktoré sa používajú.
- **Vkladanie zbytočného kódu**, je najjednoduchšie pomocou inštrukcie NOP, ktorá nemení správanie kódu, iba jeho vzhľad. Zložitejšie sa dá napríklad vkladaním zbytočných logických častí, ktoré sa nemusia vykonať (nazýva sa aj dead code) alebo aj keď sa vykonajú nemajú vplyv na výslednú logiku a správanie malwéru.
- **Kríženie kódu** mení vzhľad kódu vkladaním skokových inštrukcií, čím sa kód delí na menšie časti, ktoré sa môžu navzájom permutovať.
- **Integrovanie kódu** do inej aplikácie, tak aby sa spustil aj škodlivý kód.
- **Opaque constant** je metóda uloženia konkrétnej hodnoty bez toho aby sa dokázala získať jednoduchou analýzou. Takéto hodnoty majú mnohé využitia, pri skrývaní hodnôt premenných, adries v pamäti alebo vetvenia programu[29].

- **Skrývanie hodnôt premenných** môže prebiehať či už pomocou šifrovania (pomocou xoru, base64, ale aj zložitejších algoritmov), rozdelenia hodnoty do viacerých miest v pamäti, alebo pomocou spomínaných opaque konštánt.
- **Inlinovanie funkcie** môže pomôcť odstrániť stopy po logickom usporiadaní programu.
- **Duplikovanie funkcií** môže vytvoriť ďalšiu entropiu v programe a takisto pri analýze zmiasť analyzátora.

2.4 Sledovanie činnosti botnetu

Neskôr, keď už bol botnet detekovaný a prípadne aj zanalyzovaný, môže nás začať zaujímať celková štruktúra botnetu, pozadie fungovania botnetu či záujmy vlastníka botnetu. Informácie, ktoré sme získali pri reverznom inžinierstve kombináciou dynamickej a statickej analýzy, pomôžu sledovať celkovú činnosť botnetu. Sledovaním komunikácia C&C infraštruktúry môžeme sledovať počet botov, zistiť aké spamové správy šíri, kto je cieľom útoku (všetci používatelia, iba niekoľkých štátov, či osoby na vysoko postavených postoch, konkrétna firma a pod.). Môžeme takisto zaznamenať informácie o pripravovaných/prebiehajúcich útokoch a využiť na zabránenie jeho vykonaniu. Ďalším dôležitým dôvodom prečo je dôležité sledovať botnet, je možnosť vývoja botnetu, zmena na nové verzie, zmena šifrovacích kľúčov, C&C servera a podobne. Dôvodom je mať informácie o aktuálnych verziách, a tak aby sme mohli udržiavať krok a pripraviť vhodné protiopatrenia. Vedľajším efektom pri sledovaní botnetu môže byť aj detekcia iných botnetov a ďalšieho malwéru. Stáva sa to hlavne pri sledovaní botnetov ponúkajúcich službu pay-per-install, získame tak vzorku vhodnú na ďalšiu analýzu.

Jednou možnou cestou ako sledovať botnet, je pozorovanie vlastného bota. Na začiatok si infikujeme vlastnú fyzickú stanici. Nebudeme mať možnosť monitorovať vzorku podrobne s využitím všetkých nástrojov, ale zabránime tým možnej detekcii vlastného prostredia a prispôsobeniu správania bota,

tak aby neodhalil škodlivú činnosť. Musíme si ale dávať pozor, že spustením bota mu dovolíme nielen komunikovať s C&C serverom, ale aj vykonávať škodlivú činnosť a tým sa zúčastňovať trestného činu. Ak je komunikácia nešifrovaná, tak máme mnoho informácií takmer zadarmo. Problém nastáva ak je šifrovaná vtedy, musíme vytvoriť nástroj na odšifrovanie komunikácia čo si vyžaduje nemalú námahu. Taktiež môžu byť názvy príkazov nie príliš nápomocné, k tomu nám dopomôže analýza, aby sme zistili, ktoré príkazy čo vykonávajú. Veľkým problémom tejto metódy je jej škálovateľnosť, rôzne boty by sa mohli navzájom ovplyvňovať a ak by sme chceli pozorovať niekoľko botov naraz, pre každého by sme potrebovali vlastný fyzický systém čo je veľmi nákladné a zároveň virtualizované prostredie, ako sme už spomínali môže byť detekované. Avšak aj napriek nevýhodám je táto metóda vhodná na rýchly pohľad na činnosť botnetu.

Náročnejšie na prípravu je sledovanie botnetu pomocou špecializovaných nástrojov. Emulovaním komunikačnej časti botnetu, napodobňovaním správania reálneho bota sa môžeme pripojiť na C&C server. Takto získame všetky informácie, ktoré by sme získali spustením reálneho bota a zároveň odpadávajú právne záležitosti, keďže sa nevykonáva škodlivá činnosť. Takisto sa môže využiť jeden systém na sledovanie viacerých botnetov, keďže boty sa navzájom neovplyvňujú. Na druhej strane, vytvoriť nástroj, ktorý je tohoto všetkého schopný, si vyžaduje detailné informácie z analýzy narozdiel od sledovania reálneho bota. Pri implementácií musíme dbať na to, aby sme na každý príkaz vedeli adekvátne reagovať, a tak zabránili detekcii vlastníkom botnetu.

Sledovanie botnetu je veľmi dôležité, aby sme získali detailné informácie o fungovaní botnetu. Môžeme ním zistiť ciele vlastníkov botnetu a dokonca aj získať indície, kto by ho mohol ovládať. Komunikácia môže byť sledovaná, ak sa nepoužívajú príliš zložité systémy na šifrovanie, poprípade dokážeme sledovať iba konkrétne napadnuté systémy, o ktorých máme dostatok informácií, aby sme šifrovanie prelomili. Aby sme sa vyhli obmedzeniam zákona, využívajú sa špecializované nástroje, ktoré si vyžadujú aspoň takú prípravu,

ako by bolo pozorovanie skutočného bota[24, 85] .

2.5 Počítanie veľkosti

Dôležitý aspekt pri hodnotení botnetu je jeho veľkosť, ktorá nám môže byť istým údajom pri hodnotení veľkosti rizika, ktoré môže botnet poskytovať. Ak má vlastník botnetu k dispozícií veľké množstvo systémov naraz, môže napr. pri DDoS útoku vytvoriť omnoho väčší nápor, ktorý dokáže znefunkčniť aj väčšie a lepšie zabezpečené servery.

Pri počítaní veľkosti sú dôležité rôzne parametre. Môže to byť počet aktívnych botov počas jednotlivých častí dňa, ich geografické rozdelenie. Tieto a aj ďalšie informácie o botnete nám môžu dať hlbší prehľad o sile. Tieto parametre sú dôležité napríklad pri DDoS útoku, keď napriek ohromnej celkovej veľkosti botnetu, bude počas útoku aktívnych iba niekoľko členov a útok sa nemusí podariť. Geografické rozdelenie môže mať takisto vplyv na DDoS útoky, keď rôznorodá zmes pri rozdelení sa môže omnoho ťažšie filtrovať alebo ak sa snažíme získať od používateľov napadnutých systémov, môžeme prispôsobiť cenu výkupného podľa bohatosti regiónu [58].

Pri počítaní veľkostí sa využíva niekoľko techník, ktorých výsledky nie vždy korešpondujú s reálnymi údajmi. Na vine môže byť mnoho faktorov. Jeden z problémov môže byť ak počítame iba jednotlivé IP adresy. Vzťah medzi infikovaným systémom a jeho IP adresou je veľa ku veľa. Jeden systém sa môže započítať s niekoľkými rôznymi IP adresami, napr. ak používa na pripojenie dynamickú IP adresu keď DHCP server poskytne rôzne IP adresy alebo ak bol infikovaný systém, ktorý sa postupne pripája z rôznych fyzických miest. Dokonca ani jedna IP nemusí znamenať najviac jeden systém, keď pri preklade adries pomocou NAT sa za jednou adresou môže skrývať viacero napadnutých systémov [39]. Napríklad v článku od Arbor Networks ([63]) sa môžeme dočítať, že zatiaľ čo Arbor networks odhadol veľkosť botnetu na 800 000, Microsoft priamou metódou napočítal až 8 miliónov botov. Podobne aj v článku o získaní kontrole botnetu ([9]) napočítali za 10 dní približne 1,25

milióna IP adries, zatiaľ čo skutočná veľkosť botnetu bola približne 180 000. Na nepresné merania a nadhodnocovanie upozorňuje aj ENISA (European Union Agency for Network and Information Security), ktorá naznačuje že to môže mať za následok nesprávne používanie sily v boji proti škodlivému kódu [26].

Pri počítaní môžu nastať aj ďalšie komplikácie, napr. ak internetový provider blokuje porty, prevádzkuje vlastnú sinkhole (podvrhnutý server), čo môže viesť k podceneniu veľkosti. Takisto k podceneniu môže dôjsť, ak adresy použité pri sinkholingu (kap. 2.5.1) sa nachádzajú na čiernej listine alebo ak jednotlivé boty sú offline. Poprípade môžu čísla nadhodnotiť veľkosť, ak sa útočník aktívne snaží nespresniť výsledné čísla, napríklad tým, že sa jeden bot tvári ako niekoľko nezávislých alebo pri dlhej dobe počítania sa príliš veľa systémov pripojí z rôznych IP adries. V publikácií o fiktívnom botnete Ratbot ([92]) sa môžeme dočítať o technike vytvorenia peer-to-peer botnetu, ktorého presné čísla je nemožné získať. Problémom môže byť technika, v ktorej sa boty presúvajú z jedného botnetu do druhého, čo môže mať za následok nadhodnoteniu počtu botov v oboch prípadoch [5]. V neposlednom rade nám čísla môžu skresliť aj iné tímy, napr. ak aktívne vytvárajú falošných botov do botnetu.

V nasledujúcich častiach vám skúsime priblížiť niekoľko techník, ktoré sa používajú pri zisťovaní počtu botov zapojených do botnetu.

2.5.1 Sinkholing

Pozorovaním komunikácie jednotlivých botov sa zriedkakedy dozvieme informácie o ostatných botov, preto môžeme skúsiť, sledovať komunikáciu C&C serverov, ale to je mnohokrát mimo kompetencie a schopností tích ktorý čelia tejto úlohe. Jednoduchšie je presmerovať komunikáciu do tzv. sinkhole. Sinkholing je metóda pri ktorej sa boty nepripoja na C&C server, ale sú presmerované na iný podvrhnutý server [35, 85, 12]. Všetky spojenia smerom k sinkhole môžu byť analyzované a môžeme získať napríklad IP adresu konkrétnych botov. Počet všetkých spojení a IP adries nám môže dať prehľad

o veľkosti botnetu. Hlavný problém pri tejto technike je možnosť presmerovať všetku komunikáciu na svoj určený server. Zvyčajne sa DNS požiadavky presmerovávajú na lokálne servery, kde môžeme upraviť konfiguráciu tak aby servery odpovedali na požiadavky podvrhnutými IP adresami vždy keď sa niektorý z botov pokúsi zistiť adresu C&C servera.

Tento postup je limitovaný iba na kontrolované siete avšak za určitých podmienok sa dá rozšíriť na celý internet. V kapitole 1.4.1 sme spomínali používa domain generation algorithm, keď sa boty snažia pripájať na generovaný zoznám domén. Pri analýza malwéru či už dynamickej alebo statickej môžeme z bota extrahovať informáciu o DNS adresách na ktoré sa bude neskôr v budúcnosti snažiť pripojiť. Výskumník má v tomto prípade možnosť predbehnúť vlastníka botnetu a zakúpiť si adresu zo získaného zoznamu, ktorá ešte ostala voľná. Zaregistrovanú doménu bude smerovať na svoj server a znova úspešne získa informácie o jednotlivých botov [35].

2.5.2 Sybil-atack

Útok na peer-to-peer botnety. Peer-to-peer siete si zvyčajne pamätajú iba malý počet peerov pripojených do siete, pri posielaní správ inému peerovi, ktorého nemajú v zozname postupne získavajú informácie o peeroch, ktoré sú k nemu bližšie podľa určitých kritérií, až kým sa postupne nedostanú k jeho adrese. Pridaním dostatočného množstva peerov do siete, môžeme dosiahnuť takého dostatočného pokrytia siete nato, aby naše falošné boty boli dominantnou súčasťou routovacieho protokolu. Ak sa bot spýta na adresu iného peera v sieti, môžeme ho aktívne presmerovať na sinkhole alebo iba pasívne počítať informácie, ktoré sa nakoniec skombinujú zo všetkých falošných peerov [72]. Táto metóda ako si neskôr ukážeme má mnoho ďalších využití.

2.5.3 Enumerácia v peer-to-peer infraštruktúre

Existuje aj ďalšia možnosť, ako enumerovať členov peer-to-peer botnetu. Pri pripojení nového peera do siete sa peer potrebuje dozvedieť o ďalších peeroch, aby mohol byť súčasťou routovania v C&C infraštruktúre. Proces kedy peer získava tieto informácie sa nazýva bootstrapping. Pri bootstrappingu sa bot opakovane pýta už známych botov o ich susedoch a postupne si napĺňa svoj zoznam. Na začiatku bootstrappingu má zvyčajne niekoľko fixných botov zapísaných v svojom kóde. Za normálnych okolností peer prestane získavať informácie o ďalších peeroch, keď je jeho zoznam dostatočne dlhý.

Proces bootstrappingu môže byť napadnutý a využitý na získanie informácie o veľkosti botnetu rekurzívnym pýtaním sa peerov na ich susedov až kým neidentifikuje všetkých zúčastnených peerov. Pred začatím enumerácie peer-to-peer botnetu sa pomocou metódy reverzného inžinierstva získajú informácie o protokole, ktorý využíva botnet pri bootstrappingu.

Ak je peer-to-peer sieť rozdelená na niekoľko nezávislých častí nezapočítame všetkých botov. Ďalším problémom je, že nie všetci boti sú dostupný z internetu, napríklad systémy za NAT routermi, firewallmi. Problémy môže spôsobovať aj časová nedostupnosť. Aktívne enumerovanie botov, môže v tomto dôsledku viesť k podceneniu veľkosti botnetu. Problém nastáva aj s rozhodnutím či sme už našli všetkých možných botov, alebo aj s zistením, či sa bot znova prihlásil s rôznou IP adresou, alebo sa jedná o nového [82, 23].

2.5.4 Fast-Flux Polling

Botnety, ktoré závisia na technike fast-flux (kap. 1.4.1), využívajú rýchlo meniace sa DNS záznamy ukazujúce na mnohých klientov, ktorý sa tvária ako proxy vrstva pri komunikácií botov s C&C serverom [21, 64]. DNS záznamy majú zvyčajne krátku dobu platnosti (TTL - Time to live) a server, ktorý poskytuje tieto záznamy býva pod kontrolou útočníka. Ak sa nám podarí pri analýze zistiť DNS adresy, ktoré sa využívajú môžeme ich opakovanie sledovať a tým získať zoznam botov, ktorí boli využívaní na účel proxy. Nie sú

ale garancie, že tieto adresy patria nášmu sledovanému botnetu a tak sa nám môže podariť započítať aj časť iného botnetu. Taktiež zvyčajne množstvo botov, ktoré sa zapájajú aj ako proxy servery je dosť malé s porovnaním celej veľkosti, za následok to môže mať aj nedosiahnuteľnosť niektorých botov z internetu (firewall, NAT), a takisto ani nie všetky dosiahnuteľné musia byť použité. Pri pozorovaní botnetu Storm sa ukázalo, že iba približne 1% všetkých botov býva používané na tento účel [64] Podobne, ako pri ostatných technikách aj tu nastáva problém s dynamickými IP adresami, a tak môžeme jedného bota započítať viackrát a opačne.

2.5.5 DNS cache snooping

Voľný preklad tejto techniky je aj špehovanie DNS cache. Hlavnou úlohou zohráva ukladanie výsledkov pri hľadaní DNS záznamu do cache (dočasnej pamäte). Keď sa klient spýta DNS servera adresu, server sa ju pokúsi rekurzívne pýtaním sa iných DNS serverov zistiť a ak sa mu to podarí, výsledok si zapíše do svojej pamäte a bude si ho pamätať po dĺžku platnosti tohoto záznamu. Dôvodom je zvýšiť výkon a nezaťažovať zbytočne siete [37].

Pamätanie si už pýtaných záznamov sa dá využiť na to, aby sme získali informáciu, či už niekto pred našou žiadosťou poslal rovnakú požiadavku. Ak áno môžeme predpokladať, že niekto z tých, ktorí využívajú daný server je infikovaný. Zistiť to môžeme poslaním špeciálnej nerekurzívnej požiadavky, ktorú ale môžu DNS servery ignorovať alebo analýzou time to live hodnoty, ktorá je súčasťou odpovede. V druhom prípade si môžeme zistiť, aké hodnoty pre time to live vracia pre určite nezapamätané výsledky (pýtaním sa na neexistujúce dómeny) a porovnávať ich s hodnotou, ktorú sme dostali. Nevýhodou je, že táto hodnota sa už uloží do pamäte, a preto po dobu jej uloženia, nedokážeme zistiť či aj niekto iný sa už na ňu pýtal. Táto metóda sa dá pomerne jednoducho rozšíriť aj na zvyšné DNS servery, keďže sa navzájom neovplyvňujú, problém nastáva ak chceme takto sledovať veľké množstvo rôznych adries, keďže mnohé DNS servery majú obmedzenia na množstvo požiadaviek.

Hlavným limitom tejto metódy je jej nepresnosť. Ak sa nám podarí nájsť DNS server, ktorý mal uložené záznamy v pamäti, znamená to, že niekto už poslal rovnakú požiadavku. Nedokážeme, ale zistiť množstvo týchto požiadaviek, a tak ich mohlo byť potenciálne veľké množstvo. Napriek týmto nedostatkom sa môžeme dočítať o úspešnom použití tejto techniky[64].

Kapitola 3

Zneškodnenie botnetu

Získané informácie o botnetoch nám dávajú možnosti, ako predchádzať ich ďalšiemu šíreniu. Cieľom tejto kapitoly je, dať niekoľko návodov, ako botnety zneškodniť. Vzhľadom na to že zneškodnenie nie je jednoduchý a priamočiary proces, rozdelíme si tento proces na 3 časti. V prvej spomenieme ako zabrániť šíreniu už vyvinutých botnetov. V druhej si ukážeme spôsoby, ako bojovať s už rozšírenou hrozbou a ako ukončiť fungovanie botnetu. V poslednej tretej časti si ukážeme, ako znížiť profit produkovaný botnetmi, čo by mohlo mať za následok zníženiu vytvárania nových hrozieb.

3.1 Zabránenie ďalšej infekcie

Zabrániť už existujucej hrozbe nie je jednoduché, no existuje niekoľko riešení, ktoré znižujú riziko aj keď ho úplne nevylučujú. Problémom pri infekcii sú aj samotní používatelia, ktorí vďaka nepozonosti a nízkej informačnej gramotnosti napomáhajú pri infikovaní ich systémov.

3.1.1 Obrnenie systémov

Zabrániť šíreniu botnetu môže dopomôcť aj správna konfigurácia systému. Spomenieme si 4 typy ako ochrániť svoj systém.

- **Aktualizovanie systému** dokáže zabrániť infekcii pomocou chýb, ktoré už boli opravené. Aj keď existujú mnohé chyby, ktoré ešte neboli odhalené (0-day chyby), väčšina útokov prebieha pomocou chýb, ktoré už boli identifikované a boli pre ne vydané aktualizácie.Najnebezpečnejšie sú chyby, ktoré boli objavené nedávno a používatelia si ešte nestihli aktualizovať systém[85].
- **Inštalácia antivírusu** je ďalším dôležitým prvkom ochrany systému. Dokážu chrániť aj proti chybám, ktoré ešte neboli odstránené, detekovať útoky, malwér či chybnú konfiguráciu systému. Pomocou rôznych heuristických metód dokážu zabrániť aj novým druhom útokom avšak nie úplne dokonale. Tvorcovia malwéru sa snažia pomocou rôznych metód (kap. 2.3) zabrániť detekcii. Pomocou rôznych rootkitov a získania správcovských oprávnení sa môže malwér aktívne schovávať pred antivírusom, môže ho deaktivovať alebo inak zabrániť detekcii. Antivírusy nie sú dokonalá ochrana pred botnetmi, ale pridávajú ďalšiu vrstvu, ktorá zabráni úspešnej infekcii.
- **Zníženie počtu chýb** v systéme, sa dá nielen aktualizovaním softvéru, ale aj vypnutím služieb, ktoré nie sú potrebné. Keďže každý softvér je potencionálny zdroj chýb, znížením ich počtu znížime aj šancu na infekciu.
- **Používanie firewallov** môže takisto zabrániť infekciám. Používaním dobrých pravidiel na filtrovanie sa nám môže podariť zablokovať rôzne útoky.

3.1.2 Zvyšovanie povedomia

Na zabránenie infekcii nestačí iba zlepšovať technické riešenia, ale na predídenie infekcii treba naučiť používateľov bezpečnému správaniu. Pri popísaní druhov útokou sme si spomínali metódu sociálneho inžinierstva (kap. 1.3.2). Mnohí používatelia si nie sú vedomí, aké hrozby na nich na internete číhajú, sťahujú si malwér, otvárajú nedôveryhodné maily, stránky a podobne.

Je dôležité naučiť ľudí predísť týmto nástrahám.

Taktiež veľa používateľov nie je technicky zdatných, a treba ponúkať možnosti na jednoduchú detekciu botnetov a nástroje/návody na ich odstránenie. Výsledkom štúdie zistili že takmer polovica útokov bola v súčinnosti s používateľom, a preto je veľmi dôležité vytvoriť povedomie o hrozbách a tak zabrániť ich ďalšiemu šíreniu [85].

3.2 Možnosti zneškodnenia

Táto kapitola nám bude nápomocná pri zodpovedaní na otázku: "Ako zneškodniť už funkčný botnet a odstrániť túto hrozbu?" Popíšeme si niekoľko metód ako znefunkčniť C&C infraštruktúru a taktiež možnosť odstránenia botov zo systému.

Vývoj techník na zneškodenie botnetov mal vplyv aj na vývoj botnetov. Botnety sa naučili používať rôzne metódy na obnovenie kontroly a tak znížili úspešnosť protiopatrení. Narpíklad používajú šifrovanie či verejné podpisy na zabránenie získaniu kontroly, taktiež vyvinuli odolnejšie C&C infraštrukúry, ktoré sme už spomínali (kap. 1.4). Botherderi sa snažia ukrývať za poskytovateľov, ktorí ignorujú požiadavky na odstránenie serverov vykonávajúce škodlivú činnosť a nechávajú si možnosť, ako obnoviť ich funkčnosť.

Problémom pri znefunkčnení infraštrukúry sú boty, ktoré ostávajú v systémoch a sú zdrojom potencionálnych problémov. Tieto boty môžu obsahovať zadné vrátka na obnovenie kontroly alebo môžu spôsobiť nestabilitu systémov. Preto by súčasťou zneškodnenia botnetu mala byť kampaň/nástroje na ostránenie botov zo systému.

3.2.1 Zrušenie C&C serverov

Najjednoduchší nápad ako zneškodniť botnet je znefunkčniť jeho C&C server. Identifikovaním adresy C&C serverov môžeme vystopovať ich poskytovateľov pripojenia a serverov. Ďalším krokom je komunikácia s autoritami po celkom

svete za účelom odpojenia servera od internetu. Tu nastavajú mnohé problémy či už priamo pri komunikácie, alebo problémy so zákonmi v rôznych štátoch a možnostiach prinútenia poskytovateľa vypnúť/odpojiť server. Botherderi väčšinou využívajú poskytovateľov, ktorí im zaručujú anonymitu a ochranu pri vyšetrovaní a pokusoch o odstavenie serveru.

Botnety si vyvinuli mnoho techník, ktoré umožňujú botherderovi obnoviť komunikáciu po odstavení serveru, a preto treba podľa potreby vykonať aj mnohé ďalšie opatrenia. Ak využíva DNS adresu, je potrebné zabrániť jednoduchému presunu serveru na pravdepodobne odolnejšie miesto, kde útočník zotrvá aj náročnejšie útoky [21].

3.2.2 Zmena DNS záznamov

Pri komunikovaní s C&C serverom sa mnohokrát využíva na komunikáciu dómenová adresa. Zmenením záznamov a prepísaním ich na falošné údaje nebude môcť bot zistiť správnu adresu a tým pádom sa nepripojí na server. Takýmto jednoduchým útokom sa ale dá jednoducho predísť. Napríklad generovaním dómenových mien (kap. 1.4.1) by sme museli blokovať veľké množstvo adries, čo môže byť finančne veľmi náročné. Napríklad blokovanie všetkých dómen botnetu Conficker (generoval ich 50 000 denne) by stálo približne 100 miliónov dolárov ročne.

3.2.3 Sinkholing

Bližšie sme túto techniku popísali v kapitole 2.5.1. Cieľom je presmerovať komunikáciu medzi botmi a C&C serverom smerom na náš podvrhnutý server. Najjednoduchší spôsob je zmena DNS záznamov smerujúcich na C&C server.

Presmerovaním všetkej komunikácie už nebude môcť botherder posielať botom príkazy. Problém nastáva ak už bol nejaký útok aktívny, vtedy existuje riziko, že bude útok prebiehať do nekonečna, až kým sa neodstránia boty zo systémov alebo sa nepodnikne iné protiopatrenie. Existuje ale však možnosť,

že útok prestane automaticky alebo po reštarte systému.

Narozdiel od situácie, keď sme používali sinkholing iba na počítanie veľkosti botnetu, ak chceme zneškodniť komunikáciu, musíme vynaložiť väčšie úsilie na to, aby sme napodobnili správanie C&C servera. Ak by sa bot nedokázal pripojiť alebo by nastali problémy pri komunikovaní, mohol by využiť alternatívne možnosti spojenia, ktoré by sme tiež museli presmerovať na sinkhole, aby sme zabránili znovu pripojeniu bota to botnetu.

Zachytávanie komunikácie botnetu môže byť použité bez väčších etických problémov[85]. Vzhľadom na to, že táto technika nemusí byť invazívna, nemusíme komunikovať s napadnutými počítačmi. Problém ale je, ako sa zachovávať k citlivým informáciám, ktoré posielajú boty na server a treba sa k ním primerane správať.

3.2.4 P2P-polluting

Aj peer to peer siete majú achilovú pätu. Vzájomná komunikácia medzi peermi je náchylná na podobné problémy ako zdieľanie súborov s rovnomenným názvom. Každý bot obsahuje informácie iba o obmedzenom množstve peerov. Pomocou analýzy vzoriek botnetu sa získajú informácie o protokole, ktorý používa sieť a nájde sa spôsob, ako prepisovať záznamy o peeroch. Prepísaním informácií na falošných peerov, ktoré ani nemusia byť reálne stratí peer schopnosť komunikovať s okolím. Môže sa stať, že niekto iný má informácie o peerovi, ktorému sme prepísali záznamy a ten ho kontaktuje a môže mu pomôcť obnoviť komunikáciu. Preto nestačí manipulovať iba s jednotlivými peermi, ale treba napadnúť celú sieť [93].

Botnety používajú aj niekoľko ďalších techník, ktoré zabrania rozdeleniu siete. Napríklad botnet Gameover Zeus pri neuspešnom spojení s peermi, využije na obnovenie spojenia systém generovania domén (kap. 1.4.1). Waledac napríklad využíva infraštruktúru verejného kĺúča, aby zabránil vkladaniu falošných údajov. Tieto a rôzne ďalšie techniky môžu vynútiť používanie zložitejších techník, ktoré budú nahrádzať alebo dopĺňať útok na peer-to-peer sieť, aby zabránili obnoveniu komunikácie [78].

3.2.5 Sybil-attack

Technika, ktorá sme popísali v súvislosti s počítaním veľkosti botnetu (kap. 2.5.2) a zároveň je odporúčaná ako alternatívna technika k botnetom odolným proti P2P-polluting [93]. Do peer-to-peer siete botnetu sa vloží množstvo falošných peerov, ktorý získajú dominantnú časť siete. Následne budú títo falošní peerovia meniť routovanie v sieti, a tak znemožnia jeho správne fungovanie.

3.2.6 Vzdialené odstránenie botov

Narozdiel od predchádzajúcich metód, pri tejto priamo odstránime bota z počítača. Predchádzajúce metódy, aj keď mohli úspešne zničiť botnet, nechávali systémy so zraniteľnosťami, ktoré tam zanechal bot alebo môžu byť nestabilné z dôvodu zmien vykonaných botom [85].

Pri analýze bota sme mohli nájsť rôzne bezpečnostné zraniteľnosti, ktoré sa dajú použiť na získanie kontroly nad botom a následne ho odstrániť zo systému. Ďalšou možnosťou je vloženie vlastných príkazov, ktoré sa tvária, že idú od C&C servera (napríklad spomínaným sinkholingom) . Častou funkciou botnetov je ich aktualizácia. Poslali by sme príkaz na aktualizáciu, pričom by sme podvrhli vlastného bota, ktorý by vykonal dezinfekciu a zabezpečil systém voči ďalšej infekcii. Nektoré botnety majú taktiež príkaz na ukončenie činnosti, ktorý by sme tiež mohli poslať a boty by sa samy odstránili.

Ako vždy aj tieto metódy majú svoje nedostatky, prvý technický je, že nemusí sa nám podariť dezinfikovať všetky systémy a ak sa nám aj podarí, môžeme ich nechať zraniteľné, nestabilné. Ak sa nám nepodarí dezinfikovať všetky, tie môžu znova infikovať už vyčistené systémy a obnoviť sieť botnetu. Druhý problém je závažnejší, pretože väčšina štátov nedovoľuje invazívne riešenia, a preto býva tento postup protizákonný a zároveň je takmer nemožné získať povolenie od všetkých používateľov[21].

3.3 Prevencia - zníženie profitu

Komplikovaním nových infekcií, zneškodnením celých botnetov sa nám podarí iba dočasne zlikvidovať hrozby spôsobované botnetmi. Pokiaľ bude možné získavať a výhodné používať botnety na získavanie príjmov, budú sa útočníci snažiť vytvoriť stále komplexnejšie, lepšie techniky na dosiahnutie svojich cieľov. Zatiaľ čo si budú používatelia zosilňovať obranu vo svojich systémoch, útočníci sa budú snažiť vytvoriť nové metódy, aby zaplnili medzeru, ktorá im bola spôsobená. Preto je dôležitá prevencia, ktorá má aj prípadné úspešne útoky urobiť finančne nevýnosné, čím by sa tvorcom botnetu neoplatilo implementovať nové technológie a znižila by sa celková hrozba botnetov.

V najbližších podkapitolách sa pokúsime priblížiť techniky, ktorých cieľom bude čo najviac znížiť profit vytvorený botnetmi. Zvýšením finančných nárokov na implementáciu riešení, skrátením životného cyklu pomocou techník popísaných v predchádzajúcej kapitole a nakoniec znížením hodnoty získaných informácií sa nemusí finančne oplatiť udržiavanie botnetu, a tak sa zníži množstvo kriminality na internete spôsobovanej botnetmi.

3.3.1 Falošné údaje

Distribúciou falošných údajov môžeme znížiť profit a pomáhať tak zmenšiť hrozbu botnetov dvojakým spôsobom[21].

- Bežným využitím botnetov je krádež údajov a ich použitím získanie financií. Monitorovaním botnetu, analýzou sa môžeme dozvedieť, ako získava botnet tieto údaje. Následne sa pokúsime vložiť veľké množstvo falošných údajov. Predpokladá sa, že tieto údaje môžu vniesť istú formu nedôvery medzi jednotlivými účastníkmi, ktorý využívajú tieto informácie. Napríklad zákazník ktorý získa dáta od botherdera sa bude sťažovať na nízku kvalitu a zníži sa aj kvalita botnetovskej infraštruktúry.
- Podobne ako v predchádzajúcom prípade sa môžu vložiť falošné informácie, za účelom ich sledovania. Podobne ako sme písali v kapitole

2.1.5 o spamovacích pasciach, môžeme útočníkom podvrhnúť napríklad čísla bankových účtov, ktoré sú monitorované. Pozorovaním zmien na tomto účte môžeme nájsť a následne potrestať osoby, ktoré sa zúčastňujú protizákonnej činnosti. Predpokladom na úspešnú akciu je, ale veľká spolupráca medzi organizáciami z mnohých krajín (napríklad keď sa peniaze odošlú do inej krajiny).

3.3.2 Blokovanie portu 25

Botnety zvyčajne závisia na službách bežiacich na porte 25, aby rozširovali spam. Problémom sú hlavne otvorené servery na preposielanie mailov, ktoré nepoužívajú autentifikáciu. Namiesto toho by sa mali používať autentifikované a šifrované servery, napríklad na porte 587. Na príklade, ktorý preukázal Schmidt [76] pomocou adaptívneho blokovania portov (na základe množstva posielaných správ), sa preukázali účinky blokovania. Zo vzorky 20000 pripojení sa sťažovalo iba 5 ľudí, počet sťažností na podsieť klesol z približne 100 na 0. Taktiež počet posielaných správ sa znížil o 95%. Skúšobným odblokovaním sa počet posielaného spamu vrátil na pôvodné hodnoty.

3.3.3 Čierne listiny

Ak sa o niektorej službe zistí, že je škodlivá, pridá sa na zoznam, aby sa predišlo komunikácii a šíreniu škodlivých informácií. Existuje niekoľko rôznych čiernych zoznamov, princíp ale ostáva. Ak sa nájde škodlivý obsah, pridá sa do centrálnej databázy. V prípade ak chce niekto pristúpiť k niektorej službe, najprv si overí či je, alebo nie je škodlivá a poprípade odmietne spojenie. Spomenieme si 3 najbežnejšie využitia.

- **Blokovanie mailových serverov**, ktoré posielajú spam. Ak sa zistí, že z niektorej adresy sa posiela spam, pridá sa na zoznam. Následne pri všetkých požiadavkách na preposielanie mailu, si mail server najskôr zistí či je adresa na zozname a ak ju nájde správu neprepošle. Vzhľadom na to, že tieto zoznamy sú často využívané na čiernom trhu sa

predávajú servery, ktoré ešte neboli zablokované [85]. Ako sme už spomínali v kapitole 1.2, bežnou službou botnetou je rozosielanie spamu, zmenšením úspešnosti spamovacích kampaní sa zníži profit a následne sa zníži finančná atraktivita botnetov.

- **Blokovanie webových adries** je podobné ako pri mailových serveroch, ak sa odhalí škodlivý obsah na webových serveroch, tieto sú pridané na zoznam. Následne zvyknú internetové prehliadače zobrazovať varovné hlásenia o výskyte škodlivého obsahu [2]. Taktiež vyhľadávače zvyknú znevýhodňovať alebo vylučovať takéto stránky z výsledkov.
- **Blokovanie C&C serverov** je zamerané priamo na botnety. Príkladom je služba `https://www.abuse.ch/`, ktorá dokáže automaticky analyzovať niektoré vzorky botnetov a získať z nich adresy C&C serverov. Používateľ si môže tieto zoznamy stiahnuť a blokovať spojenia smerujúce na tieto adresy.

Problémom pri čiernych listinách býva neskorá identifikácia obsahu, čím sa spočiatku neobjaví na zozname a nemusí byť blokovaný. Zároveň útočníci môžu pravidelne kontrolovať takéto zoznamy a meniť obsah stránok, či adresy serveru, aby sa vyhli blokovaniu a mohli pokračovať v aktivite.

3.3.4 Používanie DNSSEC

Zvýšenie bezpečnosti na internete, lepšia autorizácia môže znížiť úspešnosť rôznych útokov, napríklad phishingu, ktorý sa šíri pomocou spamu. Toto bude mať za následok zníženie celkového profitu z prevádzkovania botnetu. Taktiež DNSSEC môže pomôcť proti man in the middle útokom, prevádzkovaných napríklad na routeroch [81].

3.3.5 Walled Garden

Termín “walled garden” označuje prostredie, ktoré kontroluje informácie a služby, ktoré môže používateľ používať a tiež sieťové pripojenie. Striktnosť

obmedzení môže byť od slabých až po striktné. V striktnom prostredí býva zvyčajne zamedzený prístup k internetu, zatiaľ čo pri slabých iba k tým, ktoré sa zdajú byť škodlivé a tým ktoré majú upozorniť používateľa na infekciu [57]. Pri aplikovaní obmedzení býva používateľ pomocou vlastných DNS odpovedí presmerovaný na špeciálnu stránku, ktorá ho informuje o infekcii, poskytne pomocné informácie a rady, ako sa zachovať pri takejto situácií, môže mu takisto dať možnosť dočasne alebo trvalo zablokovať aplikáciu obmedzení na jeho pripojenie.

V mnohých prípadoch dokáže ISP (poskytovateľ internetu) detekovať infekciu v sieti, napríklad skúmaním netflow dát, informácií o DDoS útokoch, posielaní spamu (2.1). Taktiež môžu použiť rôzne iné techniky, pričom musia dbať na súkromie zákazníka. ISP býva zväčša veľmi opatrný lebo prípadné falošné detekcie a následne využitie obmedzenie, by mohli mať negatívny vplyv na jeho reputáciu.

Používania rôznych obmedzení na používateľa je citlivá záležitosť. ISP si musí byť istý že používateľ sa časom pripojí na webovú stránku s informáciami o infekcii, môže sa stať že používa iba hlasové služby, cez ktoré sa o nej nedozvie a ISP môže neúmyselne zablokovať volanie na tiesňovú linku. ISP by mal teda opatrne zvážiť, ktoré pripojenia a služby povolí alebo zakáže. Dobrým príkladom je povoliť prístup na stránky bezpečnostného softvéru, ktoré pomôžu pri odstraňovaní malwéru.

3.3.6 Šifrovanie údajov a kontrola prístupu

Cieľom mnohých útokov sú súkromné dáta ku ktorým má prístup bot prítomný v systéme. Dôležitou súčasťou ako zabrániť ich krádeži je šifrovanie systémov a kontrola prístupu. Upozorníme, že nestačí šifrovanie na úrovni diskov, keďže dáta bývajú zvyčajne automaticky dešifrované na požiadanie aplikácie (napr. bota), dôležité je kontrolovať prístup ku súborom osobitne a dešifrovať ich na požiadanie. V firemných sférach je tiež odporúčané zamestnancom priradiť prístup iba k minimálnemu počtu zdrojov potrebných na jeho prácu [4].

Kapitola 4

Bankové botnety

Cieľom bankových botnetov je získať informácie pomocou ktorých by mohli vykonať platby a získať tak peniaze od obetí útokov. Cieľom útoku bývajú hlavne banky s internetovým bankovníctvom, ale ďalšími sú aj sociálne siete, mzdové systémy, internetové obchody, mobilný provideri, pracovné portály a mnohé ďalšie, ktoré by mohli byť nápomocné pri získaní potrebných informácií.

Väčšina bankových botnetov patrí to kategórie crimeware, čo je škodlivý malwér, ktorého účelom je automatizácia škodlivej činnosti. Podobne môžeme nájsť bankové botnety, pod názvom bankové trojany, keďže ich cieľom je získať dáta z počítača [42]. Na úvod kapitoly si spomenieme spoločné črty bankových botnetov - metódy a technológie používanú na získanie údajov. Popíšeme si tiež niekoľko bankových botnetov, ktoré sa vyskytli na internete v poslednej dobe, spomenieme si ich hlavné črty a technológie.

4.1 Metódy a technológie

V nasledujúcej časti si popíšeme najčastejšie používané technológie, ktoré používajú bankové trojany na získanie prístupu k účtom a autorizovanie vlastných platieb. Tieto techniky spadajú do kategórie "Krádež osobných údajov", ktorú sme si spomínali v úvodnej kapitole k botnetom (1.2).

- **Keylogger** slúžia na zachytávanie stlačených kláves. Bot nainštalovaný do systému zaznamenával stlačenia kláves, ktorých skúmaním sa dali zistiť mená, heslá do internetového bankovníctva, či iné dôverné informácie (čísla kreditných kariet, PIN kód). Na to aby útočník získal iba relevantné stlačenia, zvykne zaznamenávať iba stlačenia ak sú aktívne niektoré konkrétne aplikácie poprípade web stránky [38]. Niektoré botnety pri navštívení stránky nezaznamenávajú iba stlačenia kláves, ale robia aj snímky alebo videá z obrazovky. Dôvodom boli rôzne metódy, ktorými sa banky snažili zabrániť získaniu údajov. Využívali napríklad virtuálne klávesnice, ktoré mali zabrániť zaznamenaniu kláves, ale neboli účinné voči zaznamenávaniu obrazovky [13].
- **Man-in-the-browser** (muž v prehliadači) je bežnou technológiou bankových trojanov [43]. S touto technológiou má útočník kontrolu nad spojeniami v prehliadačoch a môže tak čítať a modifikovať komunikáciu či obsah stránok, ktoré sa zobrazia útočníkovi. Kedže útočník napáda priamo prehliadače má prístup k nešifrovaným údajom, zároveň ho ale môžu nové aktualizácie spraviť nefunkčným.
- **Zachytávanie formulárov** ktoré sú potom posielané na C&C server je základným druhom útoku použítím man-in-the-browser. Aj keď ešte niektoré botnety využívajú túto metódu, väčšina vyspelejších botnetov využíva modernejšiu technológiu webinjects.
- **Webinject** (vkladanie do webu) technológia dokáže priebežné sledovať komunikáciu medzi klientom a serverom a modifikovať ju ešte predtým ako sa zobrazí v prehliadači. Užívateľ, server ani prehliadač netušia že prebieha útok a tak sa nezobrazí žiadne upozornenie, ktoré by varovalo užívateľa o možnom úniku dát. Základnou formou útoku je vkladanie/mazanie častí stránky. Táto forma sa podobá na phishing, jej cieľom je získať osobné informácie, napríklad PIN kód, číslo kreditnej karty, ochranných kódov alebo informácie, ktoré sa pýtajú banky pri telefonickom overovaní užívateľa. Cielené stránky nemuseli byť iba internetové bankovníctvo, ale aj rôzne sociálne siete, inštitúcie na ktorých

vkladali rôzne formuláre.

- **Automatizácia platieb** je o niečo zložitejšia forma webinjectu, ktorá dokáže automaticky modifikovať stránky a vykonávať pripojenia tak aby došlo k platbe. Automatické platby sa ale stali čím ďalej náročnejšie, kvôli opatreniam bánk (napríklad pridaním ďalších kontrol, 2-faktorovej autorizácie). Časom sa tento útok stal príliš náročným na implementáciu a preto útočníci vymysleli nové druhy útokov, ktoré obchádzali novo pridané ochranné mechanizmy.
- Ďalšie formy webinjectu pomocou socialného inžinierstva sa snažili presvedčiť užívateľa aby si nainštaloval mobilnú aplikáciu do mobilu, ktorá následne zachytávala autorizačné SMS a mohla ich preposielať útočníkovi na server [49]. Tiež mohli presvedčiť útočníka aby poslal peniaze na konkrétny účet, pretože mu boli omylom zaslané a modifikáciou stránky mohol nadobudnúť dojem že sa mu naozaj zvýšil kredit a preto ochotne túto transakciu vykoná. Ďalej ho mohli kvôli kalibrácií systému presvedčiť aby poslal skúšobnú platbu [8].
- **Sledovanie užívateľa** pomocou webových kamier a mikrofónov s útokom webinject mohol mať útočník prehľad o situácií, napríklad videl ako užívateľ reaguje na zmeny v prehliadači. Tiež mohol **odpočúvať rozhovory**a získať z nich tak dôverné informácie, napríklad ak si užívateľ všimol podozrivé správanie zavolal svojej banke, kde na úvodnú identifikáciu odpovie operátorovi na otázky, čo ale zaznamená útočník a zneužije pri ďalších krokoch útoku [18].
- **Vzdialený prístup** môže byť užitočný napríklad ak sa útočníkovi nepodarí vykonať platbu automaticky, ale získa dostatočne veľa údajov na tom aby ju mohol vykonať sám. Niektoré webstránky, kontrolujú pôvodcu pripojenia a dovoľujú pripojenie iba z niektorých adries alebo z už overeného prehliadača. VNC je technológia, ktorá dáva prístup útočníkovi na systém a dovoľuje mu ho ovládať, čím úspešne obíde kontrolu a podarí sa mu vykonať transakciu. Niekedy taktiež posielajú peniaze

cez niekoľko takto ovládaných účtov, aby sa peniaze nedali vystopovať až k útočníkovi. Podobné využitie má služba backconnect, ktorá dokáže tunelovať komunikáciu cez napadnutý počítač a tak obísť niektoré kontroly.

4.2 Ukážky botnetov

Cieľom tejto sekcie bude prezentovať niektoré botnety, ktoré sa vyskytli na internete za posledné roky. Napriek veľmi veľkej snahe o ich zničenie sa útočníci z každého útoku otrasú a prinesú do tejto kategórie novšie a hlavne odolnejšie technológie. Niektoré botnety napríklad Gameover Zeus sa skladali z obrovského počtu botov(cca. 500 000 - 1 milión) a spôsobili škody za niekoľko 100 miliónov dolárov. Trendom botnetov je tiež presúvať sa na ázijský trh, ktorý má slabšie zabezpečenie.

Ako sme už spomínali väčšina botnetov je vo forme crimeware, navyše si spomenieme že väčina najúspešnejších botnetov sú/boli ponúkané na internete, kde sa dali kúpiť. Vzhľadom na to že v rokoch 2011 a 2013 unikli na internet zdrojové súbory najprv Zeusa a neskôr Carberpu mnoho ďalších botnetov prezvalo ich funkcionalitu. [42, 43, 65]

4.2.1 ZeuS

Najznámejší bankový botnet, prvý krát objavený v roku 2007, nazýva sa taktiež PRG alebo Zbot. Pravdepodobne bol vyvinutý ruskými vyjovármi, ktorý ho predávali na čiernom trhu. V roku 2011 sa na internet dostali jeho zdrojové kódy, vďaka čomu takmer každý bankový trojan obsahuje niektorú z jeho funkcionalít. Kód botnetu sa skladá z 3 častí - bota, generátora botov a kontrolného panelu. Generátor botov vytvára spustiteľnú verziu bota s konkrétnym konfiguráciou. Bot napadne systém a ukradne informácie, ktoré následne odošle na C&C server[43, 7], kde si ich môže botherder pomocou panelu prečítať.

Botnet Zeus využíva centralizovanú infraštrukúru pričom adresa sa fixne nastaví pomocou konfiguračného súboru. Súčasťou býva aj záložná adresa na ktorú sa pripojí ak je primárny server nedostupný. Na komunikáciu využíva http komunikáciu na porte 80, ktorá býva typicky neblokovaná a šifruje ju pomocou algoritmu RC4[7]. V roku 2013 sa objavila na internete 64-bitová verzia, ktorej komunikácia medzi C&C serverom a botom prebiehela cez anonymizačnú sieť Tor(kap. 1.4.1).

Krádež údajov prebiehala pomocou keyloggera, ktorý sa aktivoval pri stránkach, ktoré boli vylistované v konfiguračnom súbore. Nachádzali sa tam tiež jednoduché pravidlá pre útok webinject, ktoré vkladali/vymazavali časti stránok. Ako sme už spomínali pri webinjectoch banky pridali 2-faktorovú autentifikáciu, jednou z možností je aj posielanie autentifikačných kódov na mobilné telefóny. Práve tento problém v niektorých prípadoch riešila mobilná časť Zeusa - ZitMo (Zeus in the Mobile) [49].

4.2.2 Citadel

Bankový trojan založený na zdrojových kódoch Zeusa, ktorý sa objavil tesne po ich zverejnený v roku 2011. Oproti Zeusovi obsahuje niekoľko vylepšení, ktoré ho zaradzujú do samostatnej kategórie. Podobne ako Zeus aj Citadel používa centralizovanú infraštruktúru, namiesto šifrovania pomocou RC4, ale prešiel na modifikovanú verziu algoritmu AES. Algoritmus RC4 bol ale naďalej využívaný na šifrovanie ukladaných dát, oproti Zeusovi ale obsahoval o xorovanie počiatočnej hodnoty navyše.

Pridal možnosť útoku na Google Chrome, vylepšil technológiu pre útok webinject, ktorá sa už môže dynamicky meniť bez potreby novej konfigurácie. Taktiež obsahuje detekciu sandboxu, aby skryl skutočné adresy serverov. Novinkou je tiež možnosť vykonávať pomocou skriptu predefinované úlohy, napríklad vypnutie/reštart systému, získanie informácií o systéme, hľadanie, sťahovanie a nahrávanie súborov či možnosť dynamicky meniť sledované stránky. Pridala sa tiež možnosť DDoS útokov alebo filtrovanie spojení na stránok (napríklad na zabránenie prístupu na stránky s antivírusmi, aby sa

nemohli aktualizovať a nedetekovali tak bota)[31, 44] .

4.2.3 KINS

alebo tiež ZeusVM je taktiež postavený na Zeusovi v roku 2011. Podobne ako Citadel využíval vylepšené šifrovanie, detekciu sandboxu a na C&C server zvykol posielať informácie o bezpečnostných produktoch prítomných v systéme. Zaujímavý sa stal aj pre ďalšie funkcie, ktoré využíval.

Prvou je využívanie virtuálneho stroja, ktorý interpretoval vlastný kód a tak sa snažil vyhnúť detekcii. Ďalšou novinkou, ktorú priniesol bolo využívanie steganografie na uloženie a sťahovanie konfiguračných súborov. ZeusVM pripojil zašifrovanú verziu konfiguračného súboru k obrázku, obrázok sa dal otvoriť a normálne sa zobrazoval v prehliadačoch, čím nevyvolal podrozrenie a útočník ho tak mohol uložiť na ľubovoľné úložiska. Treťou významnou funkciou, ktorú využíval je "neviditeľná trvalosť", ktorá zabezpečovala trvalé infikovanie systému aj po reštarte, ale počas behu systému sa to nedalo identifikovať. Využíval sa na to jednoduchý trik, ktorý pri inicializácii bota odstráni stopy po infekcii a tesne pred vypnutím systému znova zapíše všetky zmeny potrebné na opätovné spustenie [42, 66].

4.2.4 GameoverZeus

Krátko po úniku Zeusa sa objavila aj verzia, ktorá nepoužívala centralizovaný C&C model ale P2P model komunikácie. Používal P2P protokol založený na Kademlia P2P protokol, ktorý na počítanie vzdialenosti využíva xor. Na komunikáciu využíva náhodné porty z rozpätia 10 000 - 30 000. Protokol UDP sa využíva pre potreby P2P siete a TCP spojenia pre potreby botnetu. Navyše aby sa zabránilo kompromitovaniu siete využíva sa podpisovanie správ pomocou RSA kľúčov [82]. Gameover Zeus využíva aj systém na generovanie adries DGA (kap. 1.4.1), ktorého cieľom je zabrániť rozdeleniu siete a strate botov, denne generuje 1000 dómen.

Pri kradnutí financií z účtov botherderi zvykli využívať aj iné botnety

na DDoS útok. Motív útoku bolo zabránenie pripojenia sa obete na účet a zrušiť plánovanú platbu.

4.2.5 Carberb

Prvý krát sa objavil na internete v roku 2009. Cieľom sú hlavne Ruské banky a taktiež podobne ako Zeus, unikli v roku 2013 jeho zdrojové súbory na internet. Skladá z 3 častí - bot, generátor botov a kontrolný panel. Používa centralizovanú C&C infraštruktúru, ktorej komunikácia je šifrovaná pomocou RC4.

Carberp mal modulárnu architektúru, ktorá mu umožnila rozširovať svoju funkcionalitu za behu. Dokázal taktiež sledovať komunikáciu vo väčšine hlavných prehliadačov, čím mohol vykonávať väčšinu klasických webinject útokov. Obsahoval tiež pluginy na vzdialený prístup, zachytávanie obrazovky, DDoS útoky. Pomocou prepísania rôznych API funkcií v knižniciach, dokázal zaznamenávať aj FTP, POP3 pripojenia.

Na svoju ochranu sa snažil identifikovať antivírusové ochrany, ktoré deaktivoval. Používal taktiež rôzne obfuskačné metódy (napr. šifrovanie stringov) a polymorfické šifrovanie [61, 41].

4.2.6 Dyre

Objavil sa tesne po zneškodnení botnetov GameoverZeus a Shylock v roku 2014. Na pripájanie k C&C serveru využíva sieť proxy serverov, pričom skoršie verzie nepoužívali šifrovanie. Novšie verzie využívajú na komunikáciu šifrovanie pomocou SSL a vlastný šifrovací mechanizmus na posielanie správ od servera, pričom sú navyše podpísané RSA kľúčom. Podpisovanie sa využíva aj pri kontrole dôveryhodnosti modulov a konfigurácie.

K primárnej možnosti pripojenia na C&C server pomocou proxy serverov používa Dyre aj 2 záložné možnosti. Prvou je používanie systému generovania domén, ktorý generuje 1000 adries denne na 8 top-level doménach. Druhou možnosťou je využite anonymizačnej sieti I2P.

Dyre má modulárnu architektúru. Dokáže sledovať a modifikovať komunikáciu v prehliadačoch Chrome, Firefox, Internet explorer. Skoršie verzie iba posielali nešifrovanú komunikáciu na C&C server. Novšie verzie si podobne ako ostatné botnety osvojili techniku webinject, pričom Dyre dokáže modifikovať aj stránky za behu, čo mu umožňuje obchádzať 2-faktorovú autentifikáciu. Najnovšie verzie pridali aj kontrolu proti dynamickej analýze pomocou sandboxu[10, 42, 74].

Kapitola 5

Smer vývoja

Smer vývoja botnetov je ťažko predikovateľný. Môžeme skúsiť hádať na základe posledného vývoju aké nové technológie začnú botherderi využívať a ako sa zmení vývin botnetov. V najbližších častiach popíšeme niektoré trendy, ktorými by sa botnety mohli uberať [77, 19, 48].

- **Sociálne siete** poskytujú pre botherderov niekoľko výhod. Obrovské množstvo ich používateľov je vynikajúci priestor na ich šírenie. Užívatelia sociálnych sietí sú dôveryhodný voči odkazom, ktoré dostanú od svojich priateľov a prípadne šírenie škodlivého obsahu je účinnejšie. Sociálne siete taktiež poskytujú priestor nielen na komunikáciu užívateľov, ale medzi botami a C&C serverom. Ako sme už spomínali už sa objavili niektoré botnety využívajúce sociálne siete ako transportný kanál a aj ďalšie botnety by mohli využívať tento trend.
- **Anonymizačné služby** sa v roku 2014 stali veľmi populárne medzi bankovými botnetmi. Používanie sietí ako Tor či I2P pomáha zvýšiť odolnosť botnetu a anonymita chráni botherdera pred jeho identifikáciou bezpečnostnými agentúrami.
- **Mobilné botnety** sa už dávnejšie stali realitou. Je veľmi známe že zločinci sa rýchlo prispôsobujú trendom a využívajú to vo svoj prospech. Rozmach mobilných zariadení, ktoré sú pripojené na internet využili

aj tvorcovia botnetov. Objavili sa botnety, ktorých hlavným cieľom je získanie dôverných informácií z mobilných zariadení (fotiek, dokumentom, číslam, ...). Môžu špehovať užívateľa alebo sa využívajú pri obchádzaní 2-faktorovej komunikácie. Niektoré taktiež volávajú alebo posielajú SMS na platené čísla.

- **Routre a Internet vecí** podobne ako mobilný segment prinášajú nové možnosti pre botnety. Dôvodom je ich veľký počet, čo vyhovuje botnetom a zároveň sú mnoho krát neaktualizované a preto ich nie je ťažké napadnúť a infikovať. Môžu byť využívané napríklad na rozposielanie spamu [67].
- **Malé botnety** bývajú využívané na špecializované útoky. Ich hlavným cieľom krádež údajov, ktoré majú vysokú finančnú alebo politickú hodnotu. Narozdiel od veľkých botnetov nevyvolávajú veľkú pozornosť, aby ostali v tajnosti.
- **Cielené sociálne inžinierstvo** môže zvýšiť úspešnosť infekcie. Využitím súkromných informácií (napríklad zdravotného stavu) môže byť použitie techník sociálneho inžinierstva. Napríklad užívateľ, ktorý v maily nájde informácie o svojom stave môže mať väčšiu tendenciu otvoriť škodlivú prílohu.
- **Využívanie čierneho trhu** na urýchlenie vývoja. Používanie komerčných modulov, môže uľahčiť vývoj botnetov a ich šírenie. Botherder si zakúpi prístup k downloaderom/dropperom, ktorých úlohou bude infikovať počítač. Taktiež čierny trh so zraniteľnosťami môže priniesť nové formy infekcií.

Záver

Snažili sme sa popísať problematiku botnetov z technického hľadiska a zároveň aj z pohľadu botherdera, kde sme sa zamerali na jeho motiváciu. Písali sme aj o možnostiach analýzy a detekcie botnetov, kde sme si spomenuli jednotlivé kategórie techník, ktoré sa používajú.

Botnety boli označené ako jedna z najväčších hrozieb internetu čo podnietilo vznik mnohých techník, ktorých cieľom je znížiť túto hrozbu. V prvom rade sme si spomenuli, že je dôležité zabezpečiť jednotlivé systémy, aby sa infekcia nemohla ľahko rozširovať. V druhom rade je dôležité zneškodniť už existujúce botnety a v poslednom rade sme popísali možnosti prevencie, ktorých cieľom bolo znížiť úspešnosť útokov a ziskov súvisiacich s botnetmi.

Podrobnejšie sme si popísali bankové botnety, kde sme si spomenuli konkrétne techniky, ktorými uskutočňujú svojé útoky. Taktiež sme si rozobrali funkčnosť niekoľkých botnetov, ktoré sa objavili na internete za posledné obdobie.

Na záver sme si popísali akým smerom sa mení technika botnetov a aké trendy boli zaznamenané za posledné roky.

Práca priniesla ucelený prehľad do problematiky a do budúcnosti by sa na ňu mohlo nadviazať štúdiami o konkrétnych botnetoch alebo priblížením technických detailov niektorých techník, ktoré využívajú botnety. Taktiež by sa nadväzujúce práce mohli zaoberať navrhnutím nových metód analýzy a detekcie botnetov.

Literatúra

[1] About project honey pot. [Citované 2015-5-14] Dostupné z `http://www.projecthoneypot.org/about_us.php`.

[2] How does built-in phishing and malware protection work? [Citované 2015-5-10] Dostupné z `https://support.mozilla.org/en-US/kb/how-does-phishing-and-malware-protection-work`.

[3] Dancho Danchev. Zeus crimeware using amazon's ec2 as command and control server, 2009. [Citované 2015-5-5] Dostupné z `http://www.zdnet.com/article/zeus-crimeware-using-amazons-ec2-as-command-and-control-server/`.

[4] Daniel Olejár a spol. Študijné materiály k štandardom základných znalostí ib, 2014. [Citované 2015-5-11] Dostupné z `http://www.informatizacia.sk/ext_dok-stud_2014_02_it_ib_ucitelia/16983c`.

[5] Moheeb et al. Abu Rajab. A multifaceted approach to understanding the botnet phenomenon. 2006.

[6] Arini Balakrishnan and Chloe Schulze. Code obfuscation literature survey. *CS701 Construction of Compilers*, 19, 2005.

[7] H. Binsalleeh, T. Ormerod, A. Boukhtouta, P. Sinha, A. Youssef, M. Debbabi, and L. Wang. On the analysis of the zeus botnet crime-

ware toolkit. In *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference on*, pages 31–38, Aug 2010.

[8] Jean-Ian Boutin. The evolution of webinjects. 2014. [Citované 2015-5-16] Dostupné z `https://www.virusbtn.com/pdf/conference/vb2014/VB2014-Boutin.pdf`.

[9] Stone-Gross Brett and Cova Marco et al. Your botnet is my botnet: analysis of a botnet takeover. 2009.

[10] Pallav Khandhar Brett Stone-Gross. Dyre banking trojan, 2014. [Citované 2015-5-25] Dostupné z `http://www.secureworks.com/cyber-threat-intelligence/threats/dyre-banking-trojan/`.

[11] Brian Krebs . Most malware tied to 'pay-per-install' market, 2011. [Citované 2015-5-1] Dostupné z `http://www.technologyreview.com/news/424241/most-malware-tied-to-pay-per-install-market/`.

[12] Guy Bruneau. Dns sinkhole, 2010. [Citované 2015-5-12] Dostupné z `http://www.sans.org/reading-room/whitepapers/dns/dns-sinkhole-33523`.

[13] ET Bureau. Online banking primer: Use virtual keypad for safe transaction, 2011. [Citované 2015-5-18] Dostupné z `http://articles.economictimes.indiatimes.com/2011-02-17/news/28554909_1_password-banking-keypad`.

[14] Carl Saturnino. "chameleon" first botnet tied to online display ad click fraud, 2013. [Citované 2015-5-1] Dostupné z `https://zvelo.com/chameleon-first-botnet-tied-to-online-display-ad-click-fraud/`.

[15] CERT-UK. Code obfuscation. [Citované 2015-5-14] Dostupné z `https://www.cert.gov.uk/wp-content/uploads/2014/11/Code-obfuscation.pdf`.

[16] Philippe Charland, Benjamin CM Fung, and Mohammad Reza Farhadi. Clone search for malicious code correlation. 2012.

[17] Chris Preimesberger. Ddos attack volume escalates as new methods emerge, 2014. [Citované 2015-5-1] Dostupné z `http://www.eweek.com/security/slideshows/ddos-attack-volume-escalates-as-new-methods-emerge.html`.

[18] Lucian Constantin. Banking malware spyeye steals info by hijacking webcams and mics, 2012. [Citované 2015-5-18] Dostupné z `http://www.techworld.com/news/security/banking-malware-spyeye-steals-info-by-hijacking-webcams-mics-3359432/`.

[19] EMC Corporation. An inside look at the changing threat landscape, 2014. [Citované 2015-5-20] Dostupné z `http://www.emc.com/collateral/white-paper/rsa-white-paper-cybercrime-trends-2015.pdf`.

[20] Damballa Labs. Dgas in the hands of cyber-criminals, 2014. [Citované 2015-5-5] Dostupné z `https://www.damballa.com/downloads/r_pubs/WP_DGAs-in-the-Hands-of-Cyber-Criminals.pdf`.

[21] Daniel Plohmann et al. Botnets: Detection, measurement, disinfection & defence, 2011. [Citované 2015-5-10] Dostupné z `https://www.enisa.europa.eu/activities/Resilience-and-CIIP/critical-applications/botnets/botnets-measurement-detection-disinfection-and-defence`.

[22] Dell SecureWorks Counter Threat Unit. Cryptowall ransomware, 2014. [Citované 2015-5-5] Dostupné z `http://www.secureworks.com/cyber-threat-intelligence/threats/cryptowall-ransomware/`.

[23] David Dittrich and Sven Dietrich. Discovery techniques for p2p botnets. 2008.

[24] Dr. Jose Nazario. Botnet tracking:tools, techniques, and lessons learned, 2010. [Citované 2015-5-5] Dostupné z `http://www.blackhat.com/presentations/bh-dc-07/Nazario/Presentation/bh-dc-07-Nazario.pdf`.

[25] Manuel et al. Egele. A survey on automated dynamic malware-analysis techniques and tools. *ACM Computing Surveys (CSUR)*, 44(2), 2012.

[26] Tom Espiner. Botnet size may be exaggerated, says enisa, 2011. [Citované 2015-5-12] Dostupné z `http://www.zdnet.com/article/botnet-size-may-be-exaggerated-says-enisa/`.

[27] Bayer Ulrich et al. Dynamic analysis of malicious code. *Journal in Computer Virology*, 2(1):67–77, 2006.

[28] Egele Manuel et. al. A survey on automated dynamic malware-analysis techniques and tools. *ACM Comput. Surv.*, 2008. Dostupné z `http://doi.acm.org/10.1145/2089125.2089126`.

[29] Moser et al. Limits of static analysis for malware detection. 2007.

[30] Rahimian Ashkan et al. Resource: A framework for online matching of assembly with open source code. In *Foundations and Practice of Security*, pages 211–226. 2013.

[31] Rahimian Ashkan et al. On the reverse engineering of the citadel botnet. In *Foundations and Practice of Security*, pages 408–425. 2014.

[32] F-secure. Threat description worm:w32/downadup.al, 2010. [Citované 2015-5-5] Dostupné z `https://www.f-secure.com/v-descs/worm_w32_downadup_al.shtml`.

[33] F-secure. Lizard stresser runs on hacked home routers, 2015. [Citované 2015-5-5] Dostupné z `http://krebsonsecurity.com/2015/01/lizard-stresser-runs-on-hacked-home-routers/`.

[34] Forensiq. Forensiq botnet project, 2014. [Citované 2015-5-1] Dostupné z `https://vimeo.com/108518208`.

[35] Albert Fruz. Understanding dns sinkholes – a weapon against malware, 2014. [Citované 2015-5-12] Dostupné z `http://resources.infosecinstitute.com/dns-sinkhole/`.

[36] GamingSupermarket. Korean poker hackers arrested, 2010. [Citované 2015-1-20] Dostupné z `http://poker.gamingsupermarket.com/news/4660/korean-poker-hackers-arrested`.

[37] Luis Grangeia. Dns cache snooping or snooping the cache for fun and profit, 2004. [Citované 2015-5-12] Dostupné z `http://cs.unc.edu/~fabian/course_papers/cache_snooping.pdf`.

[38] Nikolay Grebennikov. Keyloggers: How they work and how to detect them, 2007. [Citované 2015-5-18] Dostupné z `https://securelist.com/analysis/publications/36138/keyloggers-how-they-work-and-how-to-detect-them-part-1/`.

[39] Julian B Grizzard, Vikram Sharma, Chris Nunnery, Brent ByungHoon Kang, and David Dagon. Peer-to-peer botnets: Overview and case study. 2007.

[40] Fatih Haltas, Erkam Uzun, Necati Siseci, Abdulkadir Posul, and Bakir Emre. An automated bot detection system through honeypots for large-scale. In *Cyber Conflict (CyCon 2014), 2014 6th International Conference On*, pages 255–270. IEEE, 2014.

[41] DAVID HARLEY. Evolution of win32carberp: going deeper, 2011. [Citované 2015-5-20] Dostupné z `http://www.welivesecurity.com/2011/11/21/evolution-of-win32carberp-going-deeper/`.

[42] Dell SecureWorks Counter Threat Unit(TM) Threat Intelligence. Banking botnets persist despite takedowns. [Citované 2015-5-16] Do-

stupné z http://www.secureworks.com/assets/pdf-store/other/banking-botnets-persist-2015.pdf.

[43] Dell SecureWorks Counter Threat Unit(TM) Threat Intelligence. Top banking botnets of 2013. [Citované 2015-5-16] Dostupné z http://www.secureworks.com/cyber-threat-intelligence/threats/top-banking-botnets-of-2013/.

[44] Dell SecureWorks Counter Threat Unit(TM) Threat Intelligence. Updates to the citadel trojan, 2012. [Citované 2015-5-20] Dostupné z http://www.secureworks.com/cyber-threat-intelligence/threats/updates-to-the-citadel-trojan/.

[45] Jamie Riden. How fast-flux service networks work, 2008. [Citované 2015-5-5] Dostupné z http://www.honeynet.org/node/132.

[46] John Graham-Cumming. Inside shellshock: How hackers are using it to exploit systems, 2014. [Citované 2015-5-5] Dostupné z https://blog.cloudflare.com/inside-shellshock/.

[47] Jose Nazario. Twitter-based botnet command channel, 2009. [Citované 2015-5-5] Dostupné z http://www.arbornetworks.com/asert/2009/08/twitter-based-botnet-command-channel/.

[48] Ahmad Karim, Rosli Bin Salleh, Muhammad Shiraz, Syed Adeel Ali Shah, Irfan Awan, and Nor Badrul Anuar. Botnet detection techniques: review, future trends, and issues. *Journal of Zhejiang University SCIENCE C*, 15(11):943–983, 2014.

[49] Kaspersky. Teamwork: How the zitmo trojan bypasses online banking security, 2011. [Citované 2015-5-20] Dostupné z http://www.kaspersky.com/about/news/virus/2011/Teamwork_How_the_ZitMo_Trojan_Bypasses_Online_Banking_Security.

[50] Kathryn Stephens. Malware command and control overview, 2009. [Citované 2015-5-5] Dostupné z `http://www.nsci-va.org/WhitePapers/2010-12-30-Malware%20C2%20Overview-Stephens.pdf`.

[51] Kelly Jackson Higgins. Zero-day malvertising attack went undetected for two months, 2015. [Citované 2015-5-5] Dostupné z `http://www.darkreading.com/attacks-breaches/zero-day-malvertising-attack-went-undetected-for-two-months/d/d-id/1320092`.

[52] Kris Kendall. Practical malware analysis. 2007.

[53] Kevin Springborn. Inside a botnet: Athena and ad fraud, 2014. [Citované 2015-5-1] Dostupné z `http://www.comscore.com/Insights/Blog/Inside-A-Botnet-Athena-and-Ad-Fraud`.

[54] Marián Krčmárik. Analýza možnosti detekcie botnet sietí pomocou netflow dát. Bakalárska práca, Masarykova Univerzita, 2009.

[55] Lala Manly. New zeus gameover employs dga and fast flux techniques, 2014. [Citované 2015-5-5] Dostupné z `http://www.trendmicro.com/vinfo/us/threat-encyclopedia/spam/578/new-zeus-gameover-employs-dga-and-fast-flux-techniques`.

[56] Lavasoft. Malware from a to z. [Citované 2015-5-5] Dostupné z `http://www.lavasoft.com/mylavasoft/securitycenter/spyware-glossary`.

[57] et al. Livingood, Mody. Recommendations for the remediation of bots in isp networks, 2012. [Citované 2015-5-10] Dostupné z `http://tools.ietf.org/html/rfc6561`.

[58] CIRCL Computer Incident Response Center Luxembourg. Tr-33 analysis - ctb-locker / critroni, 2015. [Citované 2015-5-12] Dostupné z `https://www.circl.lu/pub/tr-33/`.

[59] M3AAWG. M3aawg best current practices for building and operating a spamtrap, 2013. [Citované 2015-5-12] Dostupné z `https://www.maawg.org/sites/maawg/files/news/M3AAWG_Spamtrap_Operations_BCP-2013-10.pdf`.

[60] McAfee Labs. Threats report june 2014, 2014. [Citované 2015-1-20] Dostupné z `http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q1-2014.pdf`.

[61] Jim McKenney. Malware: New capabilities & directions, 2012. [Citované 2015-5-20] Dostupné z `http://www.dailysafetycheck.com/v/vspfiles/WhitePaper_2012_Botnet.pdf`.

[62] Microsoft. Microsoft security bulletin ms08-067 - critical, 2008. [Citované 2015-5-5] Dostupné z `https://technet.microsoft.com/library/security/ms08-067`.

[63] Jose Nazario. Measuring botnet populations, 2012. [Citované 2015-5-12] Dostupné z `www.arbornetworks.com/asert/2012/05/measuring-botnet-populations/`.

[64] Jose Nazario and Thorsten Holz. As the net churns: Fast-flux botnet observations. 2008.

[65] AURELIAN NEAGU. The top 10 most dangerous malware that can empty your bank account, 2014. [Citované 2015-5-20] Dostupné z `https://heimdalsecurity.com/blog/top-financial-malware/`.

[66] Pierluigi Paganini. Detected new zeus variant which makes use of steganography, 2012. [Citované 2015-5-20] Dostupné z `http://securityaffairs.co/wordpress/22334/malware/zeus-banking-malware-nestles-crucial-file-photo.html`.

[67] Paul. When the internet of things attacks! parsing the iot botnet story, 2014. [Citované 2015-5-

25] Dostupné z https://securityledger.com/2014/01/when-the-internet-of-things-attacks-parsing-the-iot-botnet-story/.

[68] Peter Košinár. Big bucks for a little bang, 2011. [Citované 2015-5-1] Dostupné z https://www.youtube.com/watch?v=e2ieH4Haw4s.

[69] Peter Košinár. What cybercriminals learned from the traditional crooks, 2014. [Citované 2015-5-1] Dostupné z https://www.youtube.com/watch?v=o2QeLGUsGks.

[70] Peter Kálnai, Jaromír Hořejší. Dissecting banking trojan carberp, 2013. [Citované 2015-5-1] Dostupné z http://www.rsaconference.com/writable/presentations/file_upload/ht-t06-dissecting-banking-trojan-carberp_copy1.pdf.

[71] Pierluigi Paganini. Hunting malware in the deep web, 2015. [Citované 2015-5-5] Dostupné z http://resources.infosecinstitute.com/hunting-malware-deep-web/.

[72] Davis Carlton R and et al. Sybil attacks as a mitigation strategy against the storm botnet. 2008.

[73] Babak Bashari Rad, Maslin Masrom, and Suhaimi Ibrahim. Camouflage in malware: from encryption to metamorphism. 2012.

[74] Aviv Raff. New dyre version- yet another malware evading sandboxes, 2015. [Citované 2015-5-30] Dostupné z http://www.seculert.com/blog/2015/04/new-dyre-version-evades-sandboxes.html.

[75] Mike Schiffman. A brief history of malware obfuscation. [Citované 2015-5-14] Dostupné z http://blogs.cisco.com/security/a_brief_history_of_malware_obfuscation_part_1_of_2.

[76] Jonathan Schmidt. Dynamic port 25 blocking to control spam zombies. In *CEAS*, 2006.

[77] Sérgio SC Silva, Rodrigo MP Silva, Raquel CG Pinto, and Ronaldo M Salles. Botnets: A survey. *Computer Networks*, 57(2):378–403, 2013.

[78] G. Sinclair, C. Nunnery, and B.B.-H. Kang. The waledac protocol: The how and why. In *Malicious and Unwanted Software (MALWARE), 2009 4th International Conference on*, pages 69–77, 2009.

[79] Sophos Press. Zero-day vulnerability allows usb malware to run automatically, 2010. [Citované 2015-5-5] Dostupné z `https://www.sophos.com/en-us/press-office/press-releases/2010/07/stuxnet.aspx`.

[80] Steve Stasiukonis. Social engineering, the usb way, 2006. [Citované 2015-5-5] Dostupné z `http://www.darkreading.com/attacks-breaches/social-engineering-the-usb-way/d/d-id/1128081?`

[81] Alastair Stevenson. Hackers hijack 300,000 soho routers with man-in-the-middle attacks, 2011. [Citované 2015-5-10] Dostupné z `http://www.v3.co.uk/v3-uk/news/2331953/hackers-hijack-300-000-soho-routers-with-man-in-the-middle-attacks`.

[82] Brett Stone-Gross. The lifecycle of peer-to-peer (gameover) zeus, 2012. [Citované 2015-5-12] Dostupné z `http://www.secureworks.com/cyber-threat-intelligence/threats/The_Lifecycle_of_Peer_to_Peer_Gameover_ZeuS/`.

[83] Swati Khandelwal. China-made e-cigarette chargers could infect your computer with virus, 2014. [Citované 2015-5-5] Dostupné z `http://thehackernews.com/2014/11/china-made-e-cigarette-chargers-could_26.html`.

[84] Dmitry Tarakanov. The inevitable move – 64-bit zeus enhanced with tor, 2011. [Citované 2015-5-20] Dostupné z `https://securelist.com/blog/events/58184/the-inevitable-move-64-bit-zeus-enhanced-with-tor/`.

[85] H. Tiirmaa-Klaar, J. Gassen, E. Gerhards-Padilla, and P. Martini. *Botnets*. SpringerBriefs in Cybersecurity. Springer, 2013.

[86] Tom Brennan, Wong Onn Chee. Owasp plan - strawman - layer_7_ddos. [Citované 2015-5-1] Dostupné z `https://www.owasp.org/images/4/43/Layer_7_DDOS.pdf`.

[87] Ping Wang, S. Sparks, and C.C. Zou. An advanced hybrid peer-to-peer botnet. *Dependable and Secure Computing, IEEE Transactions on*, 7(2):113–127, April 2010.

[88] Wikipédia. 2007 cyberattacks on estonia. [Citované 2015-5-16] Dostupné z `http://en.wikipedia.org/wiki/2007_cyberattacks_on_Estonia`.

[89] Wikipédia. Denial-of-service attack. [Citované 2015-5-1] Dostupné z `http://en.wikipedia.org/wiki/Denial-of-service_attack`.

[90] Wikipédia. Sociální inženýrství (bezpečnost). [Citované 2015-5-5] Dostupné z `http://cs.wikipedia.org/wiki/Soci%C3%A1ln%C3%AD_in%C5%BEen%C3%BDrstv%C3%AD_(bezpe%C4%8Dnost)`.

[91] Wikipédia. Indicator of compromise, 2015. [Citované 2015-5-30] Dostupné z `http://en.wikipedia.org/wiki/Indicator_of_compromise`.

[92] Guanhua Yan, Songqing Chen, and Stephan Eidenbenz. Ratbot: anti-enumeration peer-to-peer botnets. In *Information Security*. 2011.

[93] Guanhua Yan, Duc T Ha, and Stephan Eidenbenz. Antbot: Anti-pollution peer-to-peer botnets. *Computer Networks*, 55(8):1941–1956, 2011.

[94] Ilsun You and Kangbin Yim. Malware obfuscation techniques: A brief survey. In *BWCCA*, pages 297–300, 2010.

[95] Lenny Zeltser. How antivirus software works: Virus detection techniques. [Citované 2015-5-16] Dostupné z `http://searchsecurity.techtarget.com/tip/How-antivirus-software-works-Virus-detection-techniques`.