Antti Heinonen

Yleiskatsaus Botnetteihin ja C&C-liikenteeseen

Tietotekniikan kandidaatintutkielma

2. toukokuuta 2016

Jyväskylän yliopisto

Tietotekniikan laitos

Tekijä: Antti Heinonen

Yhteystiedot: anvenehe@student.jyu.fi

Työn nimi: Yleiskatsaus Botnetteihin ja C&C-liikenteeseen

Title in English: Overview of Botnets and C&C-structures

Työ: Kandidaatintutkielma

Sivumäärä: 38+0

Tiivistelmä: Tässä kirjallisuuskatsauksessa luodaan yleiskatsaus botnetteihin ja niihin liittyviin ilmiöihin. Johdannossa perustellaan aiheen ajankohtaisuus. Luvussa 2 esitellään botnet pintapuolisena käsitteenä. Luvussa 3 esitellään tapoja luokitella botnettejä. Luvussa 4 esitetään botnettien hyödyntämiä tapoja piilottaa omaa viestiliikennettään ja luvussa 5 esitellään tapoja havaita kyseistä liikennettä. Lopuksi luvussa 6 suoritetaan yhteenveto.

Avainsanat: Botnet, C&C, IDS, DGA, DNS

Abstract: This literature review focuses on providing an overview on botnets and phenomenoms related to botnets. The first chapter argues for the relevancy of the topic. The second chapter presents the basic structure of a botnet. The third chapter futher expands on the subject and presents different ways to categorize botnets. The fourth chapter focuses on the means which botnets use to hide their communication methods, and the fifth chapter presents some ways to detect the traffic caused by this communication. The sixth chapter presents the conclusions of the literature review.

Keywords: Botnet, C&C, IDS, DGA, DNS

Kuviot

Sisältö

1 Johdanto

Yksi vakavimmista erinäisten organisaatioiden nykypäivänä kohtaamista uhista aiheutuu pahansuovista boteista. Pahansuovat botit työskentelevät yhdessä luodakseen botnetin. Botnet on verkosto tartunnan saaneita laitteita, jotka toimivat yhtenä yksikkönä. Tällä yksiköllä voidaan esimerkiksi laukaista massiivisia koordinoituja hyökkäyksiä. Botnetit koostuvat useista tuhansista tartunnan saaneista laitteista, jotka saavat käskyjä botmasterilta eli verkostoa hallitsevalta ja ohjaavalta ihmitaholta Command and Control (C&C) palvelimien kautta. (Graham & Winckles (2014))

Rikolliset hyödyntävät botnettejä monin eri tavoin: esimerkiksi roskapostin lähettämisessä, hajautetuissa palvelunestohyökkäyksissä, bitcoinien louhimisessa, keyloggereiden asentamisessa ja niin edelleen. Ilmiö on jopa levinnyt niin pitkälle että infektoituneita laitteita eli yksittäisiä botteja tai jopa kokonaisia botnettejä myydään deep webissä. Kyseessä on siis erittäin ajankohtainen ja olennainen aihe.

Koska botnetit ja niiden havaitseminen ovat jatkuvassa kehityksessä digitalisaation edetessä ja teknologian kehittyessä, on tähän kirjallisuuskatsaukseen sisällytetty pääasiassa vain viimeisen viiden vuoden aikana julkaistua materiaalia.

Itse botmasterin henkilöllisyyden piilottaminen on tärkeää, mutta useimmiten kolmannen osapuolen VPN:t, proxyt ja SSH-tunnelointi riittävät tämän toteuttamiseksi. Nämä ovat yleisimmät käytetyt väliasemat botmasterin henkilöllisyyden piilottamiseksi. (Lin & Wee (2012))

FBI:n vuoden 2010 internet rikollisuuden raportissa 303809 valituksesta ja 1420 valmistellusta rikosoikeudenkäynnistä 6 johtivat tuomioon. Tämän perusteella kyberrikollisten pääasiallinen huolenaihe ei liene oman henkilöllisyytensä piilottaminen, vaan oman botnettinsä tuottaman liikenteen salaaminen. Tämän takia henkilöllisyyden piilottamista käsittelevä osa-alue ohitetaan tutkielmassa ja sen sijasta esitellään C&C-liikenteen piilottamismetodeita. Ideaalitilanteessa botnetillä olisi mahdollisimman pitkä elinikä ja suuri uhrijärjestelmämäärä.

Luvussa 2 käsitellään yleiseltä näkökulmalta mikä on botnet ja niiden yleiset toimintaperiaatteet. Luku 3 taas rakentuu luvun 2 pohjalle lisäten esimerkkejä eri toteutuksia omaavista botneteistä ja niiden luokittelusta. Luvussa 4 käydään läpi metodeja, joita botnetit hyödyntävät oman liikenteensä piilottamiseksi. Lopulta luku 5 esittelee pääasialliset tavat havaita kyseistä botnet liikennettä.

Tutkimuksen päämääränä on tuottaa yleiskatsaus siitä, mikä on botnet ja minkälaisia erilaisia botnettien alalajeja on. Tämän lisäksi tutkimuksessa käsitellään miten botnetit piilottavat omaa C&C-liikennettään sekä lyhyesti kuinka botnettejä havaitaan, sillä niiden aiheuttaman uhan vuoksi niiden havaitseminen on olennaista. Kuten Gramam, Winckles & Moore (2014) toteavat, ”Yksittäisten botnettiin kuuluvien bottien tarvitsee kommunikoida oman C&C-palvelimensa kanssa saadakseen ohjeita, päivityksiä tai hyökkäskäskyjä. Tätä pilven tai verkon halki kulkevaa liikennettä voidaan käyttää osoittamaan botnet toiminnan läsnäoloa.”

2 Botnet

Botnetit ovat jo yli 20 vuotta vanha ilmiö, sillä ensimmäinen havaittu botnet oli Eggdrop vuonna 1993. (Li, Jiang & Zou (2009)) Aluksi botnetit ja botit olivat laillisiin tarpeisiin käytettäviä työkaluja, kuten IRC-kanavan ylläpitämiseen, silloin kun yhtäkään käyttäjää ei ollut yhteydessä kanavaan.

Ajan myötä ja tekniikan kehittyessä botnetit ovat muuttuneet kyberrikollisuuden työkaluiksi, sillä botnettejä voi suunnitella toteuttamaan lukuisia erilaisia operaatioita. Tästä syystä niitä kutsutaankin internetin maanalaisen ekonomian linkkuveitsiksi. Botnettien käyttö ei vaadi korkeita teknisiä taitoja, sillä pahansuopa käyttäjä voi vuokrata botnet-palveluja kyberrikolliselta. (Hung-Chang & Guo-Quan (2011))

2000-luvun aikana botneteistä on kasvanut yksi suurimmista kyberturvallisuuden uhista. Kuten Nagaraja, Houmasadr, Piyawongwisal, Sing, Agarwhal & Borisov (2011) asian ilmaisevat: ”Botnettien evoluutio on pääosin pohjautunut yksinkertaisen kunhan toimii-periaatteen pohjalle.” Tämän vuoksi botnetin esiintymismuodot ovat lukuisia ja vuosien varrella ne ovat kehittyneet hyvin monipuolisiksi sovelluksiksi, joita voi käyttää lukuisiin eri tarpeisiin. Tästä huolimatta idea jokaisen botnetin takana pysyy samana.

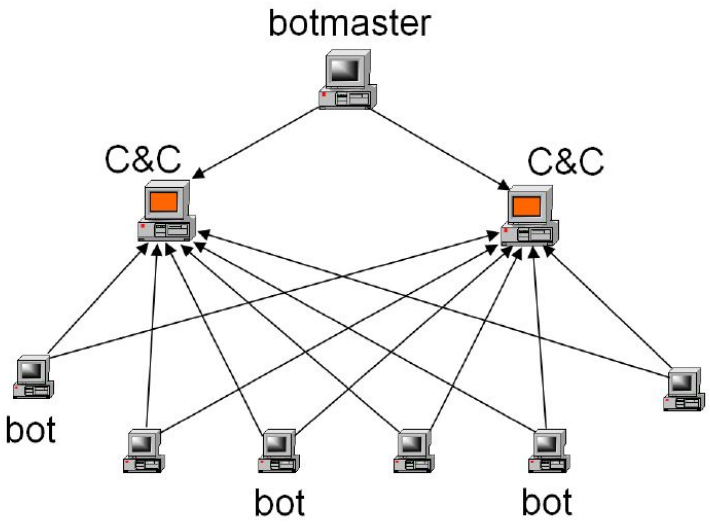
Jokainen botnet rakentuu kolmesta pääasiallisesta osasta:

1. käskyjä jakava kone
2. C&C-palvelin
3. käskyjä vastaanottava ja toteuttava kone

Nämä kolme osaa muodostavat botnet-ohjelmiston perusrakenteen, jonka jokainen botnet omaa, vaikka rakenteen toteutus vaihtelee. Kuten (Schiller& Binkley 2011, s. 30) huomauttavat: ”On kaksi vaatimusta joiden tarvitsee täyttyä, jotta jokin haittaohjelma pystytään kategorisoimaan botnetiksi. Ensinnäkin asiakasohjelmien täytyy pystyä suorittamaan toimintoja ilman että botmasterin tarvitsee kirjautua asiakkaan käyttöjärjestelmään. Toiseksi kaikkien verkostoon kuuluvien laitteiden pitää

pystyä toimimaan koordinoidusti ja yhtenäisesti jonkin tavoitteen saavuttamiseksi.”

Varsinainen ero botnetin ja tavanomaisen haittaohjelmiston välillä on siis Command and Control-rakenteen (C&C) olemassaolo. (Vaniya, Meniya & Jethva (2013)) Useimmiten tätä rakennetta kutsutaan C&C-kanavaksi, sillä se toimii botnet-järjestelmän sisäisenä kommunikaatioväylänä. C&C-kanavan avulla botmaster välittää käskyjä botnettiin kuuluville järjestelmille ja järjestelmät vastaavat botmasterille. Näitä käskyjä voivat olla esimerkiksi hajautetun palvelunestohyökkäyksen (DDOS) suorittaminen, keyloggerin käynnistäminen tai sen keräämien tietojen palauttaminen botmasterille, tietyn verkostoon kuuluvan solmun etsiminen, tilan raportointi, valmiustilaan siirtyminen ja niin edelleen riippuen siitä, miten botnet on toteutettu ja mikä on sen päämäärä. Näiden viestien avulla botmaster hallinnoi omaa bottiverkostoaan ja ohjaa verkoston toimintaa saavuttaakseen oman päämääränsä. Kuvio 1 näyttää C&C-viestinnän periaatteen.



Kuvio 1. Kuva C&C toiminnasta

2.1 Botnettien elinkaari

Yksittäisen tietokoneen päätyminen botnetin osaksi alkaa, kun järjestelmä saa tartunnan. Tartuntametodeina hyödynnetään muun muuassa sähköpostia, tietokoneoh-

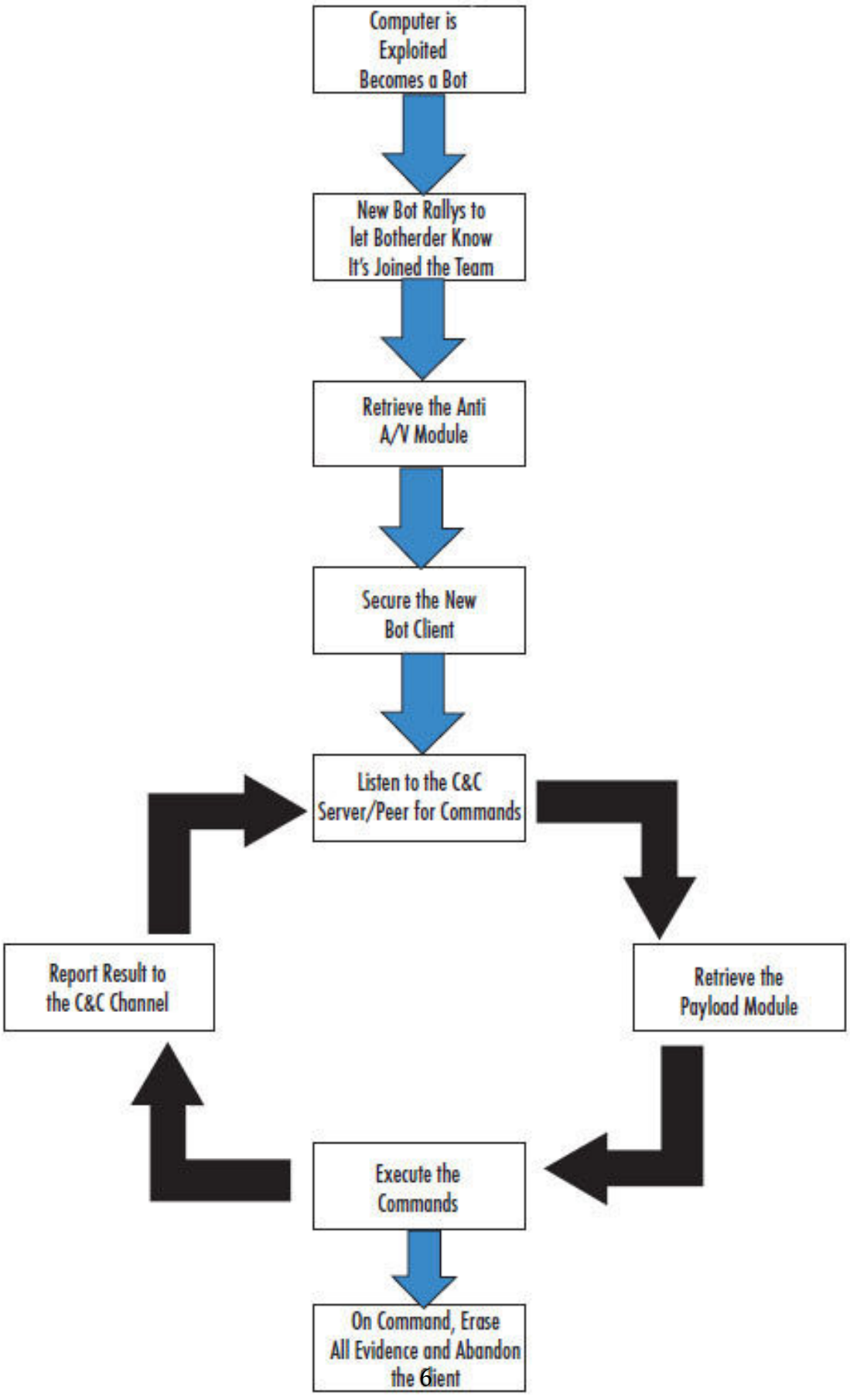
jelmiston tietoturva-aukkoja, pikaviestintää, P2P-tiedostonjakamisverkostoa tai muita botnettejä. (Mustapha, Granadillo & Debar (2011))

Tartunnassa järjestelmään siirtynyt tiedosto voi sisältää joko koko botin pohjakoodin tai komentosarjan. Jälkimmäisessä tapauksessa järjestelmä toteuttaa kyseisen komentosarjan, jonka ohjeistuksella järjestelmä lataa varsinaisen lähdekoodin jostakin ennalta määritellystä sijainnista ja asentaa sen järjestelmään. Joka tapauksessa lähdekoodin latauksen jälkeen uhrijärjestelmä asentaa botin järjestelmään.

Tartunnan jälkeen botti ottaa yhteyden omaan C&C-palvelimeensa ja pyytää ohjeistuksia. (Karim, Salleh, Shiraz, Shah, Awan, Anuar (2014)) Näiden ohjeistuksien tarkoitus on toimittaa botille sen tämänhetkiset voimassaolevat käskyt.

Tätä seuraa varsinainen käskyn toteuttaminen. Käskyt voivat olla jotain aiemmin mainituista prosesseista. Lopulta botnet ylläpitää itseään tai lataa päivityksiä ohjeistuksien mukaan. Viimeinen osa botnetin elinkaarta on, kun se käskystä poistaa itsensä ja kaikki mahdolliset todisteet asiakasjärjestelmästä. (Schiller& Binkley 2011, s. 36) Kuvio 2 esittää botnetin elinkaaren.

Botnetit säännöllisesti rekrytoivat verkostoonsa uusia zombikoneita, aiemmin mainittujen tartuntametodien mukaisesti. Muiden haittaohjelmien tavoin botnetit pyrkivät leviämään mahdollisimman useaan järjestelmään ja laajentamaan vaikutusvaltaansa. Itseasiassa botnetit ovat jo niin yleinen ilmiö, että botnetit joihin kuuluu vain satoja tai muutamia tuhansia tietokoneita lasketaan pieniksi botneteiksi. (Schiller& Binkley 2011, s. 30) Tämän lisäksi Rodriguez-Gomez, Macia-Fernandez & Garcia-Teodoro (2011) huomauttavat, että arvion mukaan vuonna 2007 noin 100–150 miljoonaa 600 miljoonasta internetin isännästä olivat osana botnettiä.



Kuvio 2. Botnettien elinkaari

3 Botnettien alakategoriat

Botnettien laajan kehityksen seurauksena nykyään on olemassa lukemattomia erilaisia botnettejä. Näitä botnettejä voidaan luokitella erilaisiin alakategorioihin. Kuten Haddadi, Cong, Porter & Zincir-Heywood (2011) asian ilmaisevat, ”Koska botnetit ovat elinkaariensa aikana alkaneet käyttämään erilaisia protokollia, topologioita ja tekniikoita välttääkseen havaitsemista, luonnollisesti on alkanut kilpavarustelu botnettien ja havaitsemisjärjestelmien välillä.” Eli havainnointijärjestelmien kehittyessä myös botnetit ovat kehittyneet lukuisin eri tavoin.

3.1 Botnettien luokittelu

Botnettien luokittelu tapahtuu muutamilla eri tavoilla. Ensimmäisenä luokittelutapana on ohjeistuksien saaminen: PUSH tai PULL. (Mustapha ym. (2011)) PUSH-mallissa botmaster työntää käskyn boteille ja nämä toteuttavat kyseisen käskyn. PULL-mallissa botmasterin käskyt sijaitsevat esimerkiksi verkkopalvelimella, josta botit hakevat uusia käskyjä säännöllisin väliajoin.

Toisena luokittelutapana on botnetin arkkitehtuuri: keskitetty tai hajautettu. (Karim ym. (2014)) Keskitetyssä mallissa botmaster valitsee yhden tai useamman pisteen komentojensa levittäjäksi, miltä käsin hän levittää viestin lopuille botnettiin kuuluvista solmuista. Tämä piste voi esimerkiksi olla yksi tai useampi botti koko verkostossa tai tietty verkkosivu, joka toimii C&C-palvelimena.

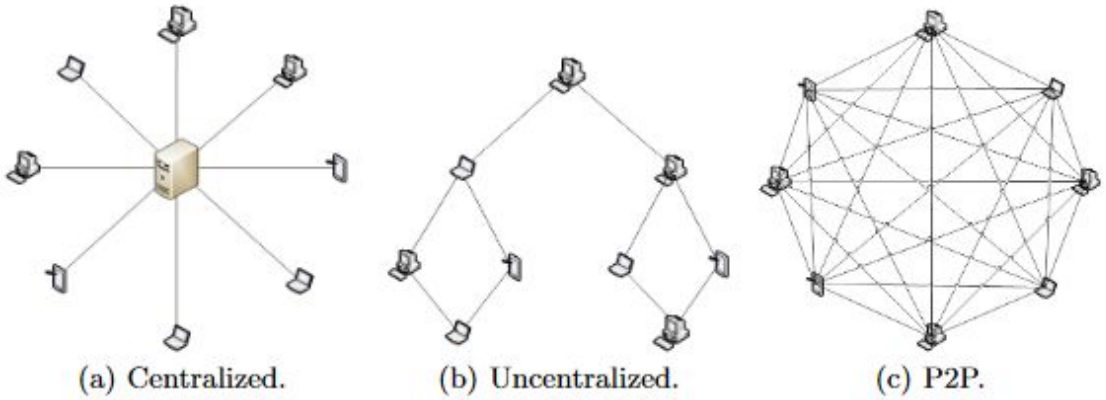
Suurin hyöty keskitetyssä arkkitehtuurissa on pieni viestilatenssi, joka tekee käskyjen jakamisesta helppoa. Keskitetyssä arkkitehtuurissa botmasterin on myös helppo ylläpitää tietoa siitä, kuinka monta järjestelmää botnettin kuuluu. Ongelmaksi kuitenkin nousee se, että tämä C&C -palvelin on koko botnetin heikoin kohta. Jos C&C-palvelin havaitaan ja sen toiminta estetään koko botnet on arvoton ja toimimaton. (Houmansadra & Borisov (2013)) Tämän lisäksi C&C-palvelimen havaitseminen verkkoliikenteessä on helpompaa keskitetyssä mallissa verrattuna hajautettuun malliin, jota käsitellään seuraavassa kappaleessa, sillä kaikki botit ottavat yhteyden

samaan pisteeseen. (Mustapha ym. (2011)) Useimmiten IRC- ja HTTP-protokollaa hyödyntävät botnetit toimivat tämän kommunikaatiomallin edustajia.

Hajautetussa mallissa ei ole yhtä keskitettyä C&C-palvelinta, vaan viestit välitetään P2P-mallilla (Peer to peer). Botmaster välittää viestinsä eteenpäin yhdelle tai useammalle botille, jotka välittävät viestin eteenpäin toisille boteille. Botnetin hajautuneisuuden määrä vaihtelee riippuen sen totetuksesta, kuten esimerkiksi kuviosta 3 huomaa. Useissa hajautetuissa botneteissä on kuitenkin jonkin tyyppinen C&C-mekanismi.

C&C-mekanismi voi olla esimerkiksi seed-lista, joka sisältää yhteystietoja verkostoon kuuluvista solmuista joilta käskyjä voi ladata. Vastasyntynyt botti ottaa yhteyden tähän seed-listaan liittyessään botnet-verkostoon ja pyrkii lataamaan joltain listaan kuuluvalta solmulta voimassaolevat käskyt. Usein tässä mallissa jokainen botti voi myös toimia joko palvelimena tai asiakkaana. (Scanlon & Kechadi (2012))

Koska P2P-kommunikaatiometodi ei nojaa kokonaan vain muutamaan palvelimeen, yhden tai useamman botin tuhoaminen ei välttämättä johda botnetin kaatumiseen. Suurimpana heikkoutena hajautetussa arkkitehtuurissa on P2P-järjestelmien kehittämisen haastavuus. Hajautetun järjestelmän kehittäminen on huomattavasti haastavampaa kuin keskitetyn järjestelmän, eikä siinä ei ole takuita latenssista tai viestin perille saapumisesta. (Mustapha ym. (2011))



Kuvio 3. Botnettien pääasialliset topologiat.

Kolmas luokittelutapa on kommunikaatiometodi eli botnettien tapa toteuttaa yhteydenpito oman C&C-palvelimensa kanssa. (Paxton, Jang, Russel, Ahn & Moskowitz (2015)) Seenivasan & Shanti (2014) jakavat C&C-viestinnän perusteella botnetit viiteen pääasialliseen eri kategoriaan:

1. IRC-botnet
2. HTTP-botnet
3. P2P-botnet
4. Mobile botnet
    4.1. SMS pohjainen
    4.2. Bluetooth pohjainen
5. Pilvibotnet

Luokittelu tosin vaihtelee hiukan lähteestä riippuen. Toisinaan listaan lisätään vielä esimerkiksi IM-pohjaiset botnetit (Instant Messaging). Tarkemmassa kategorisoinnissa otetaan huomioon kaikki edellämainitut tekijät: esimerkiksi HTTP-botnet omaa PULL-mallin käskyjenjaon ja sen voi toteuttaa joko keskitetyllä arkkitehtuurilla tai hajautetulla arkkitehtuurilla. (Shanti & Seenivasan (2015))

3.2 IRC-pohjaiset botnetit

IRC-pohjaiset botnetit (Internet relay chat) ovat vanhin havaittu botnettien muoto. Iästään huolimatta tämä on vieläkin yksi yleisimmistä botnettien muodoista. (Zhu & Lee (2015)) IRC-pohjaisessa botnetissä C&C-palvelimen viestit ja yleinen tarvittava kommunikaatio toteutetaan IRC-kanavan kautta. Kun botti on ladannut tarvittavat pohjakooditiedostot, se liittyy ennalta määriteltyyn IRC-kanavaan. Tältä kanavalta botti saa käskynsä joko privaattiviestillä tai lukemalla kanavan aiheen. Tämä tietenkin riippuu botnetin toteutuksesta, vaikka kanavan aihe toimii useimmiten pääasiallisena käskynvälittäjänä.

Käskyjen saamisen jälkeen botti ylläpitää IRC-yhteyttään, joten IRC-pohjainen botnet seuraa PUSH-mallin ohjeistusta. Botmaster työntää viestin IRC-kanavalle, josta

botit lukevat uudet ohjeistuksensa. (Eslahi, Hashim & Tahir (2013)) Monet tutkimukset ovat osoittaneet, että nykypäivänä useat olemassaolevat botnetit käyttävät IRC-protokollaa sen yksinkertaisen mutta toimivan rakenteen vuoksi. (Houmansadra & Borisov (2013))

IRC-pohjaiset botnetit on melko helppo huomata ja niiden liikenteen pysäyttämiseksi voi esimerkiksi vain sulkea IRC-protokollan käyttämät portit palomuurista, milloin C&C-palvelimen ja yksittäisen järjestelmän välinen kommunikaatio katkeaa. (Cai & Zou (2012)) Muutamia tunnettuja IRC-pohjaisia botnettejä ovat Agobot, SDBot, Spybot ja GT Bot.

3.3 HTTP-pohjaiset botnetit

Useimmiten HTTP-botnettien C&C-palvelimena toimii PHP-pohjainen verkkopalvelin. HTTP-botnettien toiminta perustuu TCP-paketteihin (Transfer Control Protocol) upotettuihin viesteihin. HTTP-botnettien tarkoituksena on suojautua havaitsemiselta naamioitumalla muun verkkoliikenteen sekaan. (Alomari, Manickam, Gupta, Singh & Anbar (2014))

Tartunnan saaneet botit ottavat ennaltamääritellyin väliajoin yhteyden omaan C&C-palvelimeensa ja odottavat vastausta. C&C-palvelimen lähettämä ohjeistus sijaitsee HTTP GET-pyynnön vastauksessa. Saatuaan vastauksen botit toteuttavat käskyn, minkä jälkeen ne toistavat samaa prosessia hakeakseen lisäohjeistuksia tai päivityksiä. Botmaster vaihtaa käskyjen sisältöä muuttamalla palvelimella sijaitsevan HTTP-sivun sisältöä. (Cai & Zou (2012)) Perusidea on seuraavanlainen:

```
while(1) {
    get_and_process_cc_commands();
    sleep(60);
}
void get_and_process_cc_commands(){
   //koodi tietojen hakemiseen ja toteuttamiseen
}
```

Tämä on PULL-mallian ohjeistustapa, sillä botit pitävät itse huolta yhteyden ylläpidosta. Cai & Zou (2012) esittelevät kolme suurinta etua HTTP-pohjaisessa botnetissä:

1. HTTP C&C käyttää C-S mallia (client to server), joten se on helppo rakentaa verrattuna P2P-botnettin.
2. HTTP on niin yleisesti käytetty protokolla, että C&C-liikenne häviää tavanomaisen verkkoliikenteen sekaan.
3. HTTP on hyvin yleinen protokolla, joten palomuurit harvoin estävät HTTP-protokollaa. Tämä tarjoaa HTTP-pohjaisille botneteille joustavamman toimintaympäristön.

HTTP-botnettien hyvien puolien vuoksi botmasterit ovat pyrkineet kehittämään lukuisia piilotettuja tapoja välittää käskyjä botnetille. Käsky saattaa sijata esimerkiksi ladattavassa kuva- tai musiikkitiedostossa, mitä ei voi nopealla katsauksella erottaa tavanomaisesta liikenteestä. Tämä vaikeuttaa botnetin havaitsemista.

Muutamia kuuluisia HTTP-pohjaisia botnettejä ovat BlackEnergy ja Zeus. Zeusta pidetään eräänä kaikista laajimmalle levinneistä botneteistä ja sen muunnelmat ovat vieläkin aktiivisia. Zeuksen uskotaan levinneen vuonna 2010 jopa 3,6 miljoonaan tietokoneeseen ainoastaan Yhdysvaltojen alueella. (Cai & Zou (2012)) Zeuksen suuri suosio johtuu osittain siitä, että kyse ei enää ole yksittäisestä botnetistä vaan työkalupakista, jolla voi rakentaa oman botnettinsä. Vaikka Zeus alunperin huomattiin vuonna 2006, on sen suosio ja variaatioiden määrä kasvanut hyvin paljon 2010-luvulla. Tämä johtuu hyvin paljon siitä, että vuonna 2011 sen lähdekoodi vuosi githubiin. Zeuksen kokonaiset taloudelliset kustannukset on arvioitu yli 100 miljoonaan. (Rickardi, Di Pietro & Vila (2011))

3.4 P2P-pohjainen botnet

P2P on hajautettu tapa toteuttaa botnettejä. Silva, Silva, Pinto & Salles (2013) lajittelevat P2P-botnetit kolmeen kategoriaan: epämuodollinen P2P-kerrotus (unstructured overlay), muodollinen P2P-kerrostus (structured overlay) ja superpeer kerros-

tus.

Epämuodollinen kerrostus referoi satunnaisiin topologioihin, joissa on vaihteleva määrä hajautumista. Epämuodolliset verkostot tukevat floodingia, satunnaiskulkua ja variaatioita näistä kahdesta. (Silva ym. (2013)) Epämuodollinen kerrostus on kaikista hajautunein versio P2P-botneteistä.

Muodollisessa P2P-kerrostuksessa botit sisältävät esilataus-mekanismin (bootstrap). Aktivoiduttuaan botti esilataa DHT-tiedoston (Distributed hash table), joka sisältää sekä avaimen että arvon. Tämän jälkeen tämä verkostoon juuri yhdistynyt botti saa voimassaolevan käskyt toiselta botilta DHT-järjestelmässä, minkä jälkeen jokainen botti ylläpitää omaa DHT-tiedostoaan. Samankaltaisia metodeja hyödyntää esimerkiksi BitTorrent DHT ylläpitäessään aktiivisia solmuja. (Silva ym. (2013).

Superpeer kerrostuksessa kaikki botit eivät ole tasa-arvoisia. Pieni osa boteista valitaan automaattisesti väliaikaisiksi palvelimiksi. Esimerkiksi Skype hyödyntää tämänkaltaista rakennetta. Superpeer-verkostot ovat näkyvämpiä ja haavoittuvaisempia kohdistetuille hyökkäyksille verrattuna muunlaisiin P2P-muodostelmiin, joten voidaan olettaa etteivät tehokkaimmat P2P-botnetit käytä superpeer kerrostusta. (Silva ym. (2013)) Palvelimina toimivat botit valitaan usein latenssin, järjestelmän tehokkuuden ja etäisyyden perusteella.

Rakenteensa perusteella jokaisella botilla on rajallinen näkymä aktiivisesta verkostosta. Tämä johtuu esimerkiksi suurimman määritellyn osoitemäärän, osoitteenmuutoksen tai palomuurin vaikutuksista. (Scanlon & Kechadi (2012)) Esimerkiksi Phatbotin muunnelma käytti kryptattua P2P-protokollaa, joka oli suunniteltu yksityiseen viestimiseen ja tiedostojen vaihtamiseen pienten luotettavien ihmisryhmien välillä. (Mustapha ym. (2011))

Tunnetuimmat ja tutkituimmat P2P-botnetit ovat Nugache, Storm, Waledac ja Confiker. (Zhang, Perdisci, Lee, Sarfraz & Luo (2011)) P2P-botnettien etuja ovat niiden havaitsemisen vaikeus sekä pääasiallisten C&C-palvelimien puuttuminen, mutta tämänkaltainen toteutus omaa myös kaikki hajautettua arkkitehtuuria hyödyntävien botnettien haittapuolet. Näitä haittapuolia ovat esimerkiksi takeettomuus viestien

perille saapumisesta ja korkea viestilatenssi.

3.5 Mobile Botnet

Mobile botnetit ovat varsin uusi ilmiö. Vaikka vielä ei ole havaittu botnettien suurta leviämistä mobiililaitteiden maailmassa, mobiililaitteiden määrän jatkuva kasvu osoittaa että tämä lienee vain ajan kysymys. (Zeng, Shin & Hu (2012))

Myös älypuhelinpohjaisten pankkipalvelujen suosio on kasvanut, vaikka niiden tietoturvaominaisuudet eivät ole verrattavissa tietokoneiden vastaaviin ominaisuuksiin. (Zeng ym. (2012)) Useimmiten Mobile botnettien C&C-viestien välittämiseen käytetään SMS-palveluja (short message service). Tämä johtuu kahdesta syystä: kaikki puhelimet omaavat SMS-palveluja, ja on mahdollista välittää SMS-viestejä internetin avulla. Internetin kautta tapahtuva SMS-viestintä halventaa kustannuksia. Tämä on olennaista, sillä liian suuret puhelinlaskut saattavat johtaa siihen, että uhripuhelimen omistaja tutkii tarkemmin oman puhelimensa toimintaa. Tällöin puhelimen omistaja saattaa huomata puhelimellaan sijaitsevan botin. Ohjeistukset usein myös naamioidaan roskapostiksi. Kuvio 4 havainnoillistaa mahdollista viestintätapaa SMS-palvelujen kautta.

Kuvio 4. Kuva mobile botnetin C&C-viestinnästä SMS-palvelujen avulla.

Kuvion 4 ensimmäinen puolisko havainnollistaa kuinka SMS-viestiin on piilotettu base64-koodauksella yksi viesti, joka on jaettu kahdeksi osaksi: ensimmäinen

osa virheviestiksi ja toinen koodiksi. Tämän tarkoituksena on saada viesti muistuttamaan roskapostia. Koodauksen purkamisen jälkeen viestin sisällöksi paljastuu 7912034218110523_7347096452_12345678. Tässä ensimmäinen osuus on etsittävän botin tunniste, toinen sen botin puhelinnumero jolle löydetty tulos välitetään ja lopuksi find node-operaation tunnuskoodi. (Zeng ym. (2012))

Hyökkääjät voivat myös hyödyntää SMS-palveluja lähettääkseen SMS-roskapostia, maksullisia tekstiviestejä tai aiheuttaakseen palvelunestohyökkäyksiä ilman puhelimen omistajan lupaa. (Alzahrani & Ghorbani (2014))

Pääasiallinen mobile botnettien hyöty tulee kuitenkin yksityisen tiedon varastamisesta, sillä botti voi nopeasti tutkia isäntälaitteen sisältämän informaation. Tämä informaatio voi esimerkiksi olla lista eri palvelujen käyttäjänimistä ja salasanoista, lista lähetetyistä tekstiviesteistä tai puhelimessa sijaitseva osoitelista. Useimmiten mobile botnetit on rakennettu puu-mallilla ja niiden havaitseminen on hyvin vaikeaa esimerkiksi IP-osoitteen puuttumisen vuoksi. (Geng, Xi, Zhang, Guo, Yang & Wei (2012))

Vähemmän yleinen versio mobiili botnettien C&C-kanavasta on bluetooth C&C. Syntymästään lähtien bluetooth on kärsinyt lukuisista turvallisuusongelmista. Näitä turvallisuusongelmia on yritetty korjata lisäämällä ominaisuuksia kuten autorisaatio, autentikaatio ja kryptaus. Vaikka bluetoothin turvallisuus on parantunut, bluetooth-teknologiaa hyödynnetään vieläkin pahansuovissa aktiviteeteissa. (Pieterse & Olivier (2014))

Botmasterit pyrkivät automatisoimaan bluetoothin naapuriin yhdistymisprosessin, jotta käyttäjällä ei olisi mitään tekemistä tämän prosessin kanssa. Kun autorisaatio ja autentikaatio on automatisoitu, datansiirto voi alkaa heti toisen laitteen saapuessa laitteen kantamalle.

Botmasterin näkökulmasta bluetooth C&C-mekanismin etuna on, että sen haavoittuvuus mahdollistaa vaivihkaisen C&C-kanavan, joka voi helposti välttää yleisimmät havaitsemismetodit. Muita etuja tässä C&C-kanavassa on bluetoothin yleisyys; bluetooth on saatavilla suurimmassa osassa mobiililaitteista ja se tarjoaa nopean ja

ilmaisen kommunikaatiotavan verrattuna SMS-palveluihin tai internettiin. (Seenivasan & Shanti (2014)) Ongelmaksi on muodostunut bluetoothin lyhyt kantavuus ja muuttuva topologia sekä bluetoothin aiheuttama akunkulutus. Esimerkiksi Zeng ym. (2012) hyödynsivät bluetoothia toissijaisena varakommunikaatiokanavana SMS-palvelujen lisänä.

3.6 Pilvibotnet

Pilvibotnet hyödyntää jatkuvassa kasvussa olevaa virtualisaatiota ja pilvipalveluiden yleistymistä. Kuten Graham, Winckles & Sanchez-Velazquez (2015) sanovat, ”Pilvipalvelut tarjoavat ideaalin alustan botnettien isännöintiin, koska

A) Pilvipalvelut tarjoavat halvan pääsyn CPU-resursseihin.
B) Virtuaalikoneet (VMs) joita isännöidään pilvessä voidaan kloonata helposti ja nopeasti suurikokoisen botnetin luomiseksi.
C) Useita virtuaalikoneita pilvessä on suhteellisen helppo hallinoida.”

Usein pilvibotneteissä botmaster asettaa yhden virtuaalikoneen C&C-palvelimeksi ja toisen datavarastoksi. (Lin & Wee (2012)) Tässä botnet muodossa botmaster voi helposti laukaista uusia C&C-palvelimia ja väliasemia pilvessä, dynaamisesti siirtää niitä uusiin verkkotunnuksiin (domain) ja lopulta poistaa virtuaaliset kuvat sekä muut jättämänsä jälkensä.

Esimerkiksi vuonna 2009 tutkijat Arbor Networksista huomasivat, että botnetti lähetti kyselyitä google_appengine-palvelussa sijaitsevaan sovellukseen. Tutkimalla havaintoa tarkemmin he löysivät haittaohjelman google_appengine-palvelusta, mitä käytettiin ohjeistuksien välittämiseen boteille. (Studer (2011)) Vuonna 2014 Dropboxia käytettiin C&C-palvelimena PlugX RAT:ille. (Graham ym. (2015)) Pilvipalveluiden arkipäivistyessä pilvibotnetit tulevat vielä yleistymään entistä enemmän ja kasvattamaan suosiotaan, vaikka niitä ei ole vielä havaittu samoissa mittakaavoissa kuin esimerkiksi vanhempia HTTP-pohjaisia botnettejä.

3.7 Botnettien muita esiintymismuotoja

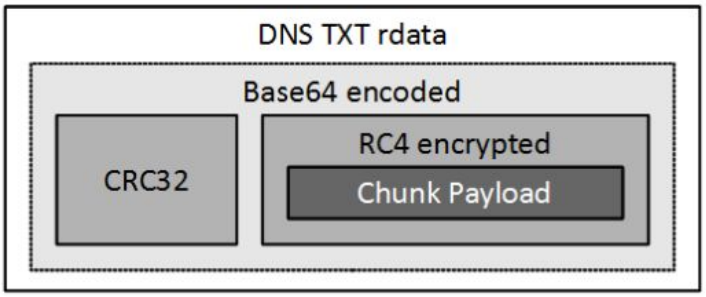
Botmastereiden on havaittu myös kehittäneen pikaviestintään (IM, Instant messaging) perustuvia botnettejä, jotka ovat hyödyntäneet esimerkiksi AIM-, Yahoo!-, ICQ- ja MSN-palvelujen tarjoamia viestipalveluja C&C-kanavinaan. Toiminnaltaan IM-botnetit muistuttavat IRC-botnettejä. Haittapuolena tässä metodissa on, että IM-palveluntarjoajat tarkkailevat omaa liikennettään. Tämän takia kyseiset C&C-kanavat usein havaitaan nopeasti ja IM-botnettien elinikä jää varsin lyhyeksi. Tästä syystä IM-pohjaiset botnetit eivät ole yhtä yleisiä kuin esimerkiksi IRC- ja HTTP-pohjaiset botnetit. (Schiller& Binkley 2011, s. 86–87)

Botnetit eivät aina käytä omia palvelimiaan C&C-viestien välittämiseen. Botnettien on havaittu esimerkiksi toteuttavan PULL-mallin komentorakenne internetin tarjoamien kolmannen osapuolen musiikinjakosivustojen avulla. Tällöin yksittäinen botti etsii musiikinjakosivustolta jotakin tiettyä ennaltamääriteltyä tiedostoa. Tämä tiedosto on botmasterin ohjeistustiedosto, jonka hän on uudelleennimennyt musiikkitiedostoksi ja ladannut kyseiselle sivustolle. (Cai & Zou (2012)) Tämä on yksi lukuisista tavoista joilla botmasterit pyrkivät kiertämään omien C&C-palvelimiensa ylläpitämisen tuomia rajoitteita, kuten C&C-palvelimien paljastumista ja niiden liikenteen estämistä.

Uutta kehityskulkua edustavat myös niin sanotut piilotetun kanavan-botnetit (Covert channel). Tällaiset botnetit käyttävät kommunikaatiokanavanaan jotakin kanavaa, jota ei ole edes tarkoitettu kommunikaatioon. Tällaisia ovat esimerkiksi Feederbot ja Stegobot.

Vuonna 2011 Dietrich, Rossow, Freiling, Bos, van Steen & Pohlman (2011) tutkivat Feederbottia, joka käytti yksittäisiä DNS-kyselyitä kommunikaatiometodinaan. Feederbotin kommunikaatiotapana toimi siis DNS-tunnelointi eli ohjeistukset kuljetettiin rdata-kentän sisällä DNS-kyselyn vastauksen resurssiarkistossa (resource record). Koska DNS TXT-resurssiarkiston alkuperäinen tarkoitus on sisältää ihmisen luettavaa tekstiä, joka sisältää esimerkiksi yleistietoa etsittävästä sivustosta, Feederbotin viestisisältö oli koodattu base64-koodauksella standardilla aakkostolla kah-

den lainausmerkin sisällä. Kuvio 5 havainnoillistaa Feederbotin hyödyntämän DNS-tunneloinnin toimintaperiaatetta.



Kuvio 5. Feederbotin DNS-tunneloinnilla toimiva C&C-viestintä.

Dietrich ym. (2011) kuitenkin huomasivat, että Feederbotin suurin etu oli sen epätavallinen C&C-kommunikaatiotapa ja kun tämä oli selvitetty DNS-tunneloinniksi oli sen kommunikaatioliikenteen havaitseminen koneoppimisen ja tilastollisten menetelmien avulla melko helppoa.

Toinen esimerkki piilottettua kommunikaatiokanavaa hyödyntävästä botnetistä on Stegobot. Stegobot on Nagaraja ym. (2011) tutkimustarkoituksiin kehittämä botnetti, joka hyödynsi kommunikaatiokanavanaan sosiaalista mediaa ja steganografiaa.

Stegobotin pääasiallinen tarkoitus oli toimia keyloggerina ja varastaa yksityistä tietoa. Kun tartunnan saaneen järjestelmän käyttäjä latasi kuvan JPEG-formaatissa sosiaaliseen mediaan botti nappasi kuvan ennen sen latausta palveluun, minkä jälkeen se piilotti kuvaan steganografialla C&C-viestin ja latasi sen sosiaalisen median palvelimelle. Kun toisen tartunnan saaneen järjestelmän käyttäjä kirjautui sosiaaliseen mediaan, hänen laitteistossaan sijaitseva botti automaattisesti latasi tämän kuvan ja tulkitsi kuvan mukana kulkeneen viestin. Varastetun tiedon kuljettaminen ja yleinen viestiminen takaisin C&C-palvelimelle suoritettiin samalla tavalla.

4 Botmastereiden hyödyntämiä metodeja C&C-liikenteen salaamiseksi

Edistys haittaohjelmien tutkimuksessa on johtanut botmasterien yrityksiin parantaa botnettien kestävyyttä, sillä botnetin pidemmästä eliniästä on suurta hyötyä botmasterille. Näitä tapoja on lukuisia ja niiden toteutustavat vaihtelevat. Tässä luvussa käsitellään lyhyesti kryptaus, DGA-algoritmi (domain generation algorithm) ja DNS fast-flux-metodi.

4.1 Kryptaus

Suurin osa botneteistä hyödyntää kryptattuja C&C-viestejä välttääkseen paljastumista. Kryptattua C&C-liikennettä on paljon vaikeampi havaita tietosisällön perusteella verrattuna kryptaamattomaan liikenteeseen. (Rossow & Dietrich (2013))

Kryptausmenetelmiä on lukuisia, mutta useat botnetit hyödyntävät XOR-pohjaista kryptausta. Esimerkiksi Zeus P2P omaa seuraavanlaisen kryptausalgoritmin:

```
void zeus_encrypt(char *plain, char*  cipher, int len) {
   plain[0] = random(); // first byte in plaintext is random
  cipher[0] = plain[0]; // use random byte as init. vector (IV)
  for(int i = 1; i < len; i++) {
       cipher[i] = plain[i] ^ cipher[i-1];
   }
}
```

Käyttämällä ensimmäistä satunnaista bittiä avaimena kaikki seuraavat bitit on kryptattu XOR-menetelmällä edeltävän salakirjoitetun bitin kanssa. Kryptauksen tehokkuus ilmenee siitä, että kaksi muuten samanlaista Zeus P2P-viestiä saattavat omata jopa 2^{16} erilaista salakirjoitettua kirjoitusasua. (Rossow & Dietrich (2013))

XOR ei silti ole ainut käytössä oleva kryptaustapa. Esimerkiksi Nugachen kryptaus-

metodissa yhteyden alussa suoritettiin RSA-salausalgoritmin avaimenvaihto, minkä jälkeen botit vaihtoivat keskenään symmetriset Rijndael-256 sessioavaimet jokaiselle vertaisyhteydelle. Nugachessa myös keylogger-tiedostot, joihin näppäimistön painallukset olivat tallennettuina, kryptattiin Rijandael avaimella. Tämä avain oli derivoitu jostain vertaisyhteyden verkkoliikenteen spesifistä tiedosta. (Khattak, Ramay, Khan, Syed & Khayam (2013))

Myös MD5 (message-digest) on suhteellisen usein havaittu käytössä oleva algoritmi viestien sisällön naamioimiseksi. (Plohmann & Gerhards-Padilla (2012))(Andriesse, Rossow, Stone-Gross, Plohman & Bos (2013)) Kryptauksen edut ovat suuret ja jopa sen yksinkertainen toteutus vaikeuttaa botnetin havaitsemista, mistä syystä se on hyvin yleisessä käytössä.

4.2 Domain generation algorithm

Usein botnetit havaitaan niiden tuottamien lukuisten samankaltaisten DNS-kyselyiden perusteella, mitkä kohdistuvat samaan verkkotunnusnimeen (domain name). Tästä syystä tämän liikenteen jonkinlainen naamioiminen on hyvin edullista botmasterille.

Liikenteen piilottamisen saavuttamiseksi on luotu useita erilaisia DGA-algoritmeja, joita monet botnetit hyödyntävät niiden tarjoaman ylimääräisen turvakerroksen vuoksi. DGA-algoritmin idea on se, että koska IDS-järjestelmät (Intrusion detection system) keräävät muistiin tunnettuja C&C-palvelimien verkkotunnuksia ja sulkevat niitä säännöllisesti, botmasterit luovat dynaamisesti suuria määriä satunnaisia verkkotunnusnimiä. Näistä nimistä vain pieni osa valitaan lopulliseen C&C käyttöön, loput toimivat harhautuksena eivätkä johda mihinkään.

Usein DGA-algoritmeilla luodut verkkotunnusnimet ovat käytössä vain lyhyen ajan, minkä jälkeen samalla algoritmilla luodaan uusi verkkotunnusnimi. (Antonakakis, Perdisci, Nadji, Vasiloglou & Abu-Nimeh (2012)) DGA-algoritmeja hyödyntävät botit ovat pystyneet luomaan jopa 50 000 nimivaihtoehtoa tunnissa, mikä tekee tutkijoille näiden nimien estämisestä tai rekisteröimisestä haittatietokantaan miltei mah-

dotonta, sillä haittatietokantojen ylläpidosta ja käyttämisestä tulee hidasta ja raskasta. (Stalmans & Irwin (2011)) Alla yksinkertainen esimerkki DGA-algoritmista python kielellä.

```
def generate_domain(year, month, day):
  domain = ""
  for i in range(16):
     year = ((year ^ 8 * year) >> 11) ^ ((year & 0xFFFFFFF0) << 17)
     month = ((month ^ 4 * month) >> 25) \^ 16 * (month & 0xFFFFFFF8)
     day = ((day ^ (day << 13)) >> 19) ^ ((day & 0xFFFFFFFE) << 12)
     domain += chr(((year ^ month ^ day) % 25) + 97)
  return domain
```

Kuten yllä näkyy, tämä aliohjelma palauttaa joka päivänä hiukan erilaisen verkkotunnusnimen, muttei esimerkiksi luo useita tuhansia nimiä ja tallenna niitä listaan kuten monimutkaisemmat DGA-algoritmit.

Luonnollisesti jos botnetin DGA-algoritmi tunnetaan, voidaan luoda ennalta mahdollisia C&C-palvelimien verkkotunnuksia ja tämän avulla selvittää mitä verkkotunnusta botnetti tulee käyttämään. Tällöin ennaltaehkäisevä reagointi kyseisen botnetin toimintaan helpottuu. Suuri osa havaitsemismetodeista perustuu haittaohjelmien takaisinmallintamiseen, mutta tämä ei ole aina mahdollista. (Stalmans & Irwin (2011))

4.3 DNS fast-flux

DGA-algoritmeilla luodaan monta verkkotunnusta joista vain osa valitaan käyttöön kun taas DNS fast-flux-tekniikassa verkkotunnusnimi (domain name) säilyy, mutta itse IP-osoite vaihtuu jatkuvasti sisällön ylläpitämisen vastuun siirtyessä botilta toiselle.

DNS fast-flux toimii niin, että luodaan lukuisia IP-osoitteita vastaamaan yhtä yksittäistä täyttä verkkotunnusnimeä (FQDN, fully qualified domain name).(Stalmans

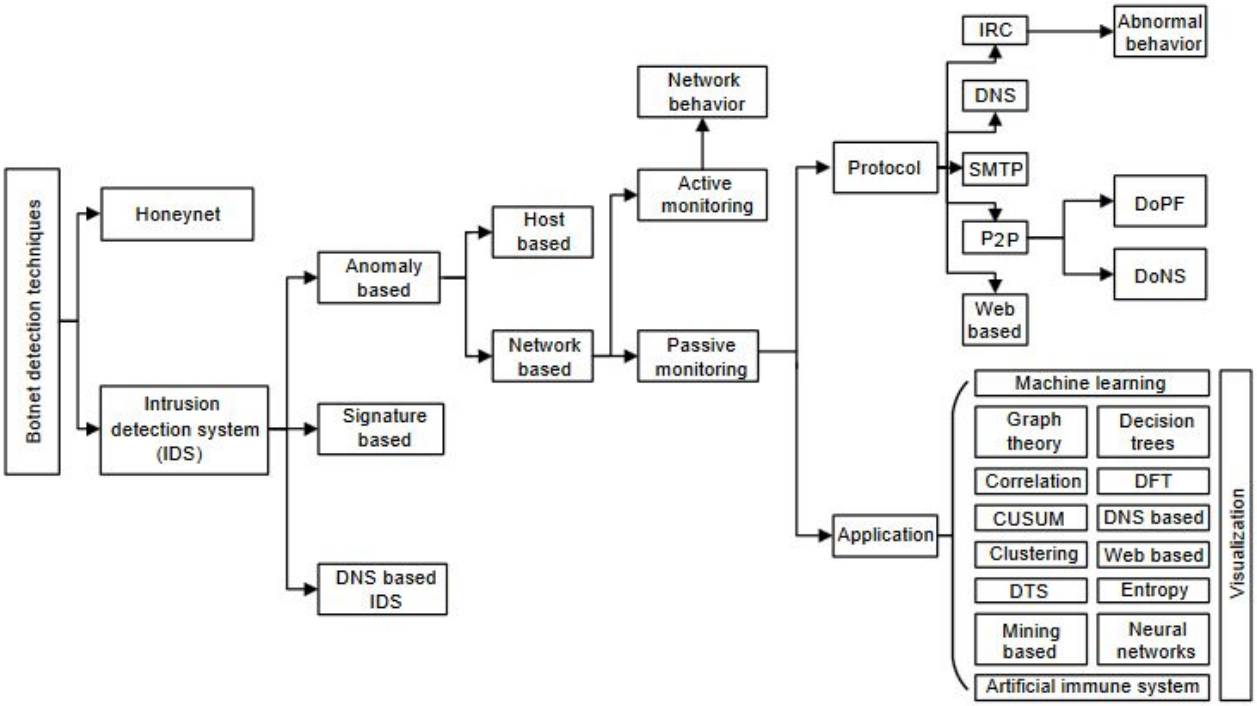
& Irwin (2011) Koska yksittäiset botit voivat sulkeutua ja niiden verkkoyhteys voi katketa suhteellisen satunnaisesti, fast-flux verkkotunnukset mainostavat useita eri IP-osoitteita niille osoitettujen DNS-kyselyiden vastauksina.

DNS fast-flux-botnetit luokitellaan kahteen eri luokkaan: single-flux ja double-flux. Single-fluxiksi lasketaan botnetit jotka vaihtavat vain toista arkistotiedoista eli joko sisältöön liittyviä palvelimia (content servers) tai nimeen liittyviä palvelimia (name servers). Jos fast-flux-verkkotunnus vaihtaa molempia näistä säännöllisesti, se lasketaan double fast-fluxiksi. (Hu, Knysz & Shin (2011))

DNS fast-flux tuo ylimääräistä turvallisuutta C&C-kanavan ylläpitämiseen, mutta tämäkään ei takaa ettei botnettiä havaittaisi. Koska kyseinen tekniikka yleistyy jatkuvasti, on esitetty useita erilaisia tapoja tarkkailla DNS-tietoliikennettä botnettien ja muiden haittaohjelmien havaitsemiseksi. (Hu ym. (2011)) Näitä tapoja ovat muun muassa DNS-liikenteen tutkiminen verkon uloskäyntipisteiltä, joiden liikennettä analysoimalla voidaan havaita tartunnan saaneita laitteita verkostossa. Toimivaksi on havaittu myös pahansuovan fast-flux-liikenteen havaitseminen rekursiivisen DNS-liikenteen jättämien jälkien analysoimisella. Tällaisella analyysilla liikenteessä havaittiin tavanomaisesta liikenteestä poikkeavia piirteitä: esimerkiksi verkkotunnusnimen lyhytikäisyys ja arkistot, jotka osoitttivat usean osoitteen vastanneen samaa nimeä. Usein IP-osoitteet jotka olivat pahansuopia olivat kyseisen verkon ulkopuolisia ja muuttuivat usein. (Hu ym. (2011))

5 Botnettien havaitseminen

Botnettien havaitsemistekniikat luokitellaan kahteen laajaan kategoriaan: IDS (Intrusion Detection System) ja honeynet-järjestelmät. Näistä tutkielmassa käsitellään lyhyesti honeynetit, poikkeamapohjainen havainnointi, tunnistepohjainen havainnointi ja DNS-havainnointi. Tarkempi luokittelu on esitetty kuviossa 6.



Kuvio 6. Botnettien havaitsemisen kategoriat.

5.1 Honeynet

Honeynetit ovat tietoverkostoja, jotka on suunniteltu vetämään puoleensa ja kaappaamaan pahansuopien käyttäjien transaktioita. Kun yksi honeynettiin kuuluvista laitteista on säädetty keräämään talteen tätä dataa, sitä kutsutaan honeypotiksi. Useimmissa tapauksissa honeypot, jossa on yksinkertainen tietoturva-aukko, saa osakseen monta hyökkäysyritystä minuuttien sisällä. (Paxton ym. (2015))

Honeynetin päämäärä on tallentaa dataa ja löytää yleinen kaava pahansuovassa lii-

kenteessä, ilman että hyökkääjä huomaa honeynettiä. Botnettejä tunnistaessa honeynetin tarkoituksena on saada tartunta ja muuttua botiksi, jolloin sitä kutsutaan honeybotiksi. Tällöin honeynet voi kerätä tietoa varsinaisesta botnetistä, sen rakenteesta, protokollista ja C&C-palvelimesta. Näitä tietoja hyödyntäen on mahdollista löytää tapoja valmistautua ja puolustautua hyökkäyksiltä. (Paxton ym. (2015))

Yksinkertaiset honeynet-toteutukset pyrkivät kaappaamaan ainoastaan automatisoitujen hyökkäyksien jättämät jäljet aliverkon sisällä, jolloin on kyse alhaisen interaktiotason honeynetistä. Vaihtoehtoisesti botnetin voidaan sallia ottaa yhteys aliverkon ulkopuolelle, milloin kyse on korkean interaktiotason honeynetistä. (Paxton ym. (2015))

Karim ym. (2014) esittävät honeynetin hyviä sekä huonoja puolia: honeynetin hyviä puolia ovat sen yksinkertainen käynnistäminen, suhteellisen pienet resurssivaatimukset ja hyödyllisyys kryptatun datan kanssa asioidessa. Heidän esittämänsä huonot puolet ovat seuraavat:

1. Honeynetin skaalautuvuus on rajallinen, sillä honeynet vaatii laitteiston käyttöönottoa.
2. Honeybottien ei voida olettaa löytävän internet-hyökkäyksiä, sillä tämä tekniikka havaitsee haitallista toimintaa vain ollessaan vuorovaikutuksessa haitallisen toiminnan kanssa.
3. Ansoiksi asetettujen järjestelmien löytäminen voi olla hankalaa eli botmaster ei aina päädy infektoimaan honeynettiä.
4. Joskus hyökkääjä voi ottaa täyden kontrollin tartuttamastaan honeybotista ja hyödyntää sitä vahingoittaakseen muita järjestelmiä ja/tai laitteita honeynetin ulkopuolella.

5.2 Intrusion detection system

Yksinkertaistettuna IDS on jonkinlainen järjestelmä, jonka päämääränä on hälyttää käyttäjää havaitessaan jonkinlaisen tietoturva-uhan. Useimmiten IDS-sensorit tarkkailevat verkostossa liikkuvia paketteja, järjestelmätiedostoja ja lokitiedostoja. IDS-

järjestelmän voi myös asentaa osaksi honeynettiä. (Schiller& Binkley 2011, s. 156)

Tarkemmin IDS-järjestelmien hyödyntämät metodologiat on luokiteltu kolmeen laajaan kategoriaan: tunniste-pohjainen havainnointi, DNS-havainnointi ja poikkeamapohjainen havainnointi. (Karim ym. (2014))

5.3 Tunnistepohjainen havainnointi

Tunniste on kaavamaisuus tai tekstinpätkä, joka vastaa jotakin tunnetun hyökkäyksen tai uhan sisältöä. Nämä tunnisteet voivat olla esimerkiksi lähdeosoite, päämääräosoite, portit, tietosisältö tai metadata. Tunnistepohjainen havainnointi on prosessi, jossa verrataan samankaltaisuuksia aiemmin kaapattun pahansuovan liikenteen ja senhetkisen liikenteen välillä.

Koska kyse on tietyistä aiemmin havaituista hyökkäyksistä tai laitteistojen haavoittuvuuksista kerätyn tiedon hyödyntämisestä, tunnistepohjainen havainnointi tunnetaan myös nimillä tietoon perustuva havainnointi tai väärinkäyttö havainnointi (Knowledge based Detection, Misuse Detection). (Liao, Lin, Lin & Tung (2013))

Tunnistepohjaisen havainnoinnin vahvuutena on se, että muiden metodien tavoin se ei tarvitse laajan verkoston tai tietoliikenteen havaitsemista. Tunnistepohjaiselle havainnoinnille riittää havaita yksittäinen yhteydenotto, joka vastaa aiemmin löydettyä tunnistetta. Tämän lisäksi metodin etuja ovat suuri tarkkuus ja minimaalinen määrä vääriä positiivisia tunnisteita. (Modi, Patel, Borisaniya, Patel, Patel & Rajarajan (2013))

Zand, Vigna, Yan & Kruegel (2014) kuitenkin huomauttavat, että ”Ongelma tunnistehavaitsemisessa on se, että ne eivät havaitse uusia ennen tuntemattomia tai päivitettyjä botnettejä. Tämän takia ne vaativat säännöllisiä päivityksiä omaan tunnistetietokantaansa. Ikävä kyllä, tämänhetkinen lähestymistapa tunnisteiden kehittämiseen on suurimmaksi osaksi manuaalinen. Tämä prosessi on työläs ja aikaa vaativa.” Tähän he lisäävät, että viimeaikainen tutkimus tämän havaitsemistavan osalta perustuu suuresti tapoihin automatisoida kyseinen työläs prosessi.

5.4 DNS-pohjainen havainnointi

DNS-pohjainen havainnointi perustuu botnettien aiheuttaman DNS-liikenteen tunnistamiseen. Metodeina hyödynnetään DNS-pyyntöjen tallentamista ja niiden analysointia, solmun historian arviointia ja joukkoliikenteen analysointia. Kuten Ichise, Jin & Iida (2015) toteavat: ”Verrattuna IRC-, HTTP- ja P2P-protokolliin DNS on fundamentaalinen protokolla internetissä. DNS ei ole ainoastaan ole olennainen protokolla IP-osoitteiden muuttamisessa nimiksi, vaan myös monet internet sovellukset käyttävät sitä sähköpostiviestimisessä. Tästä syystä kaiken DNS liikenteen sulkeminen botnet-liikenteen pysäyttämiseksi on mahdotonta.” On siis tärkeää pystyä erottamaan pahansuopa liikenne ja tavanomainen DNS-liikenne.

Eroavaisuudet botnettien liikenteessä ja normaalissa liikenteessä voi esittää kolmella pääperiaatteella: (Keermic (2011))

Botnettien yhden suhde moneen: Seuraamalla DNS-liikennettä voidaan huomata usean eri lähteen tekevän samanlaisia hakuja suunnilleen samanaikaisesti. Tämänkaltainen ryhmätoiminta kielii yhdestä liikennettä aiheuttavasta järjestelmästä eli ryhmätoiminnan avulla voidaan havaita botnettejä.

Botnettien synkronisaatio: Kun C&C-palvelin välittää käskynsä boteille, ne kommunikoivat samanaikaisesti ja hyökkäävät samanaikaisesti. Tämän perusteella voidaan havaita bottiryhmään liittyvän liikenteen kasvava määrä verrattuna normaaliin liikenteeseen.

Bottien vastausaika: Bottien vastaanottaessa käskynsä botmasterilta ne suorittavat ohjeistettuja aktiviteetteja vakiovastausajalla verrattuna vaihtelevaan normaaliin liikenteeseen. Tämän perusteella voidaan mitata vastausaikoja bottien läsnäolon havaitsemiseksi verkossa.

Tällainen spesifi käyttäytyminen on havaittavissa botnettien luomassa DNS-liikenteessä. Esimerkiksi kun C&C-palvelin siirtyy uuteen osoitteeseen syntyy lukuisa määrä samanlaisia DNS-kyselyitä boteilta, jotka etsivät uuden C&C-palvelimen sijaintia. Tätä liikennettä tarkkailemalla ja arvioimalla tilastollisin menetelmin pystytään luo-

maan erilaisia botnettien havaitsemisalgoritmeja. (Choi & Lee (2012))

5.5 Poikkeamapohjainen havainnointi

Poikkeamapohjainen havainnointi on pääasiallinen tutkimuksen osa-alue botnettien havaitsemisessa. (Karim ym. (2014)) Poikkemapohjaiset IDS-järjestelmät voidaan laskea kahteen eri alakategoriaan: isäntäpohjainen (host based, HIDS) tai verkkopohjainen (network based, NIDS). ((Schiller& Binkley 2011, s. 156))

HIDS tarkkailee suojellun järjestelmän toimintaa ja etsii epäilyttäviä tiedostoja, sovelluksia tai palvelujen käyttämistä. NIDS puolestaan monitoroi koko verkkoa, pelkän yksittäisen järjestelmän sijasta ja saa useimmat tuloksensa verkon pakettianalyysien perusteella.

Kummassakin tekniikoista on omat hyvät puolensa. HIDS huomaa helpommin hyökkäyksiä vertaisjärjestelmiltä, järjestelmän sisäisen peukaloinnin pahansuovilta käyttäjiltä ja pahansuovan koodin syöttämisen järjestelmään. Tätä pahansuopaa koodia voidaan syöttää järjestelmään esimerkiksi irrotettavalta medialta kuten USB-tikulta tai CD-levyltä. NIDS huomaa helpommin palvelunestohyökkäykset liikenteen analysoimisen avulla ja omaa paremmat edellytykset tiettyjen porttien tarkkailemiseen. (Schiller& Binkley 2011, s. 156)

Periaatteessa NIDS metodin idea on seuraava: Etsitään kaavamaisuuksia laajassa tietoliikennedatassa. Mikäli tällaista kaavamaista ja poikkeavaa toimintaa liikenteessä huomataan, laukaistaan hälytys. (Agarwal & Mitta (2012)) Poikkeavuuksia etsitään joko aktiivisella tai passiivisella monitoroinnilla. Aktiivisessa monitoroinnissa verkostoon syötetään paketteja, joiden avulla etsitään pahansuopaa toimintaa verkostossa. Tätä tekniikkaa ei tosin hyödynnetä kovin yleisesti, sillä se tuo ylimääräistä kuormitusta verkkoon.

Passiivisessa monitoroinnissa etsitään poikkeavuuksia datassa, jota kaapataan kun se kulkee jonkin laitteen, esimerkiksi reitittimen läpi. (Karim ym. (2014)) Nämä poikkeavuudet voivat olla esimerkiksi liikennettä joka liikkuu epätavallisen por-

tin kautta, korkea latenssi, kasvanut liikenteen määrä tai pahansuopaa toimintaa indikoiva järjestelmäkäyttäytyminen.

Tekniikat joita on tutkittu poikkeavuuspohjaisen havainnoinnin tehokkaaksi toteuttamiseksi sisältävät muun muuassa datan louhintaa ja tilastollista mallintamista. Avainelementti tämän lähestymistavan käyttämisessä on sääntöjen luominen. Näillä säännöillä pyritään tunnistamaan pahansuopa liikenne ja laskemaan väärien positiivisten havaintojen määrää sekä tunnettujen että tuntemattomien uhkien osalta. (Modi ym. (2013))

Poikkeavuuspohjaisessa havainnoinnissa kyse on siis pääasiallisesti tilastollisten menetelmien hyödyntämisestä verkkoliikenteen analysoimisessa. Näillä tilastollisilla menetelmillä pyritään etsimään normista poikkeavaa käyttäytymistä, hyvin samankaltaisesti kuin DNS-pohjaisessa havainnoinnissa. Kuvion 6 mukaisesti passiivinen monitorointi voidaan vielä jakaa useisiin alakategorioihin sen hyödyntämien metodejen mukaan: tarkkailtavan protokollan perusteella tai tarkkailemiseen rakennetun sovelluksen ominaisuuksien perusteella.

6 Yhteenveto

Tutkielmassa esiteltiin laajalta näkökulmalta mikä on botnet, sen yleinen rakenne, yleisimmät botnettien esiintymismuodot ja luokitteluperusteet, botnettien C&C-liikenteen piilottamismetodit ja C&C-liikenteen havaitsemisen perusteet.

Luku 2 keskittyi laajaan näkökulmaan, pyrkien muodostamaan lukijalle mahdollisimman selkeän kuvan botnettien yleisestä koostumuksesta ja toimintaperiaatteista. Luvussa 3 kyseistä käsitettä avattiin enemmän ja esiteltiin lyhyesti botnettien yläkategorioita: IRC-pohjaiset botnetit, HTTP-pohjaiset botnetit, P2P-pohjaiset botnetit, mobile botnetit, pilvibotnetit ja muutama vaihtoehtoisia kommunikaatiotapoja hyödyntävä botnet. Näiden luokitteluerojen osoitettiin pohjautuvan arkkitehtuuriin, käskyjen vastaanottamistapaan ja kommunikaatioprotokollaan.

Luvussa 4 rakennettiin aiempien lukujen tarjoaman tietämyksen pohjalta näkökulmaa botnettien erilaisiin tapoihin piilottaa omaa C&C-liikennettään. Huomiota saivat kryptaus, DGA-algoritmit ja DNS fast-flux. Lopulta luvussa 5 esiteltiin botnettien havaitsemiseen käytettäviä metodeja, milloin huomiota saivat honeynetit ja IDS-järjestelmien erilaiset alakategoriat.

Kirjallisuuskatsauksen tulokset osoittavat, että mahdollisimman kestävän botnetin luomiseksi kannattaisi hyödyntää hajautettua arkkitehtuuria ja esimerkiksi P2P- tai HTTP-protokollaa, vaikka tällaisen botnetin rakentaminen voisi osoittautua melko haastavaksi. C&C-liikenteen piilottamismetodeita käsittelevässä kappaleessa tähän lisättiin, että huomaamattoman botnetin saavuttamiseksi kaikki botnetin sisäinen viestintä kannattaisi kryptata esimerkiksi jollakin esitetyllä metodilla ja C&C-palvelinta kannattaisi ylläpitää hyödyntäen joko DNS fast-fluxia tai DGA-algoritmia, sillä vaikka näiden toteuttaminen lisäisi hastavuutta itse botnetin suunnittelemisessa ja rakentamisessa, on C&C-liikenteen piilottamismetodien tarjoama suoja havaitsemismetodeja vastaan huomattava.

Mikäli päämääränä on teknisesti mahdollisimman yksinkertaisen botnetin rakentaminen, lienee esimerkiksi IRC-protokollaan pohjautuva botnet keskitetyllä arkki-

tehtuurilla varteenotettava vaihtoehto. Tämänkaltaisen botnetin heikkoutena olisi tosin myös sen helppo havaitseminen ja toiminnan sulkeminen, minkä seurauksena botnetin suunnittelussa oikea valinta lienee etsiä kultainen keskitie teknisesti helpon toteutuksen ja kestävän toteutuksen väliltä. Kaikesta huolimatta luvun 5 esittelemien havainnointikeinojen ansiosta täydellinen C&C-liikenteen piilottaminen lienee mahdotonta.

Kokonaisuudessaan kirjallisuuskatsauksen pyrkimyksenä oli tarjota yhtenäinen ja kattava tietokatsaus tähän nykypäivän internetin ”maanalaisen ekonomian linkkuveitseen” (Hung-Chang ym. (2011)) ja siihen liittyviin ilmiöihin. Tämän kirjallisuuskatsauksen tarjoaman tietämysperustan ja lähdeluettelon avulla lukijalla on mahdollisuus tutustua omatoimisesti ilmiön eri osa-alueisiin tarkemmin.

Jatkossa tarkemman tutkimuksen aiheeksi kävisi miltei mikä tahansa esitellyistä aiheista: Eri tyyppiset botnetit, C&C-liikenteen piilottamismetodit ja botnettien havaitsemiskeinot ovat kaikki laaja-alaisia käsitteitä. Jokainen näistä osa-alueista omaa lukuisia eri osa-alueita, joista jokaisesta löytyy tutkittavaa. Tosin olennaisinta lienee vasta viime vuosina ilmentyneiden ilmiöiden tutkiminen: Pilvi- ja mobiili botnetit ovat kummatkin varsin uusia ja vähän tutkittuja osa-alueita tästä ilmiöstä. Erityisesti näitä kahta osa-aluetta kannattaisi tutkia tarkemmin, sillä niiden yleistyminen tulevaisuudessa on vääjäämätöntä.

Kirjallisuutta

Gramam, M. & Winckles, A. 2014. *An Analysis of Pre-Infection Detection Techniques for Botnets and other Malware*. Cybercrime Forensics Education and Training (CFET) 2014, Proceedings of the 7th International conference on

Gramam, M., Winckles, A. & Moore, A. 2014. *Botnet Detection in Virtual Environments Using NetFlow*. Cybercrime Forensics Education and Training (CFET) 2014, Proceedings of the 7th International conference on

Lin W. & Lee D. 2014. *Traceback Attacks in Cloud – Pebbletrace Botnet*. Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on

Li C., Jiang W. & Zou X. 2009. *Botnet: Survey and Case Study*. Innovative Computing, Information and Control (ICICIC), 2009 Fourth International Conference on

Hung-Chang C. & Guo-Quan W. 2011. *Construction P2P firewall HTTP-Botnet defense mechanism*. Computer Science and Automation Engineering (CSAE), 2011 IEEE International Conference on (Volume:1) Shanghai:IEEE, s. 33–39

Nagaraja S., Houmasadr A., Piyawongwisal P., Sing V., Agarwhal P. & Borisov N. 2011. *Stegobot: A Covert Social Network Botnet*. 13th International Conference, IH 2011, Prague, Czech Republic, May 18–20, 2011, Revised Selected Papers, Springer Berlin Heidelberg, s. 299–313

Schiller C. & Binkley J.R. 2011. *Botnets: The Killer Web Applications*. Syngress, 18.4.2011.

Schiller C. & Binkley J.R. 2011. Vania J., Meniya A. & Jethva H. B. *A Review on Botnet and Detection Technique*. International Journal of Computer Trends and Technology, Volume 4, Issue 1, 2013, s.23–29

Karim A., Salleh R.B., Shiraz M., Shah S.A.A, Awan I. & Anuar N.B. 2014. *Botnet detection techniques: review, future trends, and issues*. Frontiers of Information Technology & Electronic Engineering, Volume 15, Issue 11, 2014, s.943-983

Mustapha B.Y.,Granadillo G.G. & Debar H. 2011. *Botnets: lifecycle and taxonomy*. Network and Information Systems Security (SAR–SSI), 2011 Conference on

Rodriguez-Gomez R.A.,Macia-Fernandez G. & Garcia-Teodoro P. *Survey and Taxonomy of Botnet Research through Life–Cycle*. ACM Computing Survey. Volume 45,

Issue 4 (2013).

Haddadi F., Cong D.L., Porter L. & Zincir-Heywood A.N. *On the Effectiveness of Different Botnet Detection Approaches*. Lecture Notes in Computer Science, Volume 9065, s.121–135

Houmansadra A.& Borisovb N. 2013 *BotMosaic: Collaborative network watermark for the detection of IRC-based botnets*. Journal of Systems and Software, Volume 86, Issue 3, March 2013, s.707—715

Scanlon M. & Kechadi T. 2012 *Peer–to–Peer Botnet Investigation: A Review*. Lecture Notes in Electrical Engineering, Volume 179, s.231—238

Paxton N.C., Jang D.I., Russell S., Ahn G.J., Moskowitz I.S. & Hyden P. 2015 *Utilizing Network Science and Honeynets for Software Induced Cyber Incident Analysis*. System Sciences (HICSS), 2015 48th Hawaii International Conference on, s.5244-5252

Brezo F., Gaviria de la Puerta J., Santos I., Barroso D., Bringas P.G. 2012 *C&C Techniques in Botnet Development*. Advances in Intelligent Systems and Computing, Volume 189, s.97–108

Shanti K. & Seenivasan D. 2015 *Detection of botnet by analyzing network traffic flow characteristics using open source tools*. Intelligent Systems and Control (ISCO), 2015 IEEE 9th International Conference on, 9–10.1.2015

Seenivasan D. & Shanti K. 2014 *Categories of Botnet: A Survey*. International Journal of Computer, Electrical, Automation, Control and Information Engineering, Volume 8, Issue 9, 2014

Zhu W.& Lee C. 2015 *Internet security protection for IRC-based botnet*. Electronics Information and Emergency Communication (ICEIEC), 2015 5th International Conference on, 14–16.5.2015

Éslahi M., Hashim H. & Tahir N.M. 2013 *An Efficient false alarm reduction approach in HTTP based botnet detection*. Computers & Informatics (ISCI), 2013 IEEE Symposium on, 7–9.4.2013, s. 201–205

Cai T. & Zou F. 2012 *Detecting HTTP Botnet with Clustering Network Traffic*. Wireless Communications, Networking and Mobile Computing (WiCOM), 2012 8th International Conference on, 21–23.9.2012

Alomari E., Manickam S., Gupta B.B., Singh P. & Anbar M. 2014 *Design, deployment*

and use of HTTP-based botnet (HBB) testbed. Advanced Communication Technology (ICACT), 2014 16th International Conference on, 16–19.2.2014, s. 1265–1269

Ricardi M., Di Pietro R. & Vila J.A. 2011 *Taming Zeus by leveraging its own crypto internals*. eCrime Researchers Summit (eCrime), 2011, 7–9.11.2011

Silva S.S.C, Silva R.M.P, Pinto R.C. & Salles R.M. 2013 *Botnets: A survey*. Computer Networks, Volume 57, Issue 2, 4.2.2013, Pages 378—403

Zhang J., Perdisci R., Lee W., Sarfraz U.& Luo X. 2011 *Detecting stealthy P2P botnets using statistical traffic fingerprints*. Dependable Systems & Networks (DSN), 2011 IEEE/IFIP 41st International Conference on, 27-30.7.2011, s. 121–132

Zeng Y., Shin K.G. & Hu X. 2012 *Design of SMS commanded-and-controlled and P2P-structured mobile botnets*. WISEC '12 Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks, s. 137–148

Alzahrani A.J. & Ghorbani A.A. 2014 *SMS mobile botnet detection using a multi-agent system: research in progress*. ACySE '14 Proceedings of the 1st International Workshop on Agents and CyberSecurity, Artikkeli nro. 2

Geng G., Xu G., Zhang M., Guo Y., Yang G. & Wei C. (2012) *The Design of SMS Based Heterogeneous Mobile Botnet*. Journal of Computers, Vol 7, Issue 1 (2012), s. 235–243, 1.2012

Pieterse H. & Olivier M.S. (2014) *Bluetooth Command and Control channel*. Computers & Security, Volume 45, 9.2014, s. 75–83

Graham M., Winckles A. & Sanchez-Velazquez E.(2015) *Botnet detection within cloud service provider networks using flow protocols*. Industrial Informatics (INDIN), 2015 IEEE 13th International Conference on, 22–24.7.2015, s.1614–1619

Studer R.(2011) *Economic and Technical Analysis of BotNets and Denial-of-Service Attacks*. Communication Systems IV, University of Zurich, s. 21–29

Dietrich C.J., Rossow C., Freiling F.X., Bos H., van Steen M. & Pohlman N. (2011) *On Botnets That Use DNS for Command and Control*. 2011 Seventh European Conference on Computer Network Defense, Gothenburg, Sweden, 6–7.9.2011 ,s. 9–16

Rossow C. & Dietrich C.J. (2013) *ProVeX: Detecting Botnets with Encrypted Command and Control Channels*. Lecture Notes in Computer Science, Volume 7967, s. 21–40

Khattak S., Ramay N.R., Syed A.A & Khayam S.A. (2013) *A Taxonomy of Botnet Be-*

havior, Detection, and Defense. IEEE Communications Surveys & Tutorials, Volume 16, Issue 2, 2.10.2013, s. 898– 924

Plohmann D. & Gerhards-Padilla E. (2012) *Case study of the Miner Botnet*. Cyber Conflict (CYCON), 2012 4th International Conference on, 5–8.7.2012

Andriesse D., Rossow C., Stone-Gross B., Plohmann D. & Bos H. (2013) *Highly resilient peer-to-peer botnets are here: An analysis of Gameover Zeus*. Malicious and Unwanted Software: "The Americas" (MALWARE), 2013 8th International Conference on, 22–24.9.2013, s. 116–123

Antonakakis M., Perdisci R., Nadji Y., Vasiloglou N. & Abu-Nimeh S. (2012) *From Throw-Away Traffic to Bots: Detecting the Rise of DGA-Based Malware*. 21st USENIX Security Symposium (USENIX Sec'12), Bellevue, WA, USA 8–9.8.2012

Stalmans E. & Irwin B. (2011) *A framework for DNS based detection and mitigation of malware infections on a network* . Information Security South Africa (ISSA), 15–17.8.2011

Hu X., Knysz M. & Shin K.G. (2011) *Measurement and analysis of global IP–usage patterns of fast-flux botnets*. INFOCOM, 2011 Proceedings IEEE, 10–15.4.2011, s. 2633–2641

Liao H., Lin C.R., Lin Y. & Tung K. (2013) *Intrusion detection system: A comprehensive review*. Journal of Network and Computer Applications, Volume 36, Issue 1, January 2013, s. 6–24

Modi C., Patel D., Borisaniya B., Patel H., Patel A. & Rajarajan M. (2013) *A survey of intrusion detection techniques in Cloud*. Journal of Network and Computer Applications, Volume 36, Issue 1, January 2013, s. 42–57

Zand A., Vigna G., Yan X. & Kruegel C. (2014) *Extracting probable command and control signatures for detecting botnets*. SAC '14 Proceedings of the 29th Annual ACM Symposium on Applied Computing, NY, USA, 2014, s. 1657–1662

Ichise H., Jin Y. & Iida K. (2015) *Detection Method of DNS-based Botnet Communication Using Obtained NS Record History*. Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual, Volume 3, 1–5.7.2015, s. 676–677

Choi H., & Lee H. (2012) *Identifying botnets by capturing group activities in DNS traffic*. Computer Networks, Volume 56, Issue 1, 12.1.2012, s. 20-33

Agarwal B. Mittal M. (2012) *Hybrid Approach for Detection of Anomaly Network Traffic using Data Mining Techniques*. Procedia Technology, Volume 6, 2012, s. 996—1003

Keermic V. (2011) *Inspecting DNS Flow Traffic for Purposes of Botnet Detection*. as GEANT3 JRA2 T4 Internal Deliverable, 2011