Deckblatt

Abstracts

Eidesstattliche Erklärung

Inhaltsverzeichnis

Abkürzungsverzeichnis

Abbildungsverzeichnis

1 Einleitung

...Was was umfasst die Einleitung...

1.1 Grund der Studienarbeit

Die Digitalisierung ist aktuell ein Faktor, welcher viele Prozesse in der Welt vorantreibt. Die Menschen werden vernetzter, Unternehmen können ihre Daten besser und effizienter Speichern und wir entwickeln immer bessere Kommunikationsmöglichkeiten. Aber gerade mit dieser stetigen Verbesserung der Technik erfolgt auch eine immer größere Abhängigkeit von den Computern, welche uns helfen, die Produktivität voranzutreiben. Cyber Angriffe nehmen auf der ganzen Welt rasant zu und verursachen Schäden in Milliardenhöhe. Laut einer Umfrage der Online Plattform Statista gaben 47% der befragten deutschen Unternehmen an, im Jahr 2022 bereits Opfer eines Cyber Angriffs geworden zu sein.[1]Hinter vielen dieser Angriffe befinden sich Botnetze, also Netzwerke aus kompromittierten PCs, welche, ohne das Wissen des Benutzer dafür genutzt werden, verschiedene bösartige Aktionen auszuführen.

1.2 Ziel dieser Arbeit

Das Ziel dieser Arbeit soll es sein, dem Leser einen Einblick in die Funktionen und Arbeitsweisen eines Botnetzes zu geben, und ihn darüber aufklären, wie sich diese verbreiten und wachsen können. Es sollen zu Beginn verschiedene Grundlagen erklärt und definiert werden, welche für das Weitere Verständnis wichtig sind. Des Weiteren soll auf die Funktion der Botnetzen sowie die daraus resutierenden Gefahren eingegangen werden. Anhand von praxisorientierten Beispielen, wie der Code Analyse eines realen Botnetz, sollen die Funktionsweisen eines Botnetz noch genauer veranschaulicht werden. Im Abschließenden Kapitel soll schließlich erklärt werden, wie man Netzwerkgeräte entsprechend sichert, um somit ausreichend Schutz garantieren zu können. Des

1 https://de.statista.com/statistik/daten/studie/1230157/umfrage/unternehmen-die-in-den-letzten-12-monaten-eine-cyber-attacke-erlebt-haben/ (15.10.2022)

Weiteren wird auch ein Ausblick gegeben, wie sich Botnetze in der Zukunft dank modernerer Technik weiterentwickeln könnten.

2 Grundlagen

2.1 Lebenszyklus eines Botnetz

Der Lebenszyklus eines Botnets besteht im Grunde genommen aus 3 verschiedenen Hauptphasen, welche im folgenden Abschnitt genauer erläutert werden sollen.

2.1.1 Phase 1 – Rekrutierungsphase:

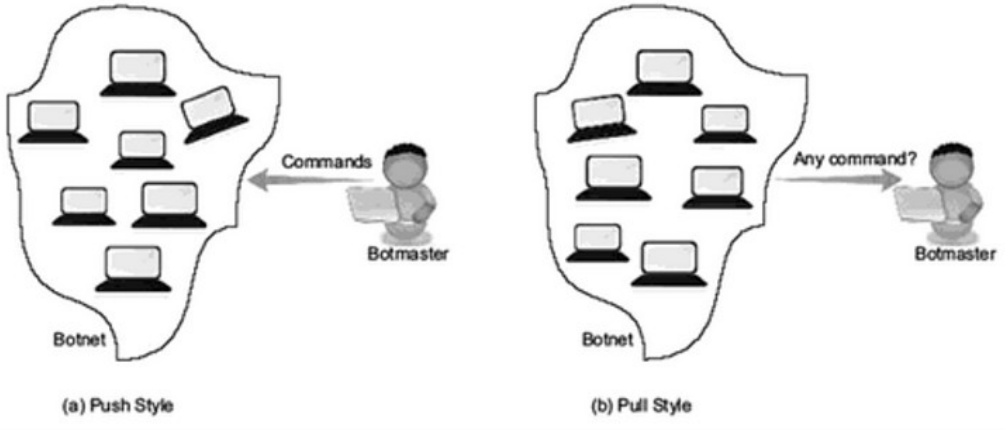
Am Anfang beginnt die Formierung des Botnetz damit, dass es versucht, so viele anfällige Maschinen wie möglich zu befallen, damit diese als entsprechende Bots für das Botnetz eingesetzt werden können. Hierfür werden die anfälligen Maschinen durch verschiedene Mechanismen mit dem Bot Code befallen. Einer der meistgenutzten Mechanismen bedient sich hierbei der Propagierungsmethoden klassischer Computerwürmer. Diese erfordern nicht einmal eine Interaktion mit dem Benutzer, da sich Würmer selbstständig über das lokale Netz und das Internet verbreiten, indem sie aktive Scans nach bekannten Schwachstellen durchführen und somit anfällige Maschinen finden können. Es gibt nun einige Mechanismen, diese anfälligen Maschinen für das Botnetz zu rekrutieren, diese erfordern jedoch einen gewissen Grad an Benutzerinteraktion. Das wohl mächtigste Werkzeug ist das social Engineering, welches darauf abzielt, menschliche Schwachstellen auszunutzen, um sein Ziel zu erreichen. Hierfür werden weitreichende Phishing Kampagnen über Email und soziale Netzwerke erstellt, welche die Benutzer davon überzeugen sollen, auf bösartige Links zu klicken, welche im Anschluss eine sogenannte Bot Binary herunterladen. In anderen Fällen wird der User dazu verführt, Webseiten zu besuchen welche aktive Inhalte wie JavaScript oder ActiveX Controls beinhalten, was dazu führt, dass automatisch Malware von diesen Seiten heruntergeladen und installiert wird, ohne dass der Benutzer etwas davon mitkriegt. Ein klassischer Weg ist es auch, die Botnetz Binaries über ein physikalisches Medium, wie beispielsweise einen USB Flash Speicher zu verbreiten. Das Problem ist hierbei, dass die Malware in Form eines Executable manuell auf dem Zielrechner installiert werden muss. Diese Methode wird eingesetzt, wenn sich der Zielrechner beispielsweise hinter einem NAT verbirgt und nicht direkt aus dem Internet angesprochen werden kann. [2]

2.1.2 Phase 2- Command & Control Phase:

Die 2. Phase widmet sich nun der Kontrolle der infizierten Maschinen. Der sogenannte Botmaster kommuniziert mit diesen über einen C&C Channel. Es gibt unterschiedliche Arten der Kommunikation zwischen Botmaster und Bots. So gibt es zum einen den Push Style, aber auch den

2 Botnets: Architectures, Countermeasures, and Challenges(CRS Series in Security, Privacy and Trust) Georgios Kambourakis, Marios Anagnostopoulus, Weihzi Meng, Peng Zhou 8. Oktober 2019 **S3-4**

Pull Style. Wird im Push Style kommuniziert, werden die Commandos direkt an den Bot gesendet und dieser reagiert auf diese Anforderungen. Im Pull Style senden die Bots in regelmäßigen Intervallen Nachrichten an den Botmaster, und erbitten neue Commandos. Verdeutlicht wird dies in der Nachfolgenden Abbildung:[3]



2.1.3 Phase 3 – Botnetz Aktivitätsphase:

Unter Aktivität versteht sich eine Reihe von Aktionen und Angriffen wie etwa Scanning oder DDOS, welche von den Bots als Antwort auf ein Kommando ausgeführt werden. Es ist wichtig, die Bandbreite eines kompromittierten Host zu kennen, damit man weiß, wie viel Kapazitäten beispielsweise während einem DDOS Angriff zur Verfügung stehen. Die Bandbreite der Bots kann ermittelt werden, indem sie angewiesen werden, Daten an viele ausgewählte Testserver zu senden und diese Sendeleistung analysiert wird.[4]

2.2 Botnetz Architektur

Es gibt kein einheitliches Format, welchem ein Botnetz folgt. Jedes Netz wird individuell aufgebaut und folgt dabei verschiedenen Mustern und Architekturen. Um ein größeres Verständnis zu diesem Thema zu erlangen, wird im Folgenden Kapitel genauer auf die Architektur und Form der Botnetze eingegangen.

2.2.1 Zentralisierte Botnetze

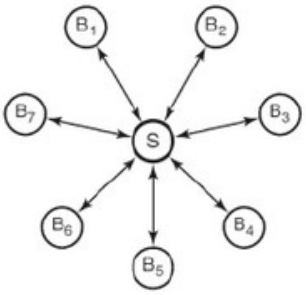
Stern Netz

Bei einer Stern Topologie bekommt jeder Bot-Rechner seine Arbeitsanweisungen direkt vom C&C Server zugewiesen, da bei dieser Architektur die direkte Kommunikation zwischen dem Server und den Bots im Fokus steht. Diese Arbeitsweise erhöht den Durchsatz des Netz, da Aufträge ohne Latenz abgearbeitet werden können und vermieden wird, Daten unnötig über dritte Rechner zu leiten. Dieser vermeintliche Vorteil birgt aber auch gleichzeitig die größte Schwäche dieser Architektur. Da der Server manuell auf jedem Rechner hinterlegt ist, lässt sich dieser leicht

3 Botnets: Architectures, Countermeasures, and Challenges(CRS Series in Security, Privacy and Trust) Georgios Kambourakis, Marios Anagnostopoulus, Weihzi Meng, Peng Zhou 8. Oktober 2019 **S4**

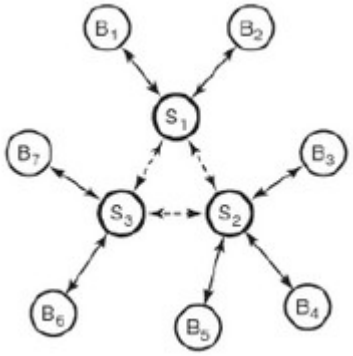
4 Botnets: Architectures, Countermeasures, and Challenges(CRS Series in Security, Privacy and Trust) Georgios Kambourakis, Marios Anagnostopoulus, Weihzi Meng, Peng Zhou 8. Oktober 2019 **S4-5**

entdecken und Übernehmen. Der Server bietet einen Single Point of Failure. Auch bei anderweitigem Ausfall des Servers zerbricht das gesamte Netzwerk.[5]



Multi Server Netz

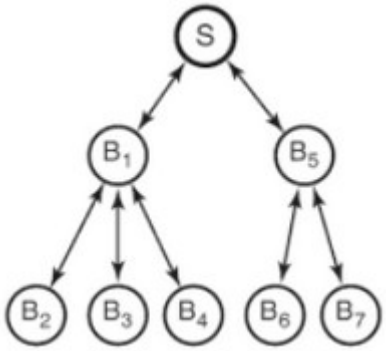
Diese Architektur stellt die logische Erweiterung der Botnetz Stern Topologie dar. Das ein-Server-Modell wurde hierbei um mehrere redundante Server erweitert, es übernehmen mehrere C&C Server die Anweisungsverteilung an die Bots. Jedem Bot wird nun eine Liste der verfügbaren C&C Server mitgegeben und Unerreichbarkeit oder Ausfall einzelner Server spielen nun keine Rolle mehr. Ein dementsprechend größerer Planungsaufbau geht mit diesem Netz somit aber auch einher. Dies ist aber in Kauf zu nehmen, da sich somit der Single Point of Failure vermeiden lässt und die Hauptschwachstelle der Sterntopologie somit neutralisiert wird.[6]



Hierarchische Netzwerke

Eine weitere Art zentralisierter Botnetze stellen die hierarchischen Netze dar. Sie erweitern das Multiserver Netz, indem dem Netz eine weitere, aus Proxyservern bestehende Ebene, hinzugefügt wird. Die Proxys dienen der Sicherheit des Netzes, da sie den Ursprung der Command & Control Server verschleiern und somit deren Entdeckung oder Übernahme erheblich erschweren. Jedoch wird der Aufbau und die Konfiguration solcher Netze schwerer als die bisherigen Strukturen, da dieses Netz durch die Proxyebene eine erhöhte Komplexität aufweist. Auch bremsen die Proxys den Traffic und erhöhen somit die Latenz des Netzwerks.[7]

5 Botnetze: Aufbau, Funktion & Anwendung Matthis C. Laass Fachhochschule Aachen S2
6 Botnetze: Aufbau, Funktion & Anwendung Matthis C. Laass Fachhochschule Aachen S3
7 Botnetze: Aufbau, Funktion & Anwendung Matthis C. Laass Fachhochschule Aachen S3-4

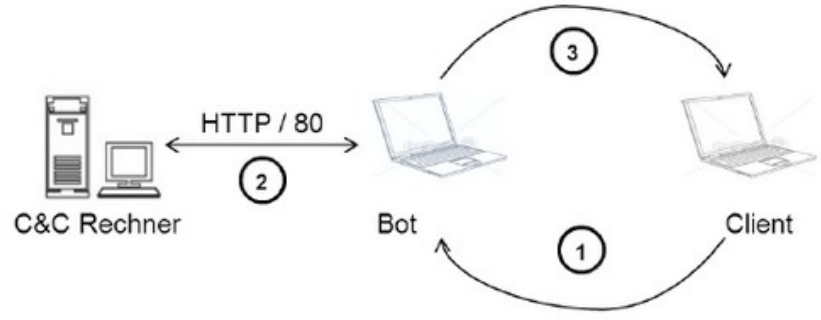


2.2.2 Dezentralisierte Botnetze

Dem Aufbau und der Logik zentralisierter Botnetze stehen nun die sog. Dezentralisierten Botnetze gegenüber. Zentralisierte Botnetze haben in ihrem Aufbau ein großes Problem: Den Single Point of Failure. Alles wird zentral über einen Server geregelt. Fällt dieser aus, bricht das komplette Netz zusammen. Um dieser Sicherheitslücke entgegenzuwirken wurde das Konzept der zentralen Botnetze überarbeitet. Es entstanden die dezentralisierten Botnetze, welche sich durch ein wesentlich komplexeres Design und die Vermeidung des Single Point of Failure auszeichnen.

Fast Flux Botnetze

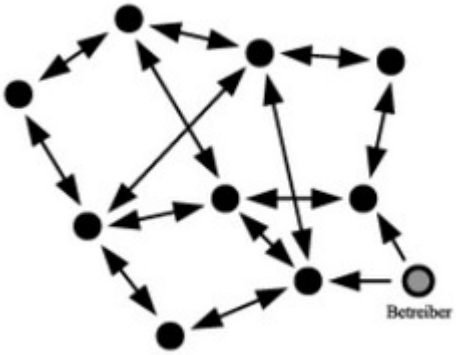
Dieses Netzdesign ist auf der Round Robin Lastenverteilung via DNS aufgebaut. Hierbei existiert eine Control Domain, welche als Nameserver verschiedene Bot IP Adressen eingetragen hat. Diese agieren alle als Proxys. Um nun einen Auftrag zu bekommen, stellt ein Bot eine Anfrage an diese Control Domain. Anschließend wird eine der konstant Rotierenden IP-Adressen der Proxys zurückgegeben, an welchen nun die Anfrage geleitet wird. Dieser Proxyrechner leitet im nächsten Schritt die Anfrage an den Bot Master weiter. Der Bot Master gibt den Auftrag an den Proxy, welcher es zum fragenden Bot weiterleitet. Durch die Rotation liefert eine weitere DNS-Anfrage im Anschluss eine andere IP-Adresse, da es sehr viele Proxys in der Control Domain gibt, unter welchen die Anfragen aufgeteilt werden. Zur Verschleierung des Bot Master werden durch periodische Updates der A Records im DNS-Server geändert. Wenn ein Proxy längere Zeit nicht erreichbar ist, muss sein seine IP-Adresse beim nächsten Update aus dem DNS entfernt werden.[8]



8 Botnetze: Aufbau, Funktion & Anwendung Matthis C. Laass Fachhochschule Aachen S6-7

Peer-to-Peer Botnetze

Peer-to-Peer stellt die bisher vorgestellten Botnetz Architekturen auf den Kopf. Bei diesem Aufbau findet die Kommunikation nicht zwischen Clients und dedizierten Servern statt. Stattdessen sind die einzelnen Netzwerkknoten gleichberechtigt. Somit sind die Clients nicht machtlos, sollte ihnen der zentrale Server eine Anfrage verweigern, oder falls der Dienst ausfällt. Es gibt nämlich keinen zentralen Server mehr. Alle Knoten übernehmen sowohl Server als auch Clientaufgaben. Sie sind ebenbürtig. Durch das Wegfallen einer zentralen Komponente wird das Überwachen als auch das Zerstören eines Peer-to-Peer Botnetz erheblich erschwert. Fallen einzelne Komponenten aus, können sie problemlos ersetzt werden bzw. ein anderer Knoten angesteuert werden.[9]



3 Botnetz Angriffe:

Hinter Botnetzen verbergen sich meist professionell organisierte kriminelle Netzwerke. Haben diese es geschafft, erfolgreich ein Botnetz aufzusetzen, so ist ihre Intention in den meisten Fällen, mit diesem einen Angriff oder andere bösartige Dinge auszuführen. In diesem Kapitel soll auf mögliche Angriffsszenarien und Attacken in Verbindung mit Botnetzen eingegangen werden.

3.1 DDOS (Distributed Denial Of Service)

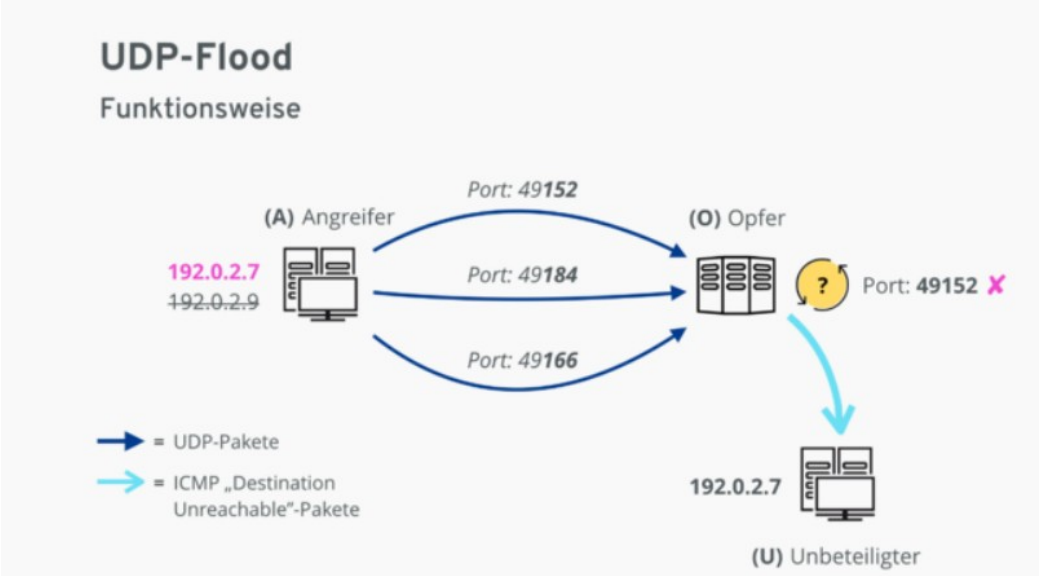
Eine DDOS-Attacke zielt darauf ab, das Zielsystem außer Betrieb zu setzen. Ein oder mehrere ausgewählte Server werden solange gezielt mit Anfragen von den verschiedenen Systemen bombardiert, bis dieser seine eigentlichen Aufgaben nicht mehr erledigen kann. Im schlimmsten Fall kann das komplette Zielsystem unter dieser Last zusammenbrechen. Gerade bei DDOS-Attacken ist es äußerst schwierig, den Angreifer zu ermitteln, da die Anfragen oft von mehreren tausenden Systemen gleichzeitig gesendet werden. DDOS-Attacken lassen sich in folgende Formen einteilen:[10]

9 Botnetze: Aufbau, Funktion & Anwendung Matthis C. Laass Fachhochschule Aachen S7-8

10 https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Cyber-Sicherheitslage/Methoden-der-Cyber-Kriminalitaet/DoS-Denial-of-Service/dos-denial-of-service_node.html Zugriff 06.11.2022

3.2 UDP Flood

Bei einer UDP Flood senden ein oder mehrere Angreifer in einem sehr kurzen zeitlichen Intervall manipulierte Datenpakete an das Zielsystem. Diese sollen das Ziel derart überlasten, sodass legitime Angriffe nicht mehr beantwortet werden können und der eigentliche Service zum Erliegen kommt. Die Funktionsweise der UDP Flood Attacke baut dabei auf den Besonderheiten des User Datagram Protokoll (UDO) auf. Wenn auf einem Server ein UDP-Paket eingeht, prüft das Betriebssystem den entsprechenden Port auf lauschende Applikationen. Wenn keine Anwendungen gefunden werden, muss der Server den Absender des Pakets darüber informieren. Dafür sendet er ein „Destination Unreachable“ ICMP Paket an den Absender. [11]



Quelle: https://www.ionos.de/digitalguide/fileadmin/DigitalGuide/Schaubilder/udp-flood.png

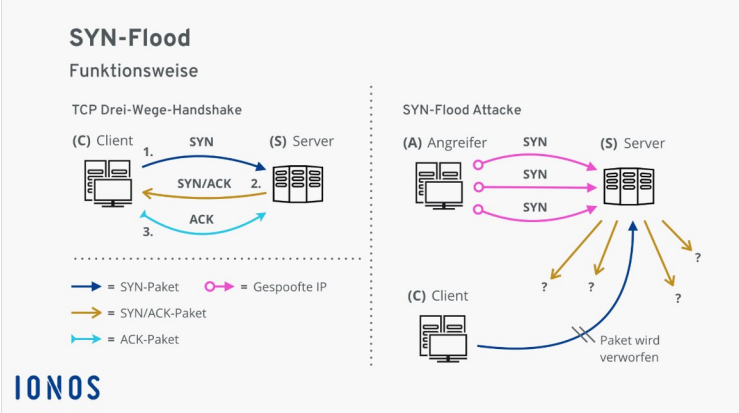
3.3 SYN Flood

SYN Flood ist eine sog. Protokoll-Attacke. Es zielt darauf ab, Schwachstellen des TCP-Protokolls zu missbrauchen, um sein Ziel überlasten zu können, sodass dieser keine legitimen Anfragen mehr beantworten kann. SYN Flood missbraucht den Mechanismus des drei Wege Handschake des Transmission Control Protokoll (TCP). Da es sich bei TCP um ein Verbindungsorientiertes Protokoll handelt, müssen Client und Server zunächst eine Verbindung aushandeln. Dieser Handshake läuft folgendermaßen ab:

1. Der Client sendet ein SYN-Paket(„synchonise“) an den Server.
2. Der Server antwortet mit einem SYN/ACK(„Acknowledge“) und legt eine Datenstruktur, Transmission Protocol Block(TCB) genannt, für die Verbindung im SYN Backlog an.
3. Der Client beantwortet dieses SYN/ACK-Packet mit einem ACK-Paket. Damit wird der Handshake abgeschlossen.

11 https://www.ionos.de/digitalguide/server/sicherheit/udp-flood/

Bei dem SYN Flood Angriff senden die Bots SYN-Pakete über eine gespoofte IP-Adresse an den Server. Dieser legt eine TCB Datenstrucktur für die halboffene Verbindung im Backlog an und belegt somit Speicher auf dem Server. Der Server sendet nun ein SYN/ACK-Paket zurück. Vom Angreifer bekommt der Server aber keine Bestätigung, da es sich um eine gefälschte IP-Adresse handelt, und sendet weitere Pakete, während er die Verbindung halboffen hält. Der Server wartet weiter auf eine Antwort, währenddessen gehen aber bereits weitere SYN-Pakete des Botnetzes ein, welche ebenfalls in das Backlog eingetragen werden müssen. Irgendwann hat der Server keinen Platz im Backlog mehr und verwirft neue, eingehende SYN-Pakete. Damit ist er von außen nicht mehr erreichbar.[12]

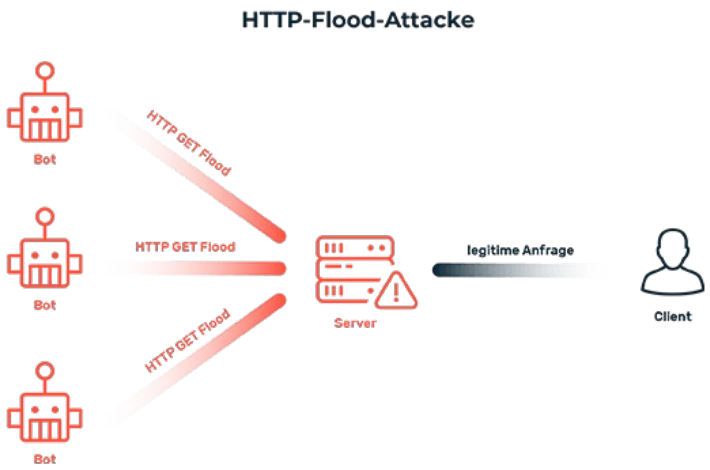


Quelle: https://www.ionos.de/digitalguide/fileadmin/DigitalGuide/Schaubilder/syn-flood-funktionsweise.png

3.4 HTTP Flood

Bei einem HTTP-Flood-Angriff senden Angreifer HTTP Anfragen an einen Webserver, welche gezielt Seiten mit großem Ladevolumen aufrufen. Durch die riesige Anfragenkapazität, welche Botnetze erreichen können, wird der Webserver überlastet und kann keine legitimen Anfragen mehr verarbeiten, wodurch der Webserver oder die Web-Applikation nicht mehr erreichbar ist. Diese Angriffsweise gehört zu der häufigsten Form der DDOS-Angriffe. Es wurden schon hTTP-Flood-Angriffe beobachtet, bei welchen die Zahl der böswilligen Anfragen bis in den mittleren dreistelligen Millionen Bereich ging.[13]

12 https://www.ionos.de/digitalguide/server/sicherheit/syn-flood/ zugriff 06.11.2022
13 https://www.myrasecurity.com/de/http-flood-attacke/ Zugriff: 6.11.2022



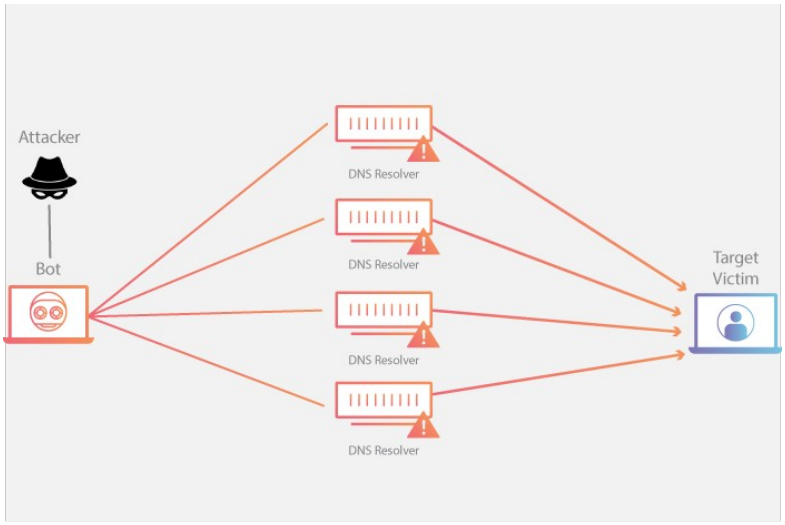
Quelle: https://www.myrasecurity.com/app/uploads/2021/09/Http_Flood_Attack_de.png

3.5 DNS Amplification Attack

Diese Art des DDOS Angriffs ist ein volumetrischer, verteilter Denial-of-Service Angriff, welcher die Funktionalität von offenen DNS-Resolvern benutzt, um so einen Zielserver mit einer verstärkten Traffic Menge zu überfluten, was in einem Ausfall der Funktionsfähigkeit des Servers und im Schlimmsten Falle in einem Ausfall des Systems resultiert. Die IP-Adresse der Bots wird bei diesem Angriff gespooft, sodass sie für den DNS Resolver wie die IP Adresse des Opfers aussieht. Anschließend stellen die Bots mit gespoofter IP-Adresse Anfragen an den DNS-Resolver. Diese Anfragen sind so strukturiert, dass die Antwort des DNS-Resolvers so groß wie möglich ausfällt. Der Angriff kann in vier Schritt unterteilt werden:

1. Über einen kompromittierten Endpunkt sendet der Angreifer UDP-Pakete mit gespoofter IP-Adresse an den DNS-Resolver.
2. Die Anfragen an den DNS-Resolver sind darauf ausgelegt, eine möglichst große Antwort zu erzielen. Oft werden dabei Argumente wie „ANY“ benutzt, um die sehr große Antwortpakete zu erhalten.
3. Nachdem er die Anfrage erhalten hat, sendet der DNS-Resolver eine große Antwort an die gespoofte IP-Adresse aka. Die IP-Adresse des Opfers
4. Das Opfer empfängt die Antwort und seine Netzwerkinfrastruktur wird mit Traffic überschwemmt was einen Denial-Of-Serivice verursacht und das Netzwerk nicht mehr erreichbar ist bzw. komplett ausfällt.[14]

14 https://www.cloudflare.com/de-de/learning/ddos/dns-amplification-ddos-attack/ Zugriff 6.11.2022



Quelle: https://cf-assets.www.cloudflare.com/slt3lc6tev37/2JmKP07Mi6jYbACILN84VI/9a91d91ecc1f414aa89ae001dbfce393/Learning_Center_DDoS_Diagrams_clean.png

4 Ziele eines Botnetz-Angriffs

…warum werden Angriffe durchgeführt...

4.1 Systemausfall

...welchen Zweck verfolgen die oben genannten Angriffe mit dem erzeigten Systemausfall...

4.2 Spam

Auch können die Bots in einem Botnetz benutzt werden, um Spam Nachrichten zu verschicken. Sie können sich gefälschte Konten in Foren oder Social Media Pattformen anlegen und hier automatisiert verschiedene Nachrichten wie z.B. Phishing Mails versenden. Da für die Erstellung eines Benutzerkontos nur sehr wenige Felder ausgefüllt werden müssen (Name, E-Mail-Adresse etc.) können die Angreifer die Bots sehr leicht dazu Programmieren, diese Formulare automatisch auszufüllen. Die Bots werden zudem genutzt, um Webseiten zu scannen und diese nach gültigen E-Mail-Adressen zu durchsuchen. Diese validen E-Mail-Adressen werden in eine Datenbank eingetragen, welche das Botnetz benutzt, um Mails an seine Opfer zu senden.[15]

Senden von Falschnachrichten

Gerade in der Heutigen Zeit, in welcher wir unsere Meinung vermehrt über Social Media kundgeben, stellen Botnetze, welche gezielt auf diesen Plattformen agieren, eine immer größere Gefahr dar. Die Botnetze werden dazu genutzt, falsche Meinungen und Bewertungen abzugeben, und so beispielsweise politische Kampagnen zu untergraben oder den Ruf von verschiedenen Firmen zu beschädigen. Eine Studie der University of California hat gezeigt, das 15% alle heute

15 https://www.cloudflare.com/de-de/learning/bots/what-is-a-spambot/ Zugriff 6.11.2022

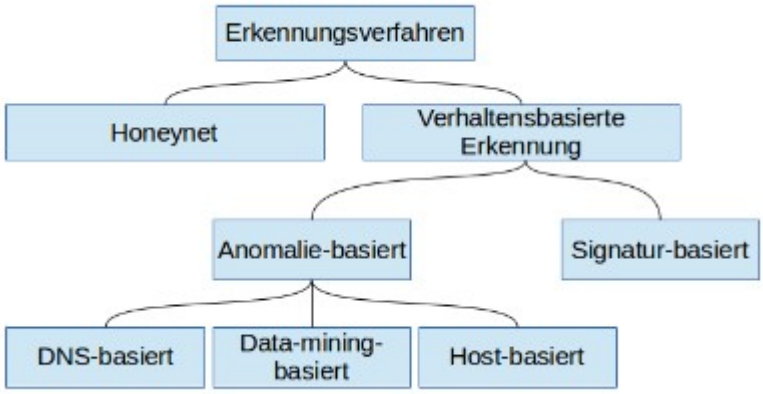
aktiven Twitter Konten von Bots gesteuert werden.[16] Durch diese gewaltige Zahl wird die Macht klar, welche die Betreiber der Botnetze haben, Meinungen online zu beeinflussen. Je weiter die Technologie voranschreitet, desto schwieriger wird es, die Bots von legitimen Usern zu unterscheiden. Deshalb ist es umso wichtiger, zuverlässigere und bessere Methoden zu entwickeln, um gegen die Botnetze vorzugehen.

4.3 Identitätsdiebstahl

...was geuna passiert bei Identitätsdiebstahl...

5 Botnetz Erkennungsmethoden

Dieses Kapitel soll sich mit der Erkennung von Botnetzen und deren Aktivitäten anhand verschiedener Indizien beschäftigen. Je nach Architektur und Kommunikationsinfrastruktur gibt es verschiedene Ansätze und Methoden zur Erkennung. Im Allgemeinen lassen sich die verschiedenen Methoden in aktive und passive Verfahren unterteilen. Passive Verfahren sind im Allgemeinen zu bevorzugen, da aktive Verfahren dem Botmaster die Möglichkeit geben zu erfahren, dass sein Botnetz untersucht wird, wodurch er seine Konfiguration ändern und sich besser absichern kann: Die passiven Erkennungsmethoden lassen sich dagegen nicht vom Botmaster erkennen.



Quelle: https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2014-08-1/NET-2014-08-1_03.pdf

5.1 Honeynet

Ein Honeynet ist ein Netzwerk, welches mehrere Honeypots enthält. Honeypots sind Computer oder Geräte, welche selbst keine produktiven Funktionen erfüllen. Ihr Zweck ist es, als Falle in Netzwerke eingebaut zu werden, um Angreifer anzulocken. Dafür greifen sie oft auf veraltete Software zurück, welche viele Sicherheitslücken besitzt, um sich für den Angreifer attraktiver zu machen. Da der Honeypot keine Funktion erfüllt, kann jeder Netzwerkverkehr in Verbindung mit dem Honeypot auf einen möglichen Angriff hindeuten. Eingehender Netzwerkverkehr könnte somit beispielsweise ein aktiver Port Scan sein. Sollte Netzwerkverkehr vom Honeypot ausgehen, so muss

16 Social Bots – Gefahren und Lösungen (entwickler.de)

dies bedeuten, dass er von Schadsoftware befallen wurde, da er selbstständig keinen Netzwerkverkehr erzeugt. Aus diesen Informationen lassen sich Schüsse auf das Botnetz und wie es kommuniziert ziehen und es können beispielsweise mögliche Schwachstellen im Botnetz entdeckt werden. Des Weiteren können somit die Werkzeuge des Angreifers analysiert werden.
Verhaltensbasierte Erkennungsmethode
Honeynets können nur bis zu einem gewissen Grad bei der Erkennung von Botnetzen helfen. Sollte der Computer zwar befallen worden sein, jedoch nicht versuchen andere Rechner zu infizieren, so bleibt er für das Honeynet unentdeckt, da somit kein Analysierbarer Traffic von ihm ausgeht. Hierfür kommen die Ansätze der verhaltensbasierten Erkennungsmethoden ins Spiel, welche versuchen, durch unterschiedliche Methoden die Strategien und Verhaltensweisen der Bots zu erkennen. Dies geschieht meistens durch die Analyse und Beobachtung des Netzwerkverkehrs. Dieser Ansatz lässt sich in Anomalie-basierte Erkennungsmethoden und Signaturbasierte Erkennungsmethoden unterteilen.

5.2 Anomalie-basierte Erkennungsmethode

Bei dieser Methode wird das Verhalten der Computer auf Anomalien, also auf Auffälligkeiten, welche sich von der Norm abheben, untersucht. Indizien können beispielsweise hohes Netzwerkvolumen oder Verwendung unüblicher Ports sein. Dieser Ansatz lässt sich noch einmal in 3 Unterkategorien gliedern.

5.2.1 DNS-Basierte Erkennungsmethode

Hierbei wird vorwiegend der durch das **D**omain **N**ame **S**ystem erzeugte Netzwerkverkehr untersucht. Das DNS wird von den Botnetzen benutzt, um mit den Command-and-Control-Servern eine Kommunikationsverbindung aufzubauen. Durch DNS müssen IP-Adressen nicht statisch im Code des Bots stehen, sondern können dynamisch aufgelöst werden. DNS-Verkehr bietet eine gute Analysegrundlage, da er in der Regel nicht sehr groß ist und sich somit so gut wie in Echtzeit scannen lässt. Ist bekannt, dass sich hinter einer Domain ein Botnetz Betreiber befindet, lässt sich diese Domain in der Regel sehr einfach sperren. Da dies den Botnetz Betreibern auch bekannt ist, nutzen diese sehr oft einen Domainname-Generation-Algorithm (DGA), welcher verwendet wird, um dynamisch neue Domains und Subdomains zu erzeugen.

5.2.2 Data-Mining-basierte Erkennungsmethode

Dieser Ansatz basiert darauf, aus gegebenen Daten neue Erkenntnisse zu sammeln. Es werden Data-Mining Strategien angewandt, um den Netzwerktraffic klassifizieren und gruppieren zu können. Die Daten sollen in zwei Kategorien unterteilt werden, um den normalen Traffic von dem, welcher vom Botnetz generiert wurde, unterschieden zu können und dadurch die infizierten Computer zu erkennen. Es werden hierbei Algorithmen für maschinelles Lernen angewandt, welche dann zum

Beispiel auch für die Erzeugung von Signaturen verwendet werden können.

5.2.3 Host-basierte Erkennungsmethode

Bei dieser Methode wird nicht das Netzwerk überwacht, um die Botnetz Aktivitäten zu erkennen, sondern die Computer selbst. Wenn der Computer von dem Botnetz infiziert wurde, lässt sich dies oft an ungewöhnlichen Systemaufrufen erkennen. So kann es sein, dass zum Beispiel der Virenscanner plötzlich deaktiviert wird, da die Schadsoftware versucht sich zu verstecken oder es werden ungewöhnliche Verbindungen zu Zielen im Internet aufgebaut. All das kann auf eine mögliche Infizierung hinweisen.

6 Simulation eines Botnetzes

Der Botcode und damit die Software zur unbemerkten Steuerung eines Hosts durch ein außenstehendes System, ist Schadcode. Ohne den Zugriff auf eine Anzahl von Rechnen, die mit Schadcode infiziert werden können, ist die Suche nach einer Alternativmöglichkeit nötig. Virtuelle Maschinen bilden das Abbild eines ganzen Systems, weshalb deren Verwendung einen leistungsfähigen Host verlangt. Mit Hilfe einer Simulation ist eine Vielzahl an Maschinen darzustellen, während die Ressourcen der simulierenden Maschine geschont bleiben. Dies ist der Grund hinter der Herangehensweise des folgenden Kapitels.

Das Kapitel handelt von der Auswahl einer geeigneten Netzwerk Simulations-Software und der Umsetzung verschiedener Szenarien der Botnetkomunikation.

6.1 Verwendete Tools

Dieser Abschnitt beschäftigt sich mit verschiedener Netzwerk Simulations-Software. Jedes Tool verfügt über charakterisierende Eigenschaften. Eine Gegenüberstellung lässt Schlüsse darauf zu, welches Simulations-Programm am besten für die Simulation eines Botnetzverhaltens geeignet ist.

6.1.1 NS2

Network Simulator 2 (NS2) ist ein Open-Source Projekt, erstmals erschienen im Jahr 1989. NS2 wurde entwickelt in der Univertity of California und Cornell University (ns2-wiki (2016)) und ist nun kostenfrei erhältlich. Dabei dient es zur Simulation reeller Komponenten in einem Netzwerk und war damit ein zentrales Tool, welcher Fortschritte in diesem Gebiet ermöglichte (Issariyakul, T et. al. (2012)). Trotz der Verwendung von C++, was der Langlebigkeit zu tut, im Quellcode ist NS2 nicht mehr auf dem neusten stand der Technik.

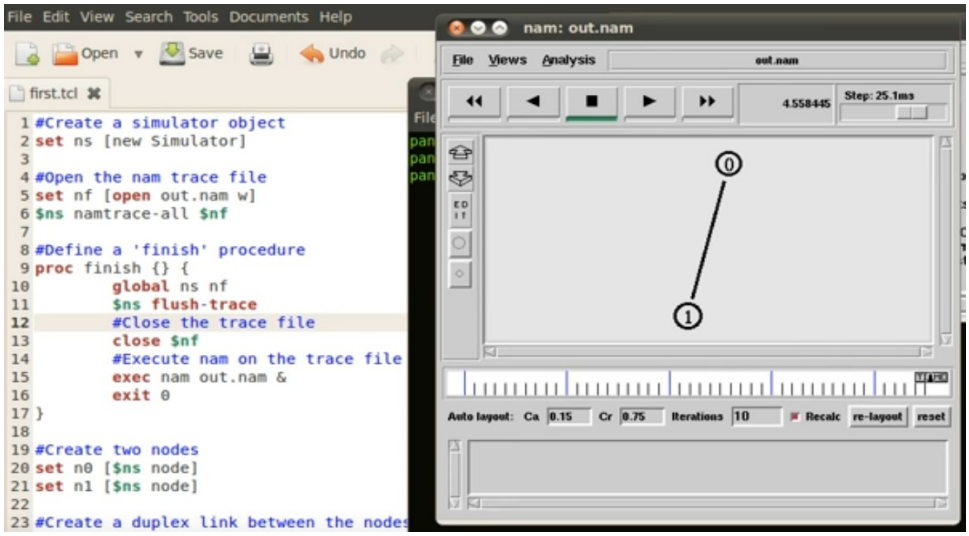


Figure 1: NS2 - Quelle: HowTo (2015)

Die Sprache Tcl wird verwendet, um ein zu simulierendes Netzwerk zu definieren und die verschiedenen Szenarien zu beschreiben (ns2-wiki (2016)). Nach dem Ausführen einer fertigen .tcl-Datei, ist die Simulation abgeschlossen und NS2 öffnet ein Graphical User Interface (GUI). Darauf zu erkennen sind die definierten Komponenten, als Knoten, und die möglichen Kommunikationswege als Kanten.

6.1.2 NS3

Network Simulator 3 (NS3) bildet, ab dem Jahr 2008, den Nachfolger zum NS2 und wurde bis zum Schreiben dieser Arbeit auf dem neusten Stand gehalten (nasnam (2022)). Ebenso wie dessen Vorgänger ist auch NS3 open-source unter der Entwicklung des „ns-3 Porjects“. Wie der eigene Quellcode werden auch die dazugehörigen Simulationen in C++ geschrieben. Ausnahmen bilden dabei Python-Abschnitte die in den C-Code eingebunden werden (Nayyar, A. et. al. (2015)). Neben der neuen Definitionssprache für die Netzwerken, besteht der Vorteil von NS3 in der Abwärtskompatibilität zu NS2 wodurch bestehender Code weiter verwendet werden kann.

NS3 bietet eine Auswahl an verschiedenen Komponenten für den Einsatz im Netzwerk. Diese sind als Objekte anzusprechen, auf denen Funktionen ausgeführt werden. Nach abgeschlossener Kompilierung der .cc-Datei wird das Szenario simuliert und die Ergebnisse an eine im Code festgelegte Stelle geloggt. NS3 verfügt selbst über keine Möglichkeit zur Visualisierung. Hierfür lässt sich beispielsweise die Open-Source-Anwendung NetAnim einsetzen (Nayyar, A. et. al. (2015)).

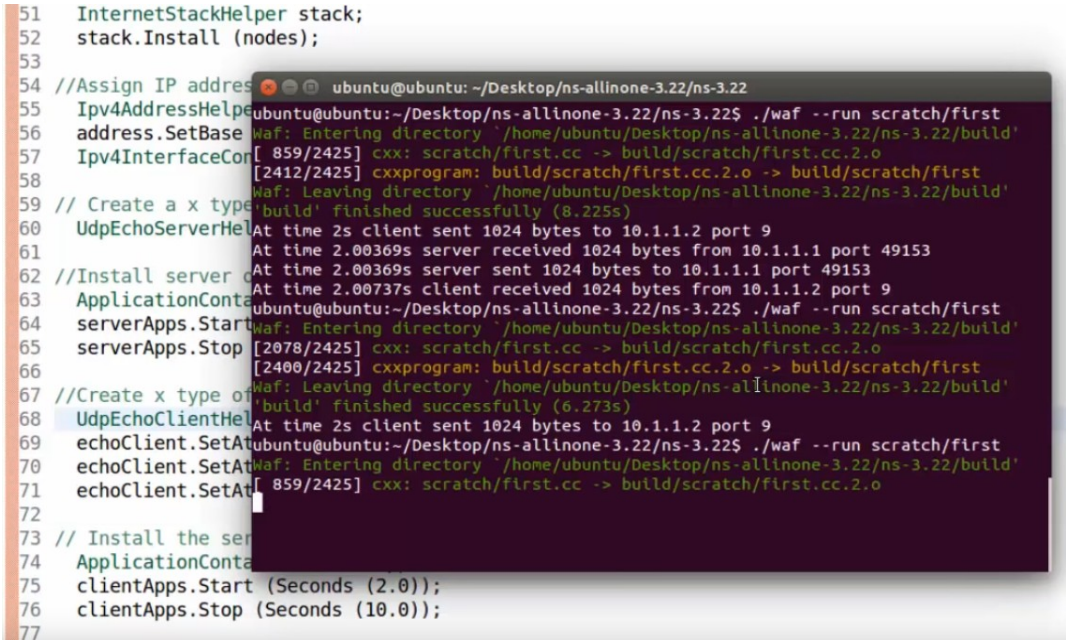


Figure 2: NS3 - Quelle: Hitesh Choudhary (2015)

6.1.3 Netsim

Netsim ist ein kostenpflichtiges Tool zur Simulation von Netzwerkverhalten. Veröffentlicht im Jahr 2005 wird es nun von zwei Firmen angeboten, Boson (Nayyar, A. et. al. (2015)) und Tetcos. Zum Zeitpunkt dieser Arbeit erhält Netsim weiterhin regelmäßig Updates. Boson bewirbt die Software in erster Linie als Lernprogramm. Zur Vorbereitung auf CCNA-Prüfungen beinhaltet das Programm ein GUI, um Netzwerke per „Drag and Drop“ zu definieren und das Komponentenverhalten zu konfigurieren (boson (2022)). Die Visualisierung der Knoten und Paketbewegungen über die Kanten findet direkt in der selben GUI statt.

Das Verhalten sowie die Einstellungsbefehle entsprechen den Cisco-Produkten. Das Durchlaufen und Durcharbeiten verschiedener „Labs“ übt den Umgang mit den Produkten. Die Labs umfassen sowohl die Aufgabenstellungen als auch eine Schritt für Schrittanleitung zu Erfüllung des Ziels.

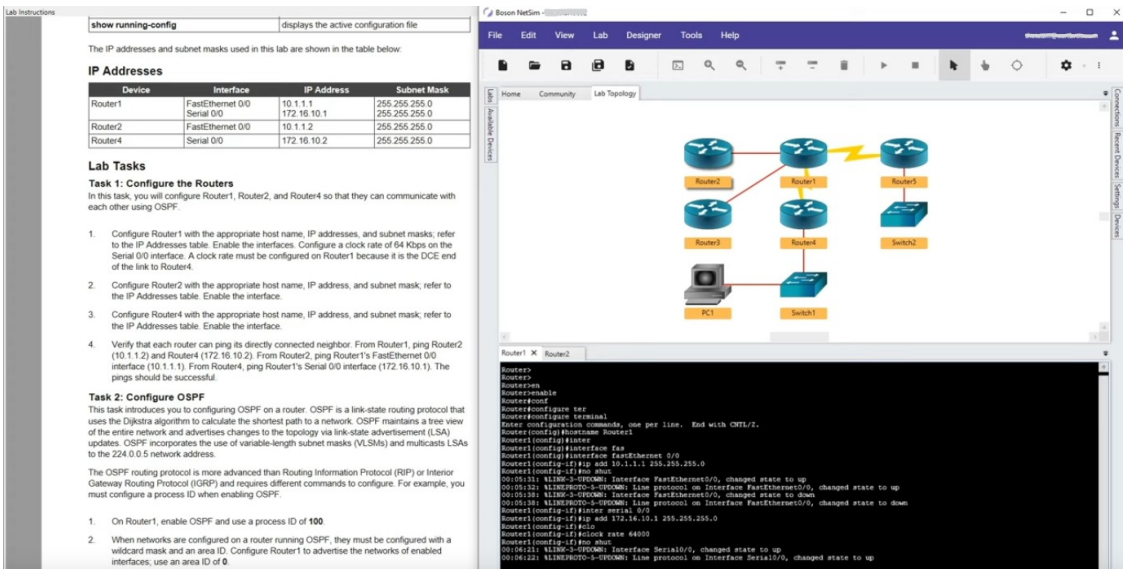
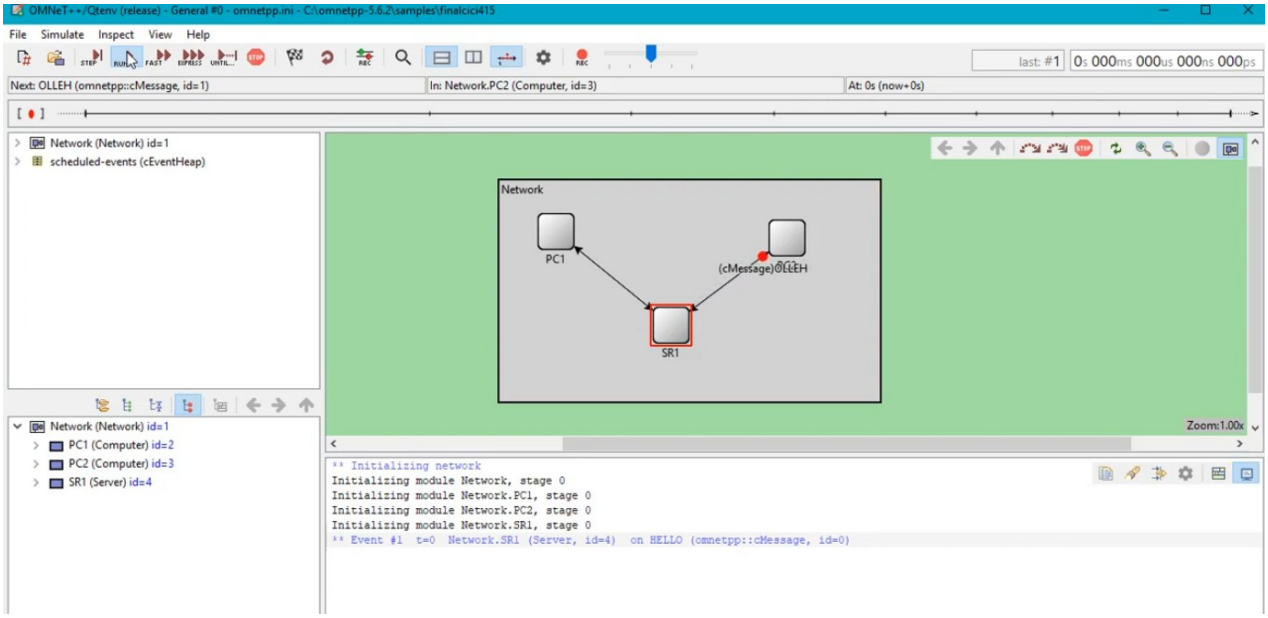


Figure 3: Netsim - Quelle: CertBros (2020)

6.1.4 Omnet++

Omnet++ ist ein kostenloses Simulations-Tool, dass 2001, von OpenSim Ltd.,in der Version 2.0 veröffentlicht wurde. Bis zur Anfertigung dieser Arbeit, wurde die Software regelmäßig mit Updates versorgt (omnetpp (2022)). Der Quellcode der Software ist in C++ geschrieben. Die Omnet++ Projekte sind jedoch nicht auf C++ beschränkt, da sowohl Java als auch C# akzeptierte Sprachen sind (Nayyar, A. et. al. (2015)). Wie bei Netsim lassen sich die Netzwerke aus einzelnen Komponenten oder Komplexeren Modulen in einem GUI miteinander verbinden. Omnet++ verwendet hierzu eine .ned-Datei, die ebenfalls in Codeform bearbeitbar ist.



Ein zentraler Vorteil gegenüber den anderen Software-Lösungen liegt in der Erstellung eigener Module. Mittels Quellcode lassen sich eigene Module und Pakete erstellen, deren Verhalten genau zu definieren ist.

Die in der .ned-Datei Beschriebenen Netzwerke erhalten bei der Simulation eines Szenarios weitere Parameter durch eine .ini-Datei, woraufhin ein gesondertes Fenster die Animierte Simulation wiedergibt.

6.2 Botnet Simulation Framework für Omnet++

Dieser Abschnitt geht auf das Botnet Simulation Framework (BSF) für Omnet++ generell ein und darauf Beispiel Szenarien welche das Framework erlaubt darzustellen.

Das BSF erweitert Omnet++ um Komponenten zur Simulation eines peer-to-peer Botnetzes. Es liegt näher an einer Bibliothek als an. BSF ist ein Open-Source Projekt und beinhaltet eine Sammlung von Modulen, welche das Verhalten von in Botnetzen Vorhanden Komponenten widerspiegeln.

6.2.1 Szenario 1 (Titel)

6.2.2 Szenario 2 (Titel)

6.2.3 Szenario 3 (Titel)

7 Analyse des Botnetzes Mirai

…wie hat Mirai in der Vergangenheit Aufmerksamkeit erregt und was macht es aus…

8 Abwehrmaßnahmen mit entsprechender Software

…Vorstellung der Software im allgemeinen, welche Schritte sind mit der Software auszuführen, was sind die Theoretischen Vorteile hinter den eingesetzten Maßnahmen…

9 Fazit

...Zusammenfassende Worte...

9.1 Erkenntnisse

…welche Informationen wurden speziell gewonnen und haben Mehrwert…

9.2 Aussicht

…auf was bzw. welche Entwicklungen lässt sich nach dieser Arbeit für die Zukunft schließen...

Quellenverzeichnis

1. Issariyakul, T., Hossain, E. (2012): Introduction to Network Simulator 2 (NS2), Boston, MA. Springer
2. ns2-wiki (2016): User Information – nsnam, [online] https://nsnam.sourceforge.net/wiki/index.php/User_Information [abgefrufen 27.12.2022]
3. Kabir, M. H., Islam, S., Hossain, M. J., & Hossain, S. (2014): Detail Comparison of Network Simulators, ResearchGate, [online] https://www.researchgate.net/publication/275654046_Detail_Comparison_of_Network_Simulators [aufgerufen 27.12.2022]
4. nasnam (2022): News & Events, ns-3, [online] https://www.nsnam.org/about/news-events [aufgerufen 27.12.2022]
5. boson (2022): NetSim Network Simulator & Router Simulator, [online] https://www.boson.com/netsim-cisco-network-simulator [aufgerufen 27.12.2022]
6. omnetpp (2022): Archived News, [online] https://omnetpp.org/posts [aufgerufen 27.12.2022]
7. Nayyar, A., Singh, R. (2015): A Comprehensive Review of Simulation Tools for Wireless Sensor Networks (WSNs), ResearchGate, [online] https://www.researchgate.net/publication/332819279_A_Comprehensive_Review_of_Simulation_Tools_for_Wireless_Sensor_Networks_WSNs [aufgerufen 28.12.2022]
8. HowTo (2015): How to run first simulation on Ns2, [YouTube-Video] https://youtu.be/4YBNBYbC09E?t=98 [aufgerufen 28.12.2022]
9. Hitesh Choudhary (2015): explaining first.cc file in ns3, [YouTube-Video] https://youtu.be/3n0M-7-6IMM?t=744 [aufgerufen 28.12.2022]
10. CertBros (2020): Is NetSim Worth it? | Boson NetSim for CCNA, [YouTube-Video] https://youtu.be/CLldIsyOCHc?t=356 [aufgerufen 28.12.2022]
11. Hasan AI-Sayyed (2021): OMnet++ Simple Project C++, [YouTube-Video] https://youtu.be/ntxgcxEcvxM?t=833 [aufgerufen 28.12.2022]