

Rishi – Identifizierung von Bots durch Auswerten von IRC Nicknamen

Jan Göbel
RWTH Aachen
Rechen- und Kommunikationszentrum
goebel@rz.rwth-aachen.de

Thorsten Holz
Universität Mannheim
Lehrstuhl für verlässlich verteilte Systeme
thorsten.holz@informatik.uni-mannheim.de

Zusammenfassung

In dieser Arbeit stellen wir eine präzise, aber dennoch einfache Methode zur Erkennung von Bot infizierten Rechnern in einem Netzwerk vor. Ziel ist es, den Kommunikationskanal zwischen Bot und Command & Control Server aufzudecken. Die hier vorgestellten Techniken basieren auf der Analyse von Netzwerkverkehr zur Aufdeckung von ungewöhnlichen IRC Nicknamen, IRC Servern oder Server Ports. Mit Hilfe der N-Gram Analyse und einem Punktesystem sind wir in der Lage, auch Bot infizierte Rechner zu erkennen, die von klassischen Intrusion Detection Systemen gar nicht, oder erst wesentlich später erkannt werden.

1 Einleitung und Motivation

Die Tatsache, dass inzwischen auch privat genutzte Rechner wertvolle Informationen und Ressourcen, zum Beispiel in Form von Daten für Online Banking oder Rechenleistung, zu bieten haben, machen sie zu einem immer interessanter werdenden Ziel für Kriminelle. In diesem Zusammenhang sind nicht ausschließlich sensitive Daten von Bedeutung, sondern auch die zur Verfügung stehende Bandbreite, die zum Versenden von Spam oder dem Durchführen von Denial of Service (DoS) Attacken eingesetzt werden kann. Darüber hinaus hat der Einsatz von sich automatisch verbreitender bössartiger Software (*Malware*) das Aufspüren und Kompromittieren von anfälligen Computern stark vereinfacht. Angriffe beschränken sich daher meist nicht mehr auf wenige ausgewählte Rechner, sondern zielen auf mehrere tausend in sehr kurzer Zeit. Diese infizierten Rechner werden oft zu

sogenannten *Botnetzen* zusammengeschlossen, um so zum Beispiel die Effizienz von verteilten DoS Angriffen zu steigern.

Deshalb verlangt die derzeitige Situation nach mehr proaktiven und insbesondere vorbeugenden Maßnahmen, um infizierte oder kompromittierte Rechner daran zu hindern, weitere Computer im Netzwerk zu attackieren und zu übernehmen. Eine der gängigsten Methoden, um Botnetze unschädlich zu machen, ist das Unterbrechen des Kommunikationskanals durch Abschalten des Command & Control (C&C) Servers oder das Ändern des entsprechenden DNS Eintrags. Dadurch können die einzelnen Bots keine Verbindung mehr zu ihrem zentralen Server aufbauen und somit auch keine weiteren Kommandos empfangen und ausführen. Der Nachteil dieser Vorgehensweise ist, dass die Rechner weiterhin infiziert und in vielen Fällen mit einer Backdoor versehen sind, die es dem Angreifer erlaubt, die Maschinen jederzeit wieder unter seine Kontrolle zu bringen. Aus diesem Grund ist es notwendig einen Weg zu finden, um nicht nur den Kommunikationskanal zu deaktivieren, sondern auch die infizierten Rechner zu identifizieren um die verantwortlichen Personen darüber zu informieren.

In dieser Arbeit stellen wir eine präzise aber durchaus einfache Methode zur Erkennung von IRC Bot infizierten Computern vor. Die Vorgehensweise beruht auf der Tatsache, dass ein Bot, kurz nach der Infektion, zunächst Kontakt zu seinem Command & Control Server aufnimmt. Dieser Kommunikationskanal weist einige eindeutige Eigenschaften auf (z.B. die Ähnlichkeit der verwendeten Nicknamen und/oder bestimmte Teilzeichenfolgen), die zur Erkennung ausgenutzt werden können.

2 Ähnliche Arbeiten

Eine der frühesten Arbeiten, die einen ähnlichen Ansatz verfolgt, ist von Kristoff [Kri04]. In seiner Präsentation erwähnt er, dass verdächtige Nicknamen, Topics oder auch Channel deutliche Anzeichen für einen „wilden“ IRC Server sein können. Wir haben diese Idee weitergeführt und überprüfen, ob die Ähnlichkeit von Nicknamen, die von Bots benutzt werden, zur Erkennung einer infizierten Maschine verwendet werden können oder nicht.

Botnetze verwenden in der Regel den selben IRC Channel. Diese Beobachtung machen sich Binkly und Singh [BS06] zu nutze, um verdächtige IRC Server zu identifizieren. Sie verbinden in ihrer Arbeit TCP-basierte Anomalie Erkennung mit einer IRC Protokoll Analyse, um auf effiziente Weise Botnetze zu erkennen. Das System verwendet Daten, die über einen ganzen Tag verteilt gesammelt worden sind. Im Gegensatz dazu arbeitet unsere Methode in Echtzeit und kann so infizierte Rechner erkennen, noch bevor sie in unserem Intrusion Detection System auffallen.

Chen [Che06] versucht Botnetze an Randpunkten des Netzwerks, wie zum Beispiel Routern oder Gatesways, zu erkennen. Chen präsentiert vorläufige Statistiken über die mittlere Paketlänge und die Verteilung von IRC Nachrichten, wie zum Beispiel JOIN oder PING/PONG. Allerdings liegen keine Statistiken über die Erfolgsrate seiner Methode vor. Der von Strayer [SWLL06] verfolgte Ansatz beruht auf der Untersuchung von verschiedenen Flusseigenschaften, wie zum Beispiel verwendete Bandbreite und zeitliche Aspekte von Netzerkpaketen.

Livadas [LWLS06] verwendet Techniken des maschinellen Lernens (künstliche Intelligenz) um den C&C Verkehr von IRC-basierten Bots zu erkennen. Da dieser Methode eine völlig andere Vorgehensweise zu Grunde liegt, ist eine Kombination mit unseres grundlegend möglich. Wir verwenden Eigenschaften des IRC Protokolls und messen die Ähnlichkeit zu bekannten Botnetzen. Die Analyse der Eigenschaften des Kommunikationskanals könnte durch Livadas erfolgen.

Ein weiterer Ansatz zur Erkennung von Bot infizierten Rechnern ist die sogenannte „Verhaltensbasierte Erkennung“. Bots besitzen eine relative hohe Leerlaufzeit, da sie die meiste Zeit mit dem Warten auf Kommandos verbringen. Des Weiteren reagieren Bots deutlich schneller auf Eingaben als ein Mensch es tun würde. Racine [Rac04] hat ein System vorgestellt, welches diese Kenndaten identifizieren kann. Allerdings wäre die Zahl der Fehlalarme relativ hoch. Ein weiterer verhaltensbasierter Ansatz wird von Stinson und Mitchell [SM07] verfolgt. Hier werden Programme auf einem Rechner dahingehend analysiert, ob ihr Verhalten dem eines Bots ähnelt.

3 Erkennung des Kommunikationskanals

Bots bilden heutzutage eine der grössten Bedrohungen im Internet, was die Entwicklung von effiziente Methoden zur Erkennung von infizierten Maschinen notwendig macht. Gängige Intrusion Detection Systeme sind hier nicht unbedingt ausreichend, da Kompromittierungen oft über E-Mail oder auch über sogenannte Drive-By-Downloads stattfinden und damit von diesen Systemen häufig nicht erkannt werden. Darüberhinaus zeigen Bots oftmals über längere Zeit keine nennenswerte Reaktion, sondern werden erst an bestimmten Tagen oder unter bestimmten Umständen aktiv.

Unser Ansatz zielt auf die Erkennung des Kommunikationskanals zwischen einem Bot und dem Command & Control Server. Dies ist, nach der Infektion, der frühest möglich Zeitpunkt, um infizierte Rechner zu erkennen.

3.1 Motivation

Allen Bots ist eine Eigenschaft gemein: Sie brauchen einen Kommunikationskanal, um Kommandos zu erhalten und Status-Information oder gesammelte Daten an den Botnetzbetreiber zurückzusenden. Dies stellt einen der Hauptunterschiede zwischen einem Wurm und einem Bot dar: Beide Malware Arten verbreiten sich automatisiert, ohne Interaktion des Angreifers, allerdings besitzt ein Wurm keine Kontrollkomponente über die sich sein Verhalten steuern lässt.

Eine der häufigsten Methoden zur Kommunikation innerhalb von Botnetzen ist immernoch das Internet Relay Chat (IRC) Protokoll. Obwohl inzwischen Peer-to-Peer Netzwerke oder auch das HTTP-Protokoll für Botnetze benutzt werden, ist IRC immernoch – gerade wegen seiner etablierten Infrastruktur und der Einfachheit der Implementation – eines der beliebtesten Kommunikationsmittel für Bots. Die infizierten Rechner verbinden sich zu einem fest eingetragenen IRC Server, betreten einen bestimmten Chatraum (Channel) und erhalten dort ihre Instruktionen. Da unsere Erkennungsmethode auf dem Auswerten von IRC Nicknamen basiert, liegt der Fokus dieser Arbeit auf IRC-basierenden Bots. Es ist allerdings möglich, unsere Methode auch auf andere Protokolle auszuweiten, sofern bestimmte, wiederkehrende Eigenschaften in den Nachrichten zwischen Bots und Command & Control Server existieren. Dies trifft zum Beispiel für einige HTTP Bots zu, die eindeutige Teilzeichenketten in den verwendeten URLs benutzen.

Der grösste Nachteil in der Verwendung des IRC Protokolls besteht aus Sicht eines Botnetzbetreibers darin, dass er die komplette Gewalt über sein Botnetz verliert, wenn der zentrale Command & Control Server nicht mehr erreichbar ist. Aus diesem Grund ist es eine gängige Methode in der Botnetzbekämpfung, den IRC Server zu deaktivieren und damit weitere Kommandos an die Bots zu unterbinden. Der große Nachteil dieser Vorgehensweise ist, dass die Rechner weiterhin infiziert und verwundbar sind. In vielen

Fällen werden diese Rechner nur kurze Zeit später erneut kompromittiert und tauchen in einem anderen Botnetz wieder auf.

Unsere Methode erlaubt neben der Identifikation der Command & Control Server auch das Identifizieren von Rechnern, die versuchen mit diesem Kontakt aufzunehmen. Der verantwortliche Administrator kann im Anschluß über die Infektion informiert werden, die betroffenen Rechner „säubern“ und so verhindern, dass weitere vertrauliche Daten in die Hände Dritter fallen. Zusätzlich können wertvolle Informationen über den C&C Server gesammelt werden, wie zum Beispiel der Name des Chatraums. Mit Hilfe dieser Informationen lässt sich der Botnetz Server zunächst infiltrieren und beobachten, bevor man diesen deaktiviert.

3.2 Erkennungsprinzip

Die Verwendung eines standardisierten Protokolls wie IRC erlaubt eine einfache Erkennung von Computern, die sich im IRC Netzwerk bewegen [Kal00]. Allerdings existieren auch Bot-Varianten die abgewandelte Formen des IRC Protokolls verwenden und somit nicht zu erkennen sind. Mehr dazu in Abschnitt 4.

Eines der ersten Kommandos, die von einem Rechner abgesetzt werden, wenn er das IRC Netzwerk verwendet, ist „NICK“, gefolgt von dem Nicknamen (Synonym), unter dem der Rechner in den Chaträumen identifiziert werden soll. Hier verbirgt sich auch die Haupteigenschaft, die wir uns bei der Erkennung von Bot infizierten Rechnern zu Nutze machen: Es ist nicht erlaubt, dass zwei Computer unter dem gleichen Nicknamen im selben IRC Netzwerk eingeloggt sind. Da Botnetze meist aus mehreren tausend Rechnern bestehen, muss jeder einen unterschiedlichen Nicknamen verwenden. Anderenfalls scheitert er beim Verbindungsaufbau zum IRC Netzwerk.

Eine gängige Methode, die von Bots verwendet wird, um entsprechende Nicknamen zu generieren ist das Verwenden konstanter Wörter in Verbindung mit einer Zufallszahl. Der Wurm Rbot.210944 [Ant04] benutzt zum Beispiel Nicknamen, die folgendermassen aufgebaut sind: *Länderkürzel|neunstellige Nummer* (z.B.: USA|016887436 oder DEU|028509327). Es gibt auch den umgekehrten Fall, ein zufälliges Wort wird mit einer konstanten Zahl kombiniert. Der Wurm Korgo.F.var [Ant05] benutzt zum Beispiel „_13“ als konstanten Suffix (z.B.: bmdut_13).

Das Prinzip hinter unserem Ansatz ist also recht einfach: Ein Bot muss einen Nicknamen verwenden, der eine zufällig generierte Komponente besitzt, um zu vermeiden, dass er auf Grund von doppelt verwendeten Nicknamen das IRC Netzwerk nicht „betreten“ kann. Neben diesem zufällig erstellten Teil enthält der Nickname oft auch einen konstanten Teil, der zum Beispiel Information über die Art des Bots, das Betriebssystem oder den Aufenthaltsort des Rechners enthält. Als Aufenthaltsort sind hier die Länderkürzel gemeint, wie DEU für Deutschland oder CHN für China. Genau dieser konstante Teil des Nicknamen bildet, neben anderen Aspekten, die Grundlage unserer Erkennungsmethode.

3.3 Projekt Rishi

Eine der ersten Aktionen, die von einer frisch infizierten Maschine durchgeführt wird, ist die Verbindung zum Botnetzserver, um erste Kommandos zu erhalten. Dadurch ist es möglich, wenn man diese initiale Verbindung abhört, einen infizierten Rechner zu erkennen, noch bevor dieser bössartige Aktionen durchführt.

Aus diesem Grund, untersucht unsere Proof-of-Concept Implementierung *Rishi* TCP Pakete auf das Auftreten folgender IRC Kommandos: NICK, JOIN, USER, QUIT und MODE.

Die Parameter, die zusammen mit diesen Befehlen über die Leitung gehen, werden extrahiert und zur weiteren Analyse gespeichert. Die Analyse bezieht sich hauptsächlich auf den übermittelten Nicknamen. Zusätzlich wird die IP Adresse des IRC Servers auf bekannte C&C Server hin untersucht und überprüft, ob der Channel Name bereits in früheren Botverbindungen verwendet wurde. Die darüber hinaus gesammelten Informationen dienen mehr der Vollständigkeit und erleichtern zum Beispiel das Infiltrieren und Beobachten eines Botnetzes.

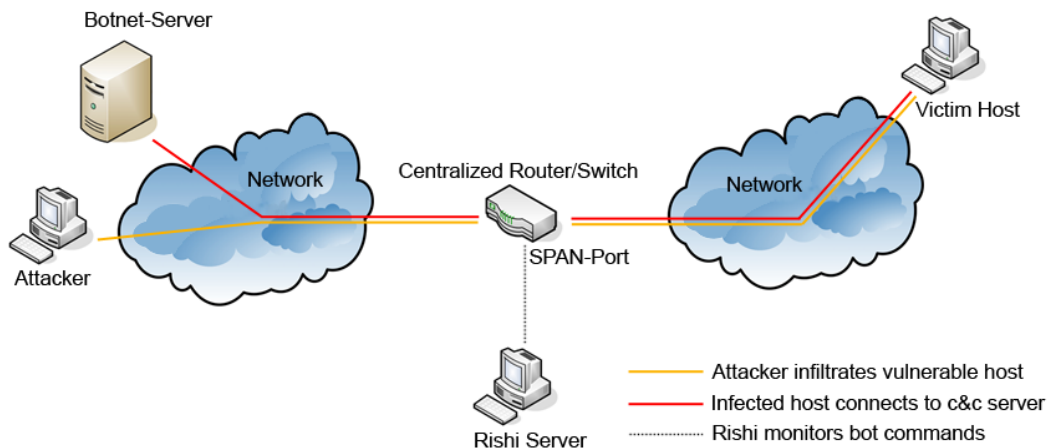


Abbildung 1: Rishi im Netzwerk

Abbildung 1 gibt einen schematischen Überblick über Rishi und zeigt, wo Rishi innerhalb eines Netzwerkes am Besten installiert wird. Zur Verdeutlichung ist auch der Ablauf einer Infektion mit anschließender Verbindung des infizierten Rechners zum entsprechenden Botnetserver dargestellt. Rishi lauscht hier nicht auf die übertragenen Kommandos, die der Bot ausführen soll, sondern auf die Verbindung zum IRC.

Rishi ist ein in Python geschriebenes Programm und bestehend aus ca. 2400 Zeilen Code. Die Netzwerkpakete werden wahlweise über eine Instanz von ngrep [Rit07] oder aber auch direkt über die Python-Bindings pcap [Koh07] gesammelt. Dadurch ist es uns möglich, den Netzwerkverkehr bereits vorzufiltern und nur die wichtigen Pakete zur Analyse durchzulassen.

Nachfolgend ist ausschnittsweise ein Aufruf von ngrep dargestellt, der zum Beispiel Netzwerkpakete herausfiltert, die IRC Protokoll Informationen enthalten:

```
ngrep [...] 'JOIN|NICK|MODE...' 'tcp[((tcp[12:1]&0xf0)
  >> 2):4] = 0x4e49434b and [...]'
```

Jedes der auf diese Weise gesammelten Netzwerkpakete wird von Rishi analysiert, wobei die folgenden Informationen (soweit vorhanden) extrahiert werden:

- Zeitstempel der Verbindung
- IP Adresse und Port des verbindenden Rechners
- IP Adresse und Port des IRC Servers
- Die Namen der Channels die besucht werden
- Verwendete Nicknamen
- Verwendeter Usermode

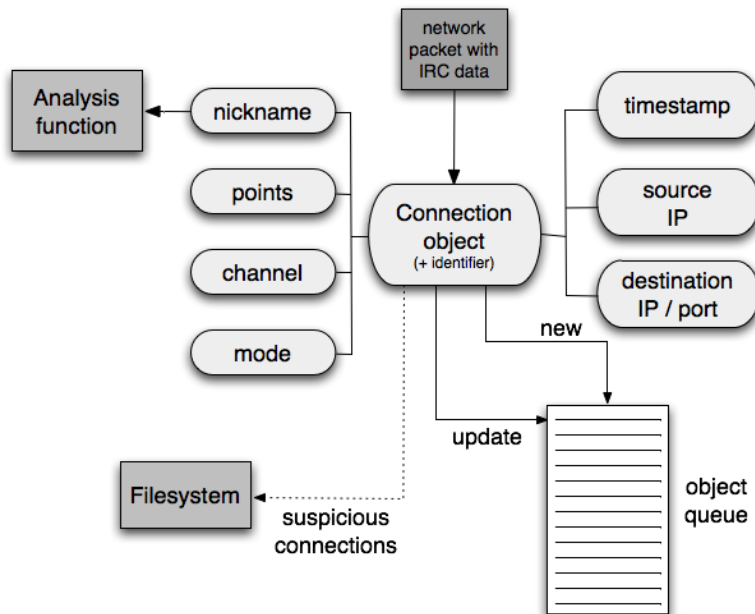


Abbildung 2: Basiskonzept von Rishi

- Weitere Channel Parameter, die mit dem Kommando `MODE` übergeben werden.

Für jede so entdeckte IRC Verbindung wird ein Verbindungsobjekt erzeugt, welches die oben aufgelisteten Informationen, zusammen mit einer eindeutigen Identifikationsnummer, speichert. Diese Identifikationsnummer wird aus dem Port und der IP Adresse des IRC Servers, sowie dem verbindenden Rechner gebildet. Dadurch ist es möglich, nachfolgende Pakete dem gleichen Verbindungsobjekt zuzuordnen und dieses zu aktualisieren. Das bedeutet, wenn ein Bot in einen anderen Channel wechselt, aber auf dem gleichen IRC Server bleibt, dann wird kein neues Verbindungsobjekt erzeugt, sondern lediglich das Bestehende aktualisiert. Um den Speicherbedarf von Rishi in Grenzen zu halten, werden die einzelnen Verbindungsobjekte in einer Queue gespeichert. Objekte, die aktualisiert werden, wandern wieder an den Anfang der Queue, so dass diese nicht so schnell gelöscht werden wie Objekte, die zu Verbindungen gehören, die bereits nicht mehr aktiv sind. Falls das IRC Kommando `QUIT` für ein existierendes Verbindungsobjekt empfangen wird, wird es direkt aus der Queue entfernt. Das Basiskonzept von Rishi wird in Abbildung 2 genauer dargestellt. Die Abbildung zeigt ein Verbindungsobjekt, die Informationen die von diesem gespeichert werden und die Queue, die alle aktuellen Verbindungsobjekte enthält. Auf die Analyse der Nicknamen und die Punktevergabe wird im folgenden Abschnitt eingegangen.

3.3.1 Punktevergabe

Sobald ein Verbindungsobjekt einen Nicknamen enthält, wird dieser an die Analysefunktion übergeben, um eine Punktzahl zu ermitteln. Diese Punktzahl gibt schliesslich Aufschluss darüber, ob es sich um eine Botverbindung handelt oder nicht. In der Analysefunktion wird ein Nickname auf verschiedene Kriterien hin überprüft, wie zum Beispiel verdächtige Teilzeichenketten, Sonderzeichen oder lange Zahlenketten. Für jeden erfolgreichen Test wird eine bestimmte Punktzahl auf den Gesamtwert des Nicknamens addiert.

Am Ende der Analyse wird die Punktzahl zusammen mit einer kurzen Zusammenfassung, welche Tests erfolgreich waren, zurück an das Verbindungsobjekt gegeben und dort gespeichert. Je höher die letztlich erreichte Punktzahl ist, desto wahrscheinlicher handelt es sich um eine Verbindung eines Bot infizierten Rechners. Wird ein festgelegter Schwellwert überschritten, löst Rishi einen Alarm aus. In diesem Fall wird das Verbindungsobjekt als mögliche Botkommunikation klassifiziert und in einer separaten Datei auf der Festplatte gespeichert.

Derzeit wird ein Schwellwert von 10 Punkten verwendet. Jede Verbindung mit mehr Punkten wird als Botverbindung eingestuft und der Nickname sowie der Channel Name werden der dynamischen Blackliste hinzugefügt. Falls 0 Punkte vergeben werden, so wird der Rechner als sauber klassifiziert und der Nickname auf die dynamische Whiteliste gesetzt.

Nachfolgend beschreiben wir die Evaluierungsfunktion genauer. Im ersten Schritt wird ein empfangener Nickname sowohl gegen eine vorkonfigurierte, als auch gegen eine dynamisch erstellte Blackliste geprüft. Erstere kann vom Anwender vor dem Start von Rishi frei konfiguriert werden. Die dynamische Liste ergänzt sich zur Laufzeit selbst, indem jeder erkannte Botnamen dieser Liste hinzugefügt wird. Nach dem direkten Vergleich mit den Blacklisten werden Ähnlichkeitsprüfungen auf Basis der N-Gram Analyse durchgeführt. Stimmt hier mehr als die Hälfte der Bi-Gramme überein, wird der Nickname als Bot klassifiziert. Auf die gleiche Weise werden Nicknamen auch gegen die vorkonfigurierte und dynamische Whiteliste geprüft. Neben diesen Tests prüfen wir zusätzlich, ob es sich bei dem Server Port um einen gängigen IRC Port handelt oder nicht. Viele Bots verwenden eher ungewöhnliche Ports für ihre C&C Server, zum Beispiel Port 80 (HTTP), 1863 (MSN) oder 5190 (ICQ), um Firewallrestriktionen zu umgehen. Desweiteren schaut die Evaluierungsfunktion nach verdächtigen Teilzeichenketten, wie zum Beispiel den Namen des Bots (RBOT oder 1337-), Länderkürzeln (DEU, GBR oder USA), oder Einträgen für das Betriebssystem (XP oder 2K). Für jeden gefundenen Teilstring wird die Gesamtpunktzahl des Nicknamen um eins erhöht. Das Auftreten von speziellen Zeichen (zum Beispiel: [,] oder |) wird analog gewertet. Als letztes Kriterium wird überprüft, ob der Nicknamen ungewöhnlich viele Zahlen aufweist. Dazu wird für je zwei aufeinander folgende Zahlen die Punktzahl ebenfalls um einen Punkt erhöht.

Der Grund für die eher geringe Erhöhung der Punktzahl beim Auftreten von Teilzeichenketten, Sonderzeichen oder Zahlen, ist, dass diese kein eindeutiges Zeichen für eine Bot infizierte Maschine darstellen. Es gibt zum Beispiel viele „normale“ IRC Benutzer, die in bestimmten Gruppen oder auch Clans Online Spiele spielen und ihre Zugehörigkeit durch sogenannte „Clan-Tags“ in ihren Nicknamen ausdrücken. In den meisten Fällen wird der Name oder die Abkürzung des Clans in eckige Klammern [,] gestellt.

Aus diesem Grund erhöhen nur echte Anzeichen für einen infizierten Rechner die gesamt Punktzahl um mehr als einen Punkt. Echte Anzeichen sind in diesem Fall:

- Ein Treffer der regulären Ausdrücke (Section 3.3.2)
- Ein Verbindung zu einem IRC Server auf der Blackliste
- Die Verwendung eines Nicknamen, der auf der Blackliste steht
- Das Betreten eines bekannten Botnet Channels

3.3.2 Reguläre Ausdrücke

Jeder gefundene Nickname, der an die Analysefunktion übergeben wird, wird gegen eine ganze Reihe von regulären Ausdrücken geprüft, welche auf bekannte Botnamen auslösen. Derzeit enthält die entsprechende Konfigurationsdatei 69 verschiedene reguläre Aus-

drücke, um mehrere hundert verschiedene Bots zu erkennen. Diese regulären Ausdrücke wurden erzeugt, nachdem wir 4000 verschiedene Bots insbesondere auf ihre verwendeten Nicknamen hin untersucht haben. Um Fehlalarme weiter einzugrenzen sind die regulären Ausdrücke sehr speziell und matchen meist nur auf wenige oder einen Typ von Bot. Wir haben zum Beispiel den folgenden Ausdruck: `\[[0-9]\|[0-9]{4,}\]`. Dieser matcht auf Nicknamen der folgenden Art: `[0|12345]`. Daneben existiert ein weiterer Ausdruck, der nur auf Nicknamen wie `|12345` matcht. Obwohl beide Ausdrücke sich zu einem zusammenfassen lassen, haben wir sie separat gehalten, um mehr Flexibilität beim Optimieren oder Entfernen von nicht mehr aktuellen regulären Ausdrücken zu haben. Ein weiteres Beispiel für einen regulären Ausdruck ist: `\[[0-9]{1,2}\|(DEU,KOR,USA,...)\|[0-9]{4,}\]`. Hiermit werden Botnamen der folgenden Art entdeckt: `[00|DEU|579660]`, `[03|USA|147700]` oder `[0|KOR|43724]`.

Alle regulären Ausdrücke werden von Rishi in einer separaten Konfigurationsdatei gehalten, die sich mittels einer Update-Funktion über das Internet aktualisieren lässt. Neben dieser offiziellen Liste besteht die Möglichkeit eine eigene Liste mit Ausdrücken zu betreiben, wodurch Rishi zum einen an aktuelle Bots angepaßt werden kann und zum anderen gewählte Einstellungen bei Update nicht verlohren gehen.

Falls ein gefundener Nickname auf einen der regulären Ausdrücke matcht, wird zu der aktuellen Punktzahl, die Mindestzahl addiert, die nötig ist, um einen Alarm auszulösen. Das heisst, falls der Schwellwert bei 10 Punkten liegt, so wird dieser Wert auf die bisherige Gesamtzahl addiert.

3.3.3 Whitelisting

Um zu verhindern, dass bestimmte Maschinen fälschlicherweise von Rishi als Bot infiziert klassifiziert werden, gibt es eine frei konfigurierbare Whiteliste. Dadurch ist es möglich, sowohl Client IP Adressen als auch Server IP Adressen als „sauber“ zu klassifizieren. Verbindungen von solchen Rechnern oder zu solchen Servern erhalten dann von der Analysefunktion immer 0 Punkte. Neben der IP basierten Whiteliste existiert eine für Nicknamen. Sollte es zum Beispiel Nicknamen geben, die nicht zu einer Botverbindung gehören, aber dennoch einen Alarm auslösen oder zu einer hohen Punktzahl führen, so kann man diese ebenfalls in der Konfigurationsdatei hinterlegen und so von der weiteren Analyse ausschliessen.

Neben dieser fest konfigurierten Whiteliste verwaltet Rishi eine interne dynamische Whiteliste. Dieser Liste werden automatisch Nicknamen hinzugefügt oder auch wieder entfernt, je nachdem welche Werte sie von der Analysefunktion erhalten. Jeder Nickname der 0 Punkte erzielt wird automatisch dieser dynamischen Liste hinzugefügt. Um das Erkennen von normalen Nicknamen auf die gleiche Weise handhaben zu können wie das Erkennen von Bots, existiert auch für die Whiteliste eine Liste regulärer Ausdrücke für „erlaubte“ Nicknamen.

Während der Analysephase wird jeder Nickname sowohl gegen die fest konfigurierte als auch gegen die dynamische Whiteliste geprüft. Taucht der Name hier auf, so wird die Punktzahl auf 0 festgesetzt und die Analyse ist beendet. Darüberhinaus prüft Rishi auch auf Ähnlichkeit mit bereits auf der Whiteliste vermerkten Nicknamen. Zu diesem Zweck verwenden wir die N-Gram Analyse [Brø04]. Bei dieser Methode wird eine Zeichenkette in N große Teilstücke zerlegt. In unserem Fall ist $N = 2$ gewählt. Während der Analyse werden die zwei zu vergleichenden Nicknamen in die einzelnen Teile, jeweils aus zwei zusammenhängenden Buchstaben zerlegt. Jedes dieser Teile wird nun mit denen des anderen Namens verglichen und die Übereinstimmung werden gezählt. Je mehr Teilstücke übereinstimmen, desto ähnlicher sind sich die beiden Nicknamen.

Diese Methode ermöglicht uns, auf einfache und schnelle Weise zu entscheiden, ob ein Nickname eine Ähnlichkeit zu einem bereits auf der Whiteliste stehenden Nicknamen, besitzt und können diesen gegebenenfalls der Liste hinzufügen. Ändert zum Beispiel ein Benutzer, dessen Nickname bereits auf der Whiteliste steht, seinen IRC Namen von `myNickname` auf `myNickname_away`, so ist die Ähnlichkeit immer noch groß genug. Bei einer Analyse erhält auch dieser Name 0 Punkte und wird der dynamischen Whiteliste hinzugefügt. Dadurch ist es nicht notwendig, jeden bekannten Nicknamen und seine, möglicherweise geringfügigen, Änderungen auf die fest konfigurierte Liste zu setzen, sondern kann Rishi entscheiden lassen

3.3.4 Blacklisting

Das selbe Konzept wie für die Whitelisten wird von Rishi auch für die Verwaltung der Blacklisten verwendet. Hierfür existiert eine frei konfigurierbare Blackliste, um manuell Nicknamen von nicht erkannten Bots zu erkennen, sowie eine dynamische Liste, die von Rishi verwaltet wird. Jeder Nickname, der von der Analysefunktion mehr Punkte erhält als nötig sind, um einen Alarm auszulösen, wird automatisch dieser Liste hinzugefügt. Die auf den Blacklisten geführten Nicknamen werden wiederum in der Analysefunktion mit Hilfe der N-Gram Analyse für den Vergleich mit neuen Namen verwendet. Dadurch ist es möglich Bots, welche nicht genügend Punkte erhalten, zum Beispiel wegen eines fehlenden regulären Ausdrucks, die aber genug Ähnlichkeit zu einem bereits erkannten Botnamen haben, zu erkennen.

Neben der Blackliste für Nicknamen gibt es auch eine Command & Control Server IP Adressen Liste. Wird eine Verbindung entdeckt, die einen auf dieser Liste befindlichen Servern kontaktiert, so wird ebenfalls die Punktzahl soweit erhöht, dass der nötige Schwellwert für einen Alarm überschritten wird. Verbindungen zu IRC Servern mit Ports, die nicht dem Standard entsprechen (derzeit auf 6666, 6667 und 6668 konfiguriert), erhalten einen zusätzlichen Punkt zu ihrer Gesamtzahl.

3.3.5 Beispiel

Um den Analyseprozess besser zu verdeutlichen, geben wir hier ein kurzes Beispiel, wie die Erkennung eines Bot infizierten Rechners stattfindet. Angenommen, der Nickname `RBOT|DEU|XP-1234` wurde, aufgrund eines passenden regulären Ausdrucks, der dynamischen Blackliste hinzugefügt. Weiterhin nehmen wir an, dass die derzeitigen regulären Ausdrücke nur die Länderabkürzungen `DEU`, `USA` und `GBR` kennen. Eine zu einem späteren Zeitpunkt aufgezeichnete IRC Verbindung enthalte den Nicknamen `RBOT|CHN|XP-5678`. Allein durch die Analysefunktion würde dieser Namen 7 Punkte erhalten:

- 1 Punkt für jede verdächtige Teilzeichenkette: `RBOT` und `XP`
- 1 Punkt für jedes Sonderzeichen: `|` und `-`
- 1 Punkt für jedes Zahlenpärchen: `56` und `78`

Obwohl 7 Punkte bereits eine sehr hohe Punktzahl ist (normale IRC Verbindungen liegen in der Regel zwischen 0 und 4 Punkten) würde hier noch kein Alarm ausgelöst werden. Durch die N-Gram Analyse vergleicht Rishi diesen Nicknamen jedoch noch gegen die bereits als Bot identifizierten Namen und würde in diesem Fall auf genug Ähnlichkeit stossen, um weitere 10 Punkte zu addieren. Als Ergebnis liefert die Analysefunktion also 17 Punkte und löst einen Alarm aus.

4 Einschränkungen

Da unser Konzept auf regulären Ausdrücken basiert, die als Signaturen für bekannte Botnamen verwendet werden, kann Rishi nur Bots als solche identifizieren, für die auch ein passender Ausdruck existiert. Um diese Einschränkung zu lockern wurden noch weitere Kriterien für die Punktevergabe hinzugenommen: Teilzeichenketten, Sonderzeichen, Anzahl Zahlen und Ziel Port der Verbindung. Daraus resultiert, dass ein Botname eine höhere Punktzahl erreicht als ein normaler IRC Nickname. Es gibt allerdings auch Bots, die normale Namen verwenden, welche sich nicht von Namen realer Personen unterscheiden lassen. Zum Beispiel benutzt der Trojaner Zapchast.AU [Goe06] eine Liste von 32.398 verschiedenen Namen, die sich von normalen Namen kaum unterscheiden lassen, um sich zum IRC Netzwerk zu verbinden.

Solche Bots lassen sich mit der von uns vorgestellten Methode nicht automatisch erfassen. Allerdings kann auch die bloße Verwendung von IRC, die von Rishi ohnehin mitgeloggt wird, bei bestimmten Rechnern, z.B. Servern, auf eine Kompromittierung hindeuten.

Eine weitere Einschränkung der Software betrifft das Überwachen von Protokollkommandos, wie `JOIN` oder `NICK`, um die nötigen Daten zu erhalten. Im Falle von veränderten Protokollen würde die Erkennung fehlschlagen. Da Bots in Zukunft auf fortgeschrittenere Technologien, wie beispielsweise Peer-to-Peer Netzwerke, setzen, muss auch das Konzept von Rishi weiterentwickelt werden.

Solange jedoch zumindest ein Kommando des IRC Protokolls unverändert bleibt, besteht die Chance, dass Rishi die Botverbindung erkennt. Veranschaulichen können wir diese Aussage anhand eines Vorfalls im Dezember 2006. Während dieser Zeit beobachteten wir mit Rishi eine ganze Reihe von Verbindungen zu einem IRC Channel auf einem Server der auf Port 54932 lief. Ausser dem Channelnamen konnten keine weiteren Informationen extrahiert werden. Durch die mitgeloggte Server IP Adresse konnten wir eine separate Netzwerkverkehr Analyse starten. Das Ergebnis der Untersuchung ergab, dass es tatsächlich ein Bot war, der sein eigenes IRC Protokoll implementiert. Einige der gängigen IRC Kommandos wurde durch andere ersetzt. `NICK` wurde durch `SENDN` ersetzt, `USER` durch `SENDU` und `PRIVMSG` durch `SENDM`. In diesem Fall hatten wir das Glück, dass zumindest ein Kommando, nämlich `JOIN`, unverändert geblieben war. Sobald das komplette Protokoll andere Befehle verwendet, gibt es für Rishi keine Chance, diese Botverbindung zu erkennen. Diesen Nachteil teilen wir mit vielen signaturbasierten Erkennungsmethoden.

Ein weiteres mögliches Problem bilden webbasierte IRC Software oder Mini-Anwendungen, die es neuen Benutzer erlauben, das IRC Netzwerk auf einfache Weise kennenzulernen. Die hier verwendeten Nicknamen haben oft ein Schema, welches Fehlalarme erzeugen könnte. Allerdings hatten wir bisher noch keine Probleme mit webbasierten Client. ICQ verwendet beispielsweise einen webbasierten IRC Client, der auch für Personen ohne ICQ Account nutzbar ist. Solche Nutzer bekommen oftmals einen Nicknamen wie `Guest_979`, `LeaP_195` oder `onn_5201`. Da es allerdings keine regulären Ausdrücke gibt, die auf diese Namen passen, erhalten solche Verbindungen derzeit in der Regel zwischen 3-4 Punkten:

- 1 Punkt für das Sonderzeichen: `_`
- 1 Punkt für einen ungewöhnlichen Server Port. Bei ICQ zum Beispiel 7012.
- 1-2 Punkte für das auftreten von aufeinanderfolgenden Zahlen.

Da die Ziel IP Adressen jedoch weitläufig bekannt ist, lassen sich diese Server auch schnell der statischen Whiteliste hinzufügen.

Es gibt jedoch auch einen Fall, der zu einem Fehlalarm führte. Es gibt eine TV Software, die auch das IRC Protokoll verwendet, zu welchem Zweck auch immer. Benutzer

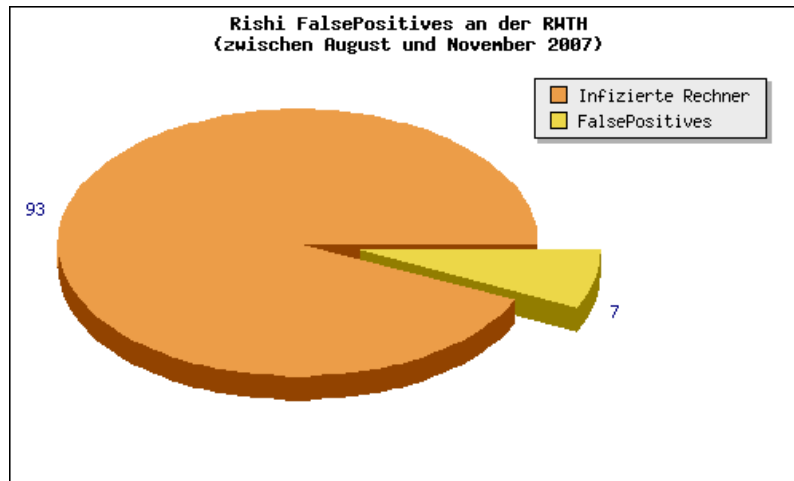


Abbildung 4: Von Rishi erkannte Bots

2006 an der RWTH und wird kontinuierlich weiterentwickelt. Der Analysezeitraum umfasst exakt 87 Tage. Während dieser Zeit wurden 100 Maschinen von Rishi als Bot infiziert gemeldet, wovon sich 7 im Nachhinein als Fehlalarm herausstellten (Abbildung 4). Nachfolgend sind die falsch klassifizierten Nicknamen aufgelistet:

- asikoreczka|000000
- S00000000000000??40783624570
- ustreamer|4595
- 000000
- h1234
- b3450497
- o5175

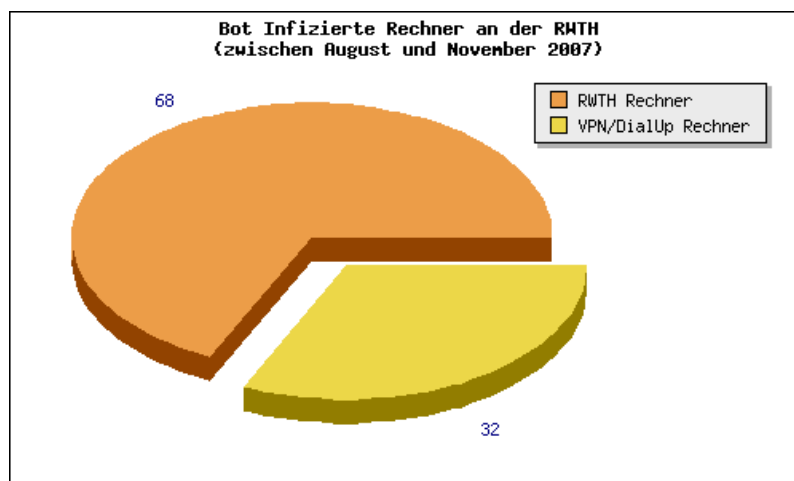


Abbildung 5: Aufteilung zwischen festen IP Adressen und Einwahlkonten

Abbildung 5 zeigt die Verteilung von infizierten Rechnern innerhalb des RWTH Netzwerkes, zwischen fest vergebenen IP Adressen, für Instituts- oder Wohnheimrechnern, und Einwahlkonten, wie für das Wireless LAN.

In manchen Fällen war es möglich die Bot infizierten Rechner noch lange bevor diese durch andere Intrusion Detection Mechanismen erkannt wurden, zu identifizieren und zu sperren. Zum Beispiel meldete Rishi am 25.10.2007 den Rechner mit der IP Adresse 137.226.xxx.xxx als Bot infiziert mit folgendem Nicknamen: [01|DEU|886017]. Der IRC Server lief auf Port 1863, einem Standardport für Microsoft Messenger. Wirklich aktiv, so dass andere Intrusion Detection Systeme den Rechner ebenfalls meldeten, wurde die Maschine erst am 28.10.2007. Diese hohe Vorlaufzeit ist aber eher eine Ausnahme, in den meisten Fällen beginnen die infizierten Rechner nur wenige Sekunden nach der Kompromittierung und dem Verbinden zum Command & Control Server mit der Weiterverbreitung ihrer Schadsoftware.

Eine genau Angabe über die Erkennungsrate von Rishi lässt sich nur sehr schwer geben, da sich nicht eindeutig feststellen lässt, wieviele Rechner in dem beobachteten Netzwerk tatsächlich mit einem IRC Bot infiziert sind. Eine Möglichkeit ist der Vergleich mit anderen im Netzwerk aufgestellten Intrusion Detection Systemen, zum Beispiel dem Blast-o-Mat [GHH06]. Der Blast-o-Mat verwendet drei verschiedene Module um kompromittierte Rechner zu erkennen:

- Erkennung von Portscans durch Zählen von TCP SYN Paketen innerhalb eines vorgegebenen Zeitraums für jeden Rechner.
- Erkennung von SPAM-verschickenden Rechnern durch Zählen von verwendeten Mailservern und verschiedenen Absenderadressen.
- Erkennung von automatisch verbreitender Malware durch den Einsatz von HoneyPot Systemen.

Während des 87 tägigen Analysezeitraums lief Rishi ohne nennenswerte Unterbrechung und erkannte, abzüglich der 7 Fehlalarme, 93 infizierte Maschinen. Von diesen Rechnern wurden nur 40 auch von unserer installierten Intrusion Detection Software erkannt. Der Grund für diesen Unterschied liegt zum grössten Teil an den verwendeten Verbreitungsmethoden der Malware. Viele der erkannten IRC Bots verwendeten Instant Messenger, wie zum Beispiel den Microsoft Messenger, um sich zu verbreiten. Wieder andere verhielten sich eher ruhig, bzw. hatten Keylogger laufen, um Benutzerdaten zu sammeln und wendeten keine Techniken zur Verbreitung an.

Da keine weitere Software zur Erkennung von IRC Bot infizierten Rechnern im Netzwerk der RWTH Aachen installiert ist, lässt sich nicht sagen, wieviele Maschinen nicht erkannt worden sind. Tatsache ist, dass es durchaus eine Reihe von Bots gibt, die sich nicht mit Rishi automatisiert erkennen lassen. Allerdings fallen diese Rechner meist bei einer manuellen Analyse der Logfiles direkt auf. Einige dieser Erkennungsmerkmale werden im nächsten Abschnitt genauer beschrieben.

5.2 Beobachtete Eigenschaften von Botnetze

Abbildung 6 zeigt die Top 10 der am häufigsten verwendeten Command & Control Server Ports. Derzeit ist Port 80 der Port, auf dem die meisten Botnetze ihre Server laufen haben. Der Grund hierfür liegt darin, dass dieser Port gewöhnlich in Firewalls freigeschaltet ist, da hier normalerweise HTTP Daten durchlaufen. Ebenfalls unter den Top-Ports liegen die Ports 1863, 83, 81, 443 und 57, die normalerweise von Diensten wie MSN, HTTP-Proxies, HTTPs oder DNS verwendet werden.

Neben den IRC-basierten Bots, tauchen auch immer häufiger Bots auf, die das HTTP Protokoll zur Kommunikation verwenden. Maschinen die mit einem HTTP Bot infiziert sind, lassen sich ebenfalls durch Analysieren des Netzwerkverkehrs, insbesondere der

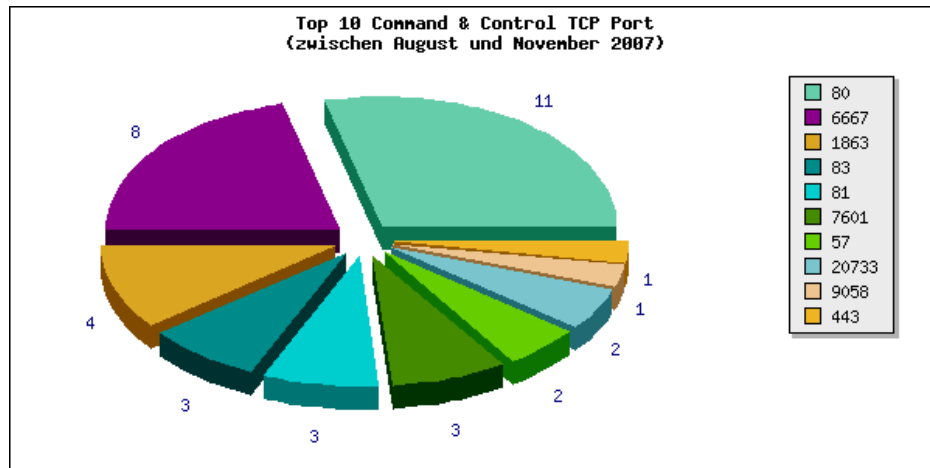


Abbildung 6: Meist verwendete Command & Control Server Ports

URLs, erkennen. Deshalb lässt sich das Konzept von Rishi auch auf diese Art der Kommunikation anwenden. Die aktuelle Version von Rishi besitzt bereits experimentelle Ansätze um auch diese Art der Malware zu entdecken. Zu diesem Zweck analysieren wir URLs auf verdächtige Zeichenketten wie zum Beispiel „cnt=DEU“ in Kombination mit „cmd.php“.

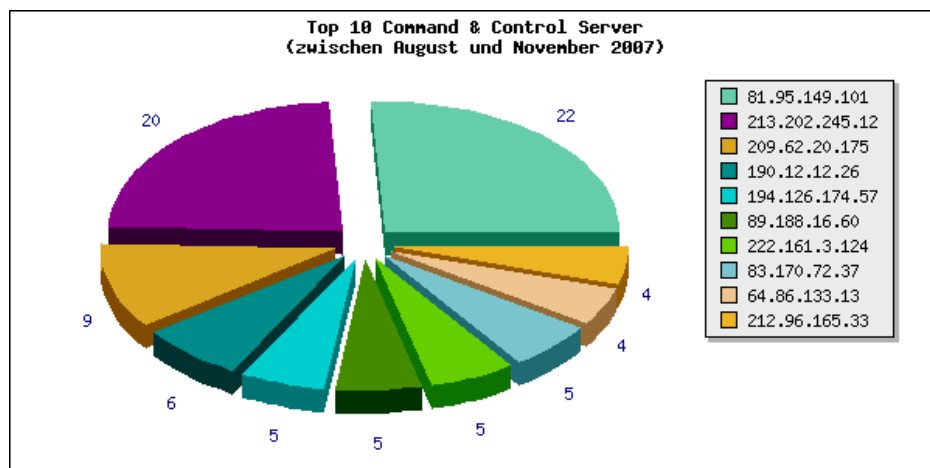


Abbildung 7: Verbindungen zu Command & Control Servern

Insgesamt konnten wir im festgelegten Zeitraum Verbindungen zu 85 unterschiedlichen IP Adressen von Botnetz Servern beobachten. Dabei wurden 24 unterschiedliche Ports verwendet. Abbildung 7 zeigt die Server, die von den meisten Bots kontaktiert worden sind.

Sortiert man alle gefundenen C&C Server innerhalb des Analyse-Zeitraums nach den den IP Adressen zugeordneten Herkunftsländern, so ergibt sich das in Abbildung 8 dargestellte Bild.

Obwohl die hier vorgestellte Methode durchaus einige Einschränkungen mit sich bringt (Abschnitt 4), bietet sie dennoch eine gute und genaue Erkennungsrate für IRC Bots, ohne viel manuellen Einsatz oder hohe False-Positives.

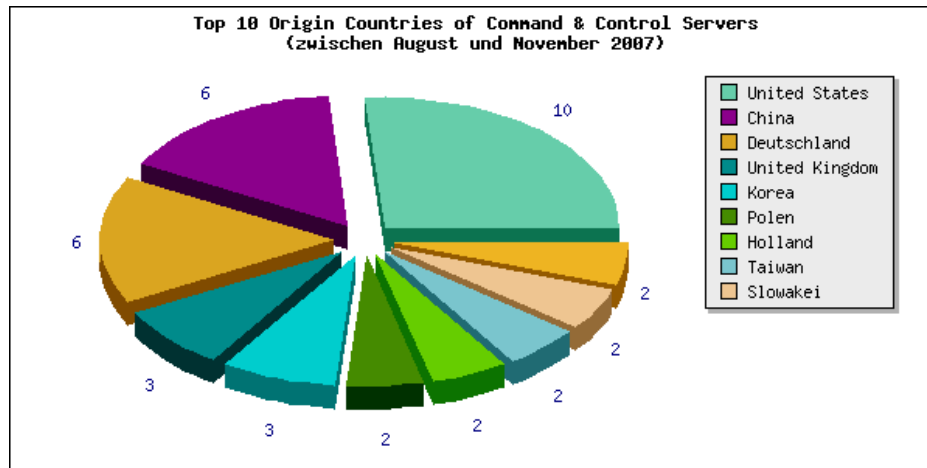


Abbildung 8: Command & Control Server sortiert nach Herkunftsland

6 Fazit

Die Erkennung von Bot infizierten Rechnern ist nicht immer einfach: Die Malware versteckt sich auf dem Rechner und wird oft nur unter bestimmten Bedingungen aktiv. Von aussen betrachtet lässt sich die Infektion nur schwer erkennen, da es viele Möglichkeiten gibt, wie ein Bot auf einen Computer gelangen kann (E-Mail, Webseite, USB-Stick, etc.). Es gibt jedoch eine Gemeinsamkeit, die alle IRC Bots verbindet: Sie benötigen einen Kommunikationskanal zu ihrem Command & Control Server. Dieser Kommunikationskanal bietet eine Möglichkeit der Erkennung.

In dieser Veröffentlichung haben wir eine effektive, aber dennoch einfache Methode vorgestellt, um IRC-basierte Bots zu erkennen. Hierfür haben wir uns die Eigenschaften des Kommunikationsprotokolls zu Nutze gemacht und können mit Hilfe einer Punktevergabe-Funktion, N-Gram Analyse und Black- und Whitelisten Bot infizierte Rechner effektiv erkennen.

Das aus der Umsetzung der Theorie hervorgegangene Programm Rishi hat sich als eine sehr gute Ergänzung zu unseren bestehenden Intrusion Detection Mechanismen erwiesen und ist seit der ersten Entwicklungsversion Ende 2006 permanent an der RWTH Aachen im Einsatz. Neben der Früherkennung von infizierten Rechnern bietet Rishi auch die Möglichkeit, Informationen über den Command & Control Server zu erfahren. Mit Hilfe der zusätzlichen Informationen lassen sich Botnetze überwachen, um mehr über diese und die durchgeführten Aktionen herauszufinden.

Literatur

- [Ant04] Avira AntiVir. *Worm/Rbot.210944 - Worm*. 2004.
http://www.avira.com/en/threats/section/fulldetails/id_vir/3469/worm_rbot.210944.html.
- [Ant05] Avira AntiVir. *Worm/Korgo.Fvar - Worm*. 2005.
http://www.avira.com/de/threats/section/fulldetails/id_vir/1874/worm_korgo.f.var.html.
- [Brø04] Tom Brøndsted. *n-gram analysis*. 2004.
<http://kom.aau.dk/~tb/ngram/>.

- [BS06] James R. Binkley and Suresh Singh. An algorithm for anomaly-based botnet detection. In *Proceedings of USENIX Steps to Reducing Unwanted Traffic on the Internet Workshop (SRUTI)*, pages 43–48, July 2006.
- [Che06] Yan Chen. Irc-based botnet detection on high-speed routers. In *ARO-DARPA-DHS Special Workshop on Botnets*, June 2006.
- [GHH06] Jan Goebel, Jens Hektor, and Thorsten Holz. Advanced honeypot-based intrusion detection. *USENIX ;login.*, 31(6), December 2006.
- [Goe06] Jan Goebel. *A short visit to trojan Zapchast.AU*. 2006.
http://zeroq.kulando.de/resource/papers/download/Trojan_Zapchast.pdf.
- [Kal00] C. Kalt. Internet relay chat: Architecture, April 2000. Request for Comments: RFC 2810.
- [Koh07] Javier Kohen. pcap – python pcap extension, Accessed: November 2007. Internet: <http://oss.coresecurity.com/projects/pcapy.html>.
- [Kri04] John Kristoff. Botnets. In *North American Network Operators' Group Meeting (NANOG32)*, October 2004.
- [LWLS06] Carl Livadas, Bob Walsh, David Lapsley, and Tim Strayer. Using machine learning techniques to identify botnet traffic. In *Proceedings of 2nd IEEE LCN Workshop on Network Security*, November 2006.
- [Rac04] Stephane Racine. Analysis of internet relay chat usage by ddos zombies. Master's thesis, Swiss Federal Institute of Technologie, Zurich, April 2004.
- [Rit07] Jordan Ritter. ngrep – network grep, Accessed: November 2007. Internet: <http://ngrep.sourceforge.net/>.
- [SM07] Elizabeth Stinson and John C. Mitchell. Charaterizing bot's remote control behaviour. In *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA)*, July 2007.
- [SWLL06] W. Timothy Strayer, Robert Walsh, Carl Livadas, and David Lapsley. Detecting botnets with tight command and control. In *Proceedings of the 31st IEEE Conference on Local Computer Networks*, November 2006.