Schadsoftwareerkennung auf Netzwerkebene ohne Einzelpaketanalyse

Florian Tegeler
Institut für Informatik
Universität Göttingen
tegeler@cs.uni-goettingen.de

Abstract: Eines der größten Sicherheitsprobleme im Internet sind Bots – gekaperte Rechner, die in großer Zahl Schäden durch SPAM, Denial-of-Service-Angriffe oder Informationsdiebstahl anrichten. Entsprechend aktiv ist die Forschungsgemeinde bei der Entwicklung von Detektionsmechanismen, insbesondere auch im Feld der netzwerbasierten Erkennung. Da dazu jedoch typischerweise Paketinhalte analysiert werden müssen, stößt diese klassische netzwerkbasierte Detektion bei verschlüsselter Kommunikation an ihre Grenzen.

Als Antwort auf dieses Problem präsentieren wir BOTFINDER – ein neuartiges System, dass Bots anhand ihrer verbindungsorientierten Kommunikationsstruktur mit dem Botmaster erkennt und vollständig auf eine Analyse von Paketinhalten verzichtet. Hierzu beobachtet es das Kommunikationsverhalten der Schadsoftware in einer kontrollierten Umgebung und errechnet hieraus aussagekräftige Modelle. Diese Modelle werden dann im Praxisbetrieb mit dem Netzwerkverkehr der einzelnen Rechner im Netz verglichen.

Unsere Ergebnisse mit einer repräsentativen Auswahl verschiedener Bots auf realen Netzwerkmitschnitten zeigen, dass BOTFINDER in der Lage ist, Bots mit hoher Trefferquote bei gleichzeitig wenigen Falsch-Positiven zu erkennen.

1 Einleitung

Eine Vielzahl von Sicherheitsproblemen im heutigen Internet lässt sich auf Bots zurückführen, also Schadsoftware, die ohne das Wissen des Nutzers den Rechner infiziert und so in ein großes Netzwerk, dem Botnet [RZMT06], einbindet. Diese Netzwerke stehen typischerweise unter zentraler Kontrolle eines kriminellen Botmasters, der mit dem Netzwerk dem Versand von SPAM, Angriffen durch Distributed-Denial-of-Service (DDoS) oder Identitätsdiebstahl nachgehen kann. Entsprechend groß sind die Bemühungen bei der Entwicklung von Gegenmaßnahmen, um bereits einzelne Infektionen zu vermeiden bzw. aufzuspüren.

Der klassische Weg zur Schadsoftwarebekämpfung ist die Installation von Anti-Virus-Software auf den Rechnern der Endnutzer. Hierbei gibt es jedoch Probleme, wie z.B. den hohen Wartungsaufwand, die Erfordernis aktualisierter Virensignaturen und die lokale Sicht des Scanners. Als Konsequenz wird angestrebt, die lokalen Anti-Virus-Scanner durch netzwerkbasierte Lösungen zu ergänzen, die es erlauben, eine Vielzahl an Rechnern gleichzeitig zu untersuchen und miteinander in Beziehung zu setzen.

Da immer mehr Bots ihre Kommunikation verschleiern und verschlüsseln, um so einer netzwerkbasierten Erkennung zu entgehen, sollte eine erfolgreiche Lösung folgende Grundanforderungen erfüllen:

(a) individuelle Bot-Infektionen im Netzwerk erkennen,

(b) nur auf Verbindungsdaten basieren, um gegenüber verschlüsseltem Netzwerkverkehr immun zu sein und

(c) auch für Bots funktionieren, die sich nicht durch aussergewöhnliches Kommunikationsverhalten wie z.B. bei DDoS oder SPAM-Versand auszeichnen.

Als eine solche Lösung stellen wir BOTFINDER [TFVK12] vor – ein System, das individuelle Bot-Infektionen durch Beobachtung des Netzwerkverkehrs erkennt. BOTFINDER nutzt aus, dass die Kommunikation zwischen infiziertem Rechner und dem Botmaster, der sogenannte *Command-and-Control (C&C)*-Verkehr, oftmals regelmäßigen Mustern unterliegt die dann zur Erkennung verwendet werden können. Diese Muster erlernt BOTFINDER durch die Beobachtung des Netzwerkverkehrs von Schadsoftware, die in kontrollierter Umgebung ausgeführt wird. Aus diesen Mustern wird dann ein Modell des Bots erstellt, mit dem unbekannter Netzwerkverkehr verglichen und Bots darin erkannt werden können.

Aufgrund seines Designs arbeitet BOTFINDER anders als bisher veröffentlichte netzwerkbasierte Ansätze und ist in der Lage, einzelne Botinfektionen im Netzwerk zu erkennen. Zudem benötigt es keine Aktivitätskorrelationen der Netzwerkkommunikation wie beispielsweise BotSniffer [GZL08] oder BotMiner [GPZL08]. Die erwähnten Systeme erfordern zudem oft auffällige Aktivitäten wie SPAM oder DDoS Netzwerkverkehr, was die Erkennung von besonders getarnten Bots erschwert. Natürlich gibt es einige Systeme [WBH+09, GH07, GPY+07], die die Erkennung von individuellen Botinfektionen erlauben; diese arbeiten jedoch auf Basis der *deep packet inspection*, also der Analyse jedes einzelnen Paketinhaltes. Demgegenüber verwendet BOTFINDER nur NetFlow-ähnliche [Cla04] Informationen über die Netzwerkverbindungen und ist somit auch in der Lage, auf verschüsseltem Netzwerkverkehr zu arbeiten.

Um unseren Ansatz zu evaluieren, wurden Erkennungsmodelle für einige, zurzeit aktive, Schadsoftwarefamilien erzeugt, die eine Mischung verschiedener Infektions- und *C&C*-Strategien anwenden. Unsere Ergebnisse zeigen, dass BOTFINDER Netzkommunikation von Schadsoftware mit hoher Genauigkeit erkennt. Die erzeugten Modelle wurden zudem auf zwei Datensätzen – dem Mitschnitt der Netzkommunikation eines akademischen Forschungslabors und den Verbindungsdaten eines großen Internetanbieteres (mit mehreren Milliarden Verbindungen) – angewandt. BOTFINDER erzeugte dabei vielversprechende Erkennungsergebnisse bei niedriger Falsch-Positiv-Rate.

Insgesamt stellen wir folgende Ergebnisse vor:

- Die Beobachtung, dass der Netzwerkverkehr von Bots gewisse Periodizitäten aufweist, die genutzt werden können, um eine Unterscheidung zu normalem, unbedenklichem Netzwerkverkehr vorzunehmen. Durch die Unabhängigkeit von Paketinhalten kann unser Ansatz auch verschlüsselte bzw. verschleierte Kommunikation verarbeiten und Bots erkennen.

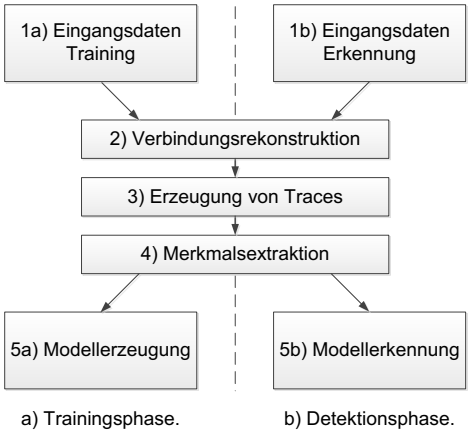

Abbildung 1: BOTFINDER-Architektur.

- BOTFINDER, ein System, das durch Beobachtung des Netzwerkverkehrs von Schadsoftware in kontrollierter Umgebung Modelle erzeugt, die dann zur Erkennung eingesetzt werden können.
- Ein Prototyp von BOTFINDER, der in der Lage ist, auf hochperformanten Netzwerken zu arbeiten und dort individuelle Schadsoftwareinfektionen zu erkennen, wie die Evaluation auf Basis verschiedener Netzwerkmitschnitte zeigt.

2 BotFinder

BOTFINDER erkennt Schadsoftware auf Netzwerkebene durch den Vergleich statistischer Merkmale des Netzwerkverkehrs mit vorab erlerntemn Modellen, die das Bot-Verhalten widerspiegeln. Dementsprechend arbeitet BOTFINDER in zwei Phasen: der *Trainingsphase* und der *Detektionsphase*. Während der Trainingsphase analysiert BOTFINDER die charakteristischen Merkmale des *C&C*-Verkehrs verschiedener Botfamilien und erzeugt statistische Modelle, die das Verhalten der Bots möglichst genau abbilden. In der Detektionsphase werden die Modelle auf den zu analysierenden Netzwerkverkehr angewandt und potentielle Botinfektionen identifiziert.

Wie Abbildung 1 zeigt, unterteilen sich beide Phasen in die fünf im Folgenden dargestellten Schritte.

2.1 Eingangsdatenverarbeitung

Im ersten Schritt wird Schadsoftware in kontrollierter Umgebung – wie z.B. Anubis [BKK08], BitBlaze [SBY+08], CWSandbox [WHF07] oder Ether [DRSL08] – ausgeführt und aller Netzwerkverkehr mitgeschnitten. Eine besondere Herausforderung hierbei ist die korrekte Klassifikation der binären Schadsoftwareprogramme zu Familien, wobei unsere Klassifikation auf Ergebnissen verschiedener Anti-Viren-Scanner, die über VirusTotal[1] abgefragt wurden, und Verhaltensanalysen in Anubis [BCH+09] beruhen. Nichtsdestotrotz können fehlerhafte Zuordnungen auftreten, was jedoch als statistische Schwankung die grundlegende Modellierung der Botfamilie nicht zu stark einschränkt.

2.2 Verbindungsrekonstruktion

In einem zweiten Schritt wird, wenn kein NetFlow direkt verwendet wird, der Paketdatenstrom logisch zu Verbindungen zusammengefasst. Wird ein normaler Netzwerkmitschnitt verwendet, erzeugt BOTFINDER Daten ähnlichen Formats zu NetFlow, dem Industriestandard für die Sammlung und Auswertung von Netzwerkinformationen.

2.3 Erzeugung von Traces

Im dritten Schritt werden Verbindungen zwischen zwei Netzwerkendpunkten zu chronologisch geordneten Folgen, sogenannten *Traces*, zusammengefasst. Die Traces bilden den wesentlichen Baustein von BOTFINDER, da auf Ihnen die gesamte statistische Auswertung erfolgt. Wie in Abbildung 2 im Beispiel dargestellt, zeigen einige Traces, wie $\mathcal{T}_2$ von A zu C auf Port 80, deutliche Regularitäten. Diese Regelmäßigkeit, erlaubt es BOTFINDER die Kommunikation durch das statistische Merkmal der nahezu konstanten Abstände zwischen zwei Verbindungen zu charakterisieren.

Um statistisch aussagekräftige Daten zu gewinnen, ist es notwendig, eine Mindestanzahl (typischerweise 10-50) von Verbindungen $|\mathcal{T}|_{min}$ zu beobachten. Eine weitere Herausforderung bei der Ermittlung der Traces ist die Unterscheidung zwischen echtem *C&C*-Verkehr und Störverbindungen, die teilweise bewusst von der Schadsoftware erzeugt werden [FSP+06] um den eigentlichen Kontrollverkehr zu tarnen. Um nicht-relevante Anteile des Netzwerkverkehrs zu ermitteln, verwenden wir zum Einen eine kurze Whitelist mit oft aufgerufenen Diensten wie Microsoft oder Google und zum Anderen eine Blacklist um Verbindungen zu bekannten Schadsoftwareservern zu erkennen und auszuwerten. Interessanterweise zeigt sich, dass fehlerhaft zum Kontrollverkehr gerechnete Netzwerkkommunikation oder der Verzicht auf die Blacklistauswertung die Modellergebnisse nur unwesentlich verschlechtert.

[1] `http://www.virustotal.com`

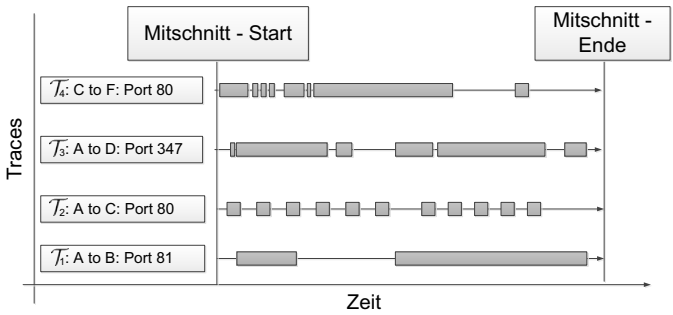

Abbildung 2: Verbindungssequenzen mit unterschiedlichen Verhaltensmustern..

2.4 Merkmalsextraktion

BOTFINDER erzeugt dann im vierten Schritt die grundlegenden statistischen Merkmale:

1. Das **durchschnittliche Intervall zwischen zwei Verbindungen** zwischen Quell- und Ziel-IP-Addresse auf einem festen Port. Dieses Zeitintervall ist oft periodisch, da die Bots regelmäßig den *C&C*-Server kontaktieren, um neue Befehle zu erhalten. Eine Kommunikation, bei der der Server ein neues Kommando des Botmasters an alle Bots weiterleitet, ist aufgrund der weiten Verbreitung von NATs und Firewalls oft nicht möglich, so dass die Bots selbstständig regelmäßig um neue Befehle bitten müssen. Interessanterweise ist es nicht einfach, zufällig erscheinenden Netzwerkverkehr, der trotzdem eine regelmäßige Kommunikation ermöglicht, zu erzeugen [TFVK12].
2. Die **durchschnittliche Zeit dieser Verbindungen**. Da bei den wiederholten Kommandoabfragen oftmals keine neuen Kommandos bereitstehen, ist die Kommunikation häufig gleich lang und von gleichem Transfervolumen. Selbst wenn neue Kommandos oder bspw. eine neue SPAM-Liste zur Verfügung steht, ist die Anfrage immer noch von festem Format.
3. Die **durchschnittliche Menge an Daten**, die in Quell- und Zielrichtung übertragen wurden.
4. Eine **Fouriertransformation der Folge an Startzeiten** in der Trace. Durch diese Frequenzanalyse können kleinere Unregelmäßigkeiten in der Kommunikation besser ausgeglichen werden und die grundlegende Periodizität wird besser erfasst.

2.5 Modellerzeugung

Im letzten Schritt clustered BOTFINDER die jeweiligen Merkmale und erzeugt so statistisch aussagekräftige Modelle. Jedes Merkmal wird hierbei für sich genommen verarbeitet,

Name	Daten-menge	Interne Rechner	Parallele Verbind.	Start-zeit	Länge	Verb.
LabCapture	$\approx$ 3.6 TB	$\approx$80	$\approx$60	2011-05-04	84 days	$\approx 64.3 \cdot 10^6$
ISPNetFlow	$\approx$ 540 TB	$\approx$1M	$\approx$250k	2011-05-28	37 days	$\approx 2.5 \cdot 10^{10}$

Tabelle 1: Datensätze

da wir oft keine Korrelation zwischen den verschiedenen Merkmalen beobachten konnten. Als Beispiel seien zwei Versionen des gleichen Bots angeführt, die sich zu zwei verschiedenen *C&C*-Servern C_1 und C_2 verbinden und je nach Version unterschiedliche Datenmengen übertragen. Nichtsdestotrotz können die beiden Versionen das gleiche Verbindungsintervall verwenden.

Nach dem Clustern werden die kleineren Cluster als nicht-repräsentativ verworfen und aus den großen Clustern die Modellbeschreibung abgeleitet. Ein Modell kann daher beispielsweise verstanden werden als:

> *Ein durchschnittliches Intervall von 850 oder 2.100 Sekunden, ein Transfer von 51kB zur Quelle und 140 Bytes zur Destination, eine Verbindungsdauer von 0,2 oder 10 Sekunden und eine Kommunikationsfrequenz von 0,0012Hz oder 0,04Hz deuten auf eine Dedler-Infektion hin.*

2.6 Modellerkennung

In der Detektionsphase wird der zu untersuchende Netzwerkverkehr genau wie der Trainingsverkehr statistisch analysiert und die Merkmale extrahiert. Diese Merkmale werden nun mit den vorliegenden Bot-Modellen verglichen und für jeden "Treffer" im Modell M wird eine Bewertungsvariable γ_M erhöht. Wie stark die Erhöhung ausfällt, hängt von der Qualität des Clusters und der vergleichenden Trace ab; je regelmäßiger der Netzwerkverkehr ist, desto stärker wird der Treffer gewichtet. Ist γ_M größer als ein – vom BOTFINDER-Administrator zur setzender – Wert a, gilt das Modell als akzeptiert. Der Administrator kann zudem verlangen, dass mindestens h verschiedene Merkmale des Modells getroffen werden, um Zufallstreffer so weit wie möglich zu reduzieren.

Für das Modell M mit dem größten γ_M wird, sofern $\gamma_M > a$, durch BOTFINDER ein Alarm ausgelöst.

3 Evaluation

Bei der Evaluation kamen sechs repräsentative Schadsoftwarefamilien mit durchschnittlich 32 verschiedenen Varianten zum Einsatz: **Banbra**, ein Spyware-Trojaner, die **Bifrose** (oder Bifrost) Familie, der DDoS Bot **Blackenergy** der bereits Polymorphismus und verschlüsselte Kommunikation beherrscht, der Spambot **Dedler**, **Pushdo**, ein stark verbrei-

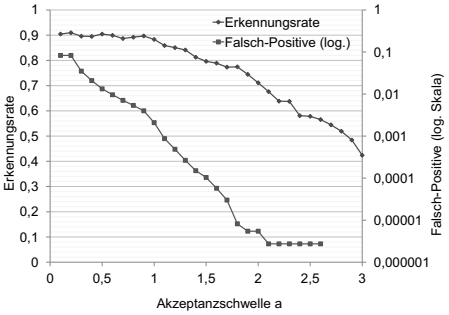

Abbildung 3: Detektions- und Falsch-Positiv-Rate von BOTFINDER im Kreuzvalidierungsexperiment.

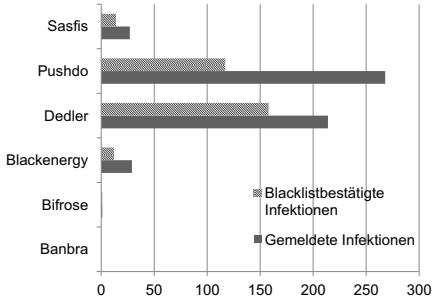

Abbildung 4: Infektionen nach Botfamilien aufgeschlüsselt. 56% der Verbindungen wurden zu bekannten Schadsoftwareservern aufgebaut.

teter SPAM-Bot, der auch unter den Namen *Pandex* oder *Cutwail* bekannt ist, und **Sasfis**, ein klassischen Trojaner.

Insgesamt wurden drei Experimente zur BOTFINDER-Evaluation durchgeführt: Ersten, ein Kreuzvalidierungsexperiment auf Basis von vollständigen Netzwerkmitschnitten aus einem Forschungslabor, zweitens ein Vergleich mit *Bothunter*, obwohl *Bothunter* Paketinhalte analysiert, jedoch kein direkt vergleichbares Produkt existiert, und drittens der Analyse eines Netzwerkmitschnittes eines großen Internetanbieters. Als Datensätze wurden die beiden in Tabelle 1 dargestellten Datensätze *LabCapture* und *ISPNetFlow* verwendet, wobei *LabCapture* ein vollständiger Mitschnitt ist, so dass potentielle Infektionen manuell überprüft werden können, und *ISPNetFlow* ein reiner NetFlow-Datensatz ohne Paketinhalte ist.

3.1 Kreuzvalidierung

Um die Erkennungsrate von BOTFINDER zu ermitteln, wurden die in kontrollierter Umgebung erhobenen Trainingsdaten mit dem *LabCapture* Datensatz vermischt und für verschiedene Erkennungsgrenzwerte α jeweils 50 Durchläufe folgender Art gemacht:

1. Die Schadsoftwaretraces wurden in ein Trainingsset $\mathcal{W}$ (70%) und ein Erkennungsset $\mathcal{D}$ (30%) geteilt.

2. $\mathcal{D}$ wurde mit dem *LabCapture* Datensatz vermengt und diente unter der Annahme, dass *LabCapture* infektionsfrei ist, als Grundlage zur Bestimmung der Falsch-Positiven.

3. BOTFINDER wurde auf $\mathcal{W}$ trainiert.

4. Die so erzeugten Modelle wurden auf die aus $\mathcal{D}$ und *LabCapture* erzeugten Daten angewandt.

Schadsoftware-Familie	BOTFINDER Erkennungsrate	BOTFINDER Falsch-Positive	*Bothunter* Erkennungsrate
Banbra	100%	0	24%
Bifrose	49%	0	0%
Blackenergy	85%	2	21%
Dedler	63%	0	n/a
Pushdo	81%	0	11%
Sasfis	87%	1	0%

Tabelle 2: Erkennungsrate und Falsch-Positive bei $a = 1.8$ im Kreuzvalidierungsexperiment und im Vergleich zu *Bothunter*.

Sehr niedrige Grenzwerte führen erwartungsgemäß zu hohen Erkennungsraten von mehr als 90%, aber auch hohen Fehlerquoten. Jedoch wird auf Abbildung 3 deutlich, dass die Erkennungsrate linear, die Fehlerrate dagegen exponentiell abfällt. Ein Optimum liegt für $a \in [0, 3]$ und $h = 3$ vor, wo z.B. für $a = 1.8$ 77% Erkennungsrate bei $5 \cdot 10^{-6}$ Falsch-Positiven erreicht wird. Wie Tabelle 2 zeigt, variieren die Erkennungsraten jedoch für die verschiedenen Bot-Familien.

3.2 Vergleich mit BotHunter

Bei einem direkten Vergleich mit *Bothunter* [GPY+07], einem paketinhaltbasierten, weiterentwickelten Snort[2], fällt auf, dass die Erkennungsrate von BOTFINDER deutlich über der von *Bothunter* liegt. *Bothunter* versucht den gesamten Infektionsprozess nachzuvollziehen und arbeitet stark mit Blacklists und fertigen Signaturen. Bei unserer Untersuchung konnten wir feststellen, dass *Bothunter* zudem stark auf Downloads von Binaries reagiert und dort auch viele Falsch-Positive (37 von 41 Alarmen) hervorruft, jedoch einige echte Infektionen übersieht. Interessanterweise hat *Bothunter* noch einige Infektionen im Forschungsnetz (*LabCapture*) aufdecken können, die von BOTFINDER nicht entdeckt wurden, da es auf diese Bots nicht trainiert worden war.

3.3 *ISPNetFlow*-Analyse

Da der *ISPNetFlow*-Datensatz nur Verbindungsdaten beinhaltet, können keine absoluten Aussagen über die Richtigkeit einer gemeldeten Bot-Infektion gemacht werden, jedoch können die Ziel-IPs der identifizierten Schadsoftwareverbindungen mit bekannten Blacklists verglichen werden um so eine Abschätzung der Erkennungsqualität zu erhalten. Die Daten wurden in 24-Stunden-Zeitschlitze unterteilt und durch BOTFINDER, das vorher mit allen erhobenen Schadsoftwaretraces trainiert wurde, ausgewertet. Insgesamt wurden im Durchschnitt 14.6 Traces als verdächtig markiert (542 insgesamt) – eine Menge, die

[2] `http://www.snort.org`

problemlos von einem Administrator im Tagesgeschäft genauer analysiert werden kann. Abbildung 4 zeigt die Gesamtzahl von Infektionen der verschiedenen Familien, wobei insbesondere die beiden SPAM-Bots Pushdo und Dedler mit 268 bzw. 214 Infektionen dominieren.

Ein Vergleich der Zieladdressen von als infiziert eingeschätzten Traces mit Informationen öffentlich verfügbarer Blacklists[3] ergab, dass es sich bei mindestens *56% davon um Verbindungen zu bekannten Schadsoftwareservern* handelt. Interessanterweise finden sich von den nicht in den Blacklists geführten 238 IP-Addressen 85 in 5 großen Clustern. Der größte Cluster zeigt zwar auf Apple Inc., die vier weiteren Cluster jedoch zu Firmen, die Dedicated-Server in Russland und auf den Seychellen vermieten. Fügt man Apple zur Whitelist, hinzu ergibt sich eine Trefferquote von 61% der identifizierten Zieladdressen in Blacklists.

4 Abschlussbetrachtung

In dieser Arbeit haben wir BOTFINDER vorgestellt, ein neues netzwerkbasiertes System zur Erkennung von einzelnen Schadsoftwareinfektionen im Netzwerk. BOTFINDER lernt aus der Beobachtung von Schadsoftware, die in kontrollierter Umgebung ausgeführt wird, erzeugt daraus Modelle und nutzt diese, um Bots ohne Analyse von Paketinhalten zu identifizieren. Hierbei erreicht es Erkennungsraten von nahezu 80% bei niedriger Falsch-Positiv-Rate.

Die hier sehr kurz vorgestellten Ergebnisse sind auf der ACM CoNEXT 2012 im Paper "BotFinder: Finding Bots in Network Traffic Without Deep Packet Inspection" [TFVK12] ausführlicher präsentiert worden und basieren auf der Dissertation "Content Agnostic Malware Detection in Networks" [Teg12].

Literatur

[BCH+09] Ulrich Bayer, Paolo M. Comparetti, Clemens Hlauschek, Christopher Kruegel und Engin Kirda. Scalable, Behavior-Based Malware Clustering. 2009.

[BKK08] Ulrich Bayer, Christopher Kruegel und Engin Kirda. Anubis: Analyzing Unknown Binaries. In *http://anubis.iseclab.org/*, 2008.

[Cla04] B. Claise. Cisco Systems NetFlow Services Export Version 9. RFC 3954", Internet Engineering Task Force, 2004.

[DRSL08] Artem Dinaburg, Paul Royal, Monirul Sharif und Wenke Lee. Ether: malware analysis via hardware virtualization extensions. In *Proceedings of the 15th ACM conference on Computer and communications security*, CCS '08, Seiten 51–62, New York, NY, USA, 2008. ACM.

[3] RBLS: `http://rbls.org/`

[FSP+06] Prahlad Fogla, Monirul Sharif, Roberto Perdisci, Oleg Kolesnikov und Wenke Lee. Polymorphic blending attacks. In *Proceedings of the 15th conference on USENIX Security Symposium - Volume 15*, Berkeley, CA, USA, 2006. USENIX Association.

[GH07] Jan Goebel und Thorsten Holz. Rishi: Identify Bot Contaminated Hosts by IRC Nickname Evaluation. In *HotBots'07: Proceedings of the First Workshop on Hot Topics in Understanding Botnets*, Seiten 8–8, Berkeley, CA, USA, 2007. USENIX Association.

[GPY+07] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong und Wenke Lee. BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation. In *Proceedings of the 16th USENIX Security Symposium (Security'07)*, August 2007.

[GPZL08] Guofei Gu, Roberto Perdisci, Junjie Zhang und Wenke Lee. BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection. In *Proceedings of the 17th USENIX Security Symposium (Security'08)*, 2008.

[GZL08] Guofei Gu, Junjie Zhang und Wenke Lee. BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic. In *Proceedings of the 15th Annual Network and Distributed System Security Symposium (NDSS'08)*, February 2008.

[RZMT06] Moheeb A. Rajab, Jay Zarfoss, Fabian Monrose und Andreas Terzis. A Multifaceted Approach to Understanding the Botnet Phenomenon. In *IMC '06: Proceedings of the 6th ACM SIGCOMM on Internet measurement*, Seiten 41–52. ACM Press, 2006.

[SBY+08] Dawn Song, David Brumley, Heng Yin, Juan Caballero, Ivan Jager, Min Kang, Zhenkai Liang, James Newsome, Pongsin Poosankam und Prateek Saxena. BitBlaze: A New Approach to Computer Security via Binary Analysis. In R. Sekar und Arun Pujari, Hrsg., *Information Systems Security*, Jgg. 5352 of *Lecture Notes in Computer Science*, Seiten 1–25. Springer Berlin / Heidelberg, 2008.

[Teg12] Florian Tegeler. *Content Agnostic Malware Detection in Networks*. Dissertation, University of Gttingen, Germany, `http://ediss.uni-goettingen.de/handle/11858/00-1735-0000-000D-F068-1`, 2012.

[TFVK12] Florian Tegeler, Xiaoming Fu, Giovanni Vigna und Christopher Kruegel. BotFinder: Finding Bots in Network Traffic without Deep Packet Inspection. In *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, CoNEXT '12, Seiten 349–360, New York, NY, USA, 2012. ACM.

[WBH+09] Peter Wurzinger, Leyla Bilge, Thorsten Holz, Jan Goebel, Christopher Kruegel und Engin Kirda. Automatically Generating Models for Botnet Detection. In *14th European Symposium on Research in Computer Security (ESORICS), Lecture Notes in Computer Science, Springer Verlag*, September 2009.

[WHF07] Carsten Willems, Thorsten Holz und Felix Freiling. Toward Automated Dynamic Malware Analysis Using CWSandbox. *IEEE Security and Privacy*, 5:32–39, 2007.

Florian Tegeler studierte Informatik (BSc/MSc) und Physik (Diplom) in Göttingen und promovierte von 2008 bis 2012 am dortigen Institut für Informatik. Über längere Auslandsaufenthalte an der University of Cambridge, der Columbia University in New York und der University of California in Santa Barbara konnte er seine verschiedenen Interessen von Internet Security über Netzwerktechnik zu Social Network Analysis verfolgen.