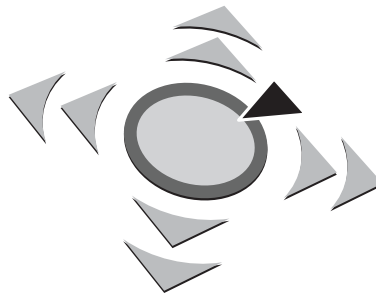


3. GI FG SIDAR Graduierten-Workshop über  
Reaktive Sicherheit

# SPRING

Ulrich Flegel, Thorsten Holz (Hrsg.)

08. August 2008, Mannheim



SIDAR-Report SR-2008-01

## Vorwort

SPRING ist eine wissenschaftliche Veranstaltung im Bereich der Reaktiven Sicherheit, die Nachwuchswissenschaftlern die Möglichkeit bietet, Ergebnisse eigener Arbeiten zu präsentieren und dabei Kontakte über die eigene Universität hinaus zu knüpfen. SPRING ist eine zentrale Aktivität der GI-Fachgruppe SIDAR, die von der organisatorischen Fachgruppenarbeit getrennt stattfindet. Die Veranstaltung dauert inklusive An- und Abreise einen Tag und es werden keine Gebühren für die Teilnahme erhoben. SPRING findet ein- bis zweimal im Jahr statt. Die Einladungen werden über die Mailingliste der Fachgruppe bekanntgegeben. Interessierte werden gebeten, sich dort einzutragen (<http://www.gi-fg-sidar.de/list.html>). Für Belange der Veranstaltung SPRING ist Ulrich Flegel (SAP Research Center Karlsruhe) Ansprechpartner innerhalb der Fachgruppe SIDAR.

Nach der Premiere in Berlin fand SPRING in Dortmund und Mannheim statt. Die Vorträge deckten ein breites Spektrum ab, von noch laufenden Projekten, die ggf. erstmals einem breiteren Publikum vorgestellt werden, bis zu abgeschlossenen Forschungsarbeiten, die zeitnah auch auf Konferenzen präsentiert wurden bzw. werden sollen oder einen Schwerpunkt der eigenen Diplomarbeit oder Dissertation bilden. Die zugehörigen Abstracts sind in diesem technischen Bericht zusammengefaßt und wurden über die Universitätsbibliothek Dortmund elektronisch, zitierfähig und recherchierbar veröffentlicht. Der Bericht ist ebenfalls über das Internet-Portal der Fachgruppe SIDAR zugänglich (<http://www.gi-fg-sidar.de/>). In dieser Ausgabe finden sich Beiträge zur den folgenden Themen: Intrusion Detection, Malware und Mobile Systeme.

Wir danken allen, die mitgeholfen haben.

Mannheim, August 2008

Ulrich Flegel, Thorsten Holz

## Contents

Malware-Klassifizierung und -Clustering mit MIST <i>Philipp Trinius</i> . . . . .	4
Towards Clustering of Malware Behavior <i>Martin Apel</i> . . . . .	5
Online Malware Identification Through Dynamic Trace Detection <i>Clemens Kolbitsch</i> . . . . .	6
Aspects of In-Vehicle Security <i>Michael Müter</i> . . . . .	7
Anmerkungen zur Verwundbarkeit mobiler Web-Browser <i>Michael Becher</i> . . . . .	8
Anomaly Detection on Smartphones <i>Aubrey-Derrick Schmidt</i> . . . . .	9
A hierarchical approach to intrusion detection and response in MANETs <i>Stephan Schmidt</i> . . . . .	10
CIMD- Collaborative Intrusion and Malware Detection <i>Rainer Bye</i> . . . . .	11
P2P-basierte Analyseverteilung für Intrusion Detection Systeme <i>Michael Vogel</i> . . . . .	12
Removing Web Spam Links from Search Engine Results <i>Manuel Egele</i> . . . . .	13
Flow analysis in the security context <i>Tobias Limmer and Falko Dressler</i> . . . . .	14
Verifikation von Signaturen - Spezifikationsfehlern auf der Spur <i>Sebastian Schmerl</i> . . . . .	15
Anomaly-based Intelligent Intrusion Prevention <i>Tammo Krueger</i> . . . . .	16

Diesen Bericht zitieren als:

Ulrich Flegel, Thorsten Holz, editors. Proceedings of the Third GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2008-01, GI FG SIDAR, Mannheim, August 2008,

Beiträge zitieren als:

Autor. Titel. In Ulrich Flegel, Thorsten Holz, editors, Proceedings of the Third GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2008-01, page xx. GI FG SIDAR, Mannheim, August 2008.

# Malware-Klassifizierung und -Clustering mit MIST

Philipp Trinius

Universität Mannheim, D-68159 Mannheim  
philipp.trinius{at}informatik.uni-mannheim.de

Aktuell wird bei der Erkennung und Klassifizierung von Malware größtenteils auf signaturbasierte Verfahren zurückgegriffen. Die dabei eingesetzten Signaturen beschreiben lediglich Byte-Sequenzen von als böse eingestuftem Code und lassen sich daher relativ einfach mit Hilfe von polymorphem Code oder über *Code-Obfuscation* aushebeln. Der an der Universität Mannheim verfolgte und hier vorgestellte Ansatz zur Klassifizierung von Malware setzt daher keine klassischen Signaturen ein sondern versucht, allein auf Basis des Malware-Verhaltens eine zuverlässige Aussage zu treffen. Dazu wird die Malware in der CWSandbox ausgeführt und das beobachtete Verhalten protokolliert. Erste Versuche einer automatisierten Klassifizierung von Malware auf Basis dieser Reports brachte durchaus vielversprechende Ergebnisse [Rieck et al., DIMVA '08].

Um die maschinelle Verarbeitung der Daten zu optimieren und somit eine noch zuverlässigere Klassifizierung sowie *sauberes* Clustering der Malware zu erreichen, wurde MIST – Malware Instruction Set – entwickelt. Dabei handelt es sich um eine eigens zu diesem Zweck entwickelte Sprache, die das Verhalten von Malware abbilden kann. Die folgende Auflistung zeigt in (a) die CWSandbox- und in (b) die MIST- Codierung des API-Befehls zum Laden einer Bibliotheks-Datei.

(a) `<load_dll filename="H:\WINDOWS\system32\kernel32.dll" successful="1"  
address="&#x24;7C800000" end_address="&#x24;7C907000" size="1077248"/>`

(b) `02|02||00107000||02|0002|000011|0000||7C800000||7C907000||1||`

In MIST entsprechen die einzelnen Attribute hexadezimalen Werten und werden durch eine doppelte Pipe voneinander getrennt. Während numerische Attribute, wie die Dateigröße, einfach hex-codiert übernommen werden, werden die Werte der übrigen Attribute, z.B. Dateinamen, in Tabellen eingetragen. In die MIST-Darstellung wird in diesem Fall lediglich der entsprechende Index übernommen. Neben der verkürzten Darstellung erfolgt eine Ordnung der Attribute. Dabei werden die Parameter der API-Befehle “entsprechend ihrer Konstanz” von links nach rechts angeordnet, wobei sehr variable oder für die Klassifizierung irrelevante Attribute vollständig entfallen oder ihre Werte ersetzt werden können. Beispielsweise werden Prozess-IDs durch den Dateinamen (inkl. Pfad) des Prozesses ersetzt. Für obiges MIST-Wort ergibt sich somit die folgende Sortierung: `load_dll|size|filename|address|end_address|successful`. Die Sortierung nach “Konstanz” wird auch für Pfadangaben enthaltende Attributwerte übernommen. Diese werden nach `extension|path|filename|paramter` sortiert. Das Ziel der Sortierung ist in beiden Fällen, verschiedene Granularitäten bei der Klassifizierung und dem Clustering der Reports zu ermöglichen.

Als Input werden zur Zeit CWSandbox-Reports in XML-Darstellung verwendet. Neben diesen Reports benötigt der Konvertierungsalgorithmus lediglich die dazugehörige Schemabeschreibung. Das MIST-Alphabet selbst baut sich erst während der Konvertierung der Reports auf, indem es beim Auftreten unbekannter Attributwerte um diese erweitert wird.

Um die Eignung von MIST zur Klassifizierung und dem Bilden von Malware-Familien zu bestimmen, wurden die MIST-Reports eines Test-Sets im ersten Schritt mit Hilfe der *Edit-Distance* und dem *Fuzzy hashing*-Algorithmus *ssdeep* untersucht. Im Vergleich mit Analysen auf den CWSandbox-Reports wurden die Malware-Binaries mit einem bedeutend höheren Prozentsatz ihren Familien zugeordnet, und auch der Abstand zwischen den einzelnen Familien war prozentual größer.

# Towards Clustering of Malware Behavior

Martin Apel\*

\* TU Dortmund

D-44221 Dortmund

`martin.apel{at}udo.edu`

Bösartige Software (im folgenden Malware) ist eine der größten Bedrohungen des Internets. War das Schreiben neuer Malware bis vor einigen Jahren noch Experten vorbehalten, existieren heute Baukastensysteme, mit denen sich *individuelle* Malware leicht und ohne Expertenwissen erstellen lässt. Die momentane Erkennungsmethodik stützt sich im wesentlichen auf Signaturen ab, welche Byte-strings der Binaries enthalten. Um der Erkennung zu entgehen, werden von den Malware-Autoren daher Techniken wie Polymorphie und Metamorphie verwendet, so daß es von einer Malware viele *Erscheinungsformen* gibt. Dies erschwert die Erkennung von Malware und erhöht die für die Erkennung benötigte Anzahl von Signaturen enorm. Um der wachsenden Flut von Malware (und Signaturen) Herr zu werden, werden daher neue Verfahren benötigt.

Ein vielversprechender Ansatz ist die Konzentration auf das Verhalten von Malware, denn obwohl die oben genannten Techniken das Erscheinungsbild verändern, bleibt doch das Verhalten der Malware weitestgehend gleich. Um die Anzahl der Signaturen möglichst klein zu halten, können Familiensignaturen verwendet werden, welche auch Varianten einer Malware erkennen. Es werden also Verfahren benötigt, welche Malware anhand ihres Verhaltens in Familien gruppieren.

Dieser Ansatz soll auch im Projekt *Amsel* (Automatisch Malware Sammeln und Erkennen Lernen) verfolgt werden. Hierbei sollen Verfahren aus dem Bereich des Unüberwachten Lernens und insbesondere Verfahren zur Clusteranalyse verwendet werden. Diese Verfahren finden zu einer gegebenen Menge von Datenpunkten Gruppierungen (Cluster) ähnlicher Datenpunkte. Für die Clusteranalyse wird eine Abstandsfunktion benötigt, die den Abstand bzw. die Ähnlichkeit zwischen zwei Verhaltenssequenzen charakterisiert.

Für diese Abstandsfunktion stehen mehrere Kandidaten zur Verfügung, welche sich mehr oder minder stark unterscheiden. Da Verhalten sequentiell ist, bieten sich Abstandsfunktionen für sequentielle Daten, insbesondere die Levenshtein Distance oder die *NCD* (Normalized Compression Distance [2]), an. Eine gänzlich andere Möglichkeit besteht darin, die Verhaltenssequenzen mit Hilfe von Häufigkeitsanalysen in einen hochdimensionalen Vektorraum einzubetten ([1]) und eine der vielen über Vektorräumen definierten Abstandsfunktionen zu verwenden (Euklid, Manhattan, Mahalanobis). Die Verfahren über Vektorräumen arbeiten im allgemeinen schneller, betrachten jedoch nur Häufigkeiten von n-Grammen oder Token.

In dem Vortrag sollen die Vor- und Nachteile einiger Abstandsfunktionen für die Verwendung bei der Clusteranalyse von Malware Verhalten beleuchtet werden.

## Literatur

- [1] Efficient Algorithms for Similarity Measures over Sequential Data: A Look Beyond Kernels. K. Rieck, P. Laskov, K.-R. Müller. Pattern Recognition, 28th DAGM Symposium, 374-383, September 2006
- [2] Clustering by compression, Cilibrasi, R.; Vitanyi, P.M.B., Information Theory, IEEE Transactions on, Volume 51, Issue 4, April 2005 Page(s): 1523 - 1545

# Online Malware Identification Through Dynamic Trace Detection

Clemens Kolbitsch\*

\*International Secure Systems Lab  
Technical University Vienna  
ck@iseclab.org

Current means for online detection of malware, such as botnet clients, viruses, and worms, is two sided: One possibility is the use of a static binary signature, typically created by a human analyst, that is compared to the executable of an unknown binary. Another, rather new, method for the detection of executing malware is finding known patterns in traces of system- or API calls.

Whereas the creation of call trace signatures, necessary for the latter approach, can be automated to some extent [1, 2], generating binary signatures is still a time-consuming and error-prone process. Furthermore, both mechanisms share the problem of obfuscation. Polymorphism and metamorphism have become common practice in malware concealment to complicate binary detection. Alike, including ineffectual system calls can be used to trick the latter detection mechanism.

In our current work, we try to circumvent these practices by identifying an executing malware by its immutable execution characteristics. We claim that each malware family (i.e. all possible polymorphic and metamorphic variants of one malware sample) shares a common *trace* signature. We define a trace to be the *algorithm used* by the malware sample *to transform* arbitrary *input* (such as private information read from the filesystem, recorded keystrokes invoked by the user, or the malware's own image used for replication) *into output data* that is sent out over a network connection (or stored into a temporary file for latter transmission).

Our detection works by storing trace signatures of various malware samples. Using these previously generated signatures, we rerun the candidate algorithms on the input processed by a suspicious binary and compare their output with the output produced by the executable under observation. If the observed and generated outputs share salient characteristics, we can issue an alert to the system's user or take adequate actions (like stopping the malicious binary).

The prototype implementation of our trace extraction is based on a modified version of the dynamic analysis framework TTAalyze/Anubis [3]. In a first step, we extract the transformation algorithm by observing the execution on tainted input data by a (possibly packed) malware sample. Simultaneously, we record the executed instructions and observed control flow inside the binary. In a second step, we use this information, to extract *slices* containing only the transformation-relevant pieces of the malware code and reconstruct DLLs containing the transformation algorithm.

In a last step, these DLLs can be used by an online scanner for replaying the input-output transformation algorithm and performing the detection of common patterns with the

## References

- [1] Manuel Egele, Christopher Kruegel, Engin Kirda, Heng Yin, and Dawn Song: *Dynamic Spyware Analysis*, in *USENIX Annual Technical Conference*, 2007.
- [2] Heng Yin, Dawn Song, Manuel Egele, Christopher Kruegel, and Engin Kirda: *Panorama: Capturing System-wide Information Flow for Malware Detection and Analysis*, in *14th ACM Conference on Computer and Communications Security*, 2007.
- [3] Ulrich Bayer, Andreas Moser, Christopher Kruegel, and Engin Kirda: *Dynamic Analysis of Malicious Code*, in *Journal in Computer Virology, Springer Computer Science*, 2006.

# Aspects of In-Vehicle Security

Michael Müter

Daimler AG

Stuttgart, Germany

michael.mueter{at}daimler.com

The vehicles of today are becoming more and more complex and can comprise over 60 electronic control units, different domains and gateways. Numerous interfaces like USB, SD card, GSM or Bluetooth allow communication with the outside world. Future developments like the introduction of vehicular communication networks [C2C08] even enhance the trend of the vehicle shifting from a closed architecture to a more open and exposed system.

Hand in hand with this development the risk that attackers try to get access to the vehicle network in order to exploit the system is continuously rising. Attacker scenarios could be ranging from theft of confidential information like planned routes or stored address book entries up to the interference with vital vehicle functions like brakes or engine control. Therefore, vehicular IT security has gained increasing attention in recent years [LPW05, HKLD07, SE08].

As vehicular systems are very different to standard desktop computers the well-known protection measures cannot be used immediately within this context. Typical bus systems that are used within the vehicle are CAN, MOST and Flexray, each of which has its own particular concept and technical specialization. Open challenges like how to protect the vehicular system against attacks and how can a vehicular architecture justifiably be tagged secure need to be considered within this area. Due to the long life span and the safety-critical functionality of vehicles, especially the question how to detect malicious intrusions in real-time is considered highly important. Vehicular intrusion detection systems may be able to provide one possible answer to this challenge.

## References

- [LPW05] Kerstin Lemke, Christof Paar and Marko Wolf *Embedded Security in Cars: Securing Current and Future Automotive IT Applications 2005*, Springer-Verlag New York, Inc.
- [HKLD07] Tobias Hoppe, Stefan Kiltz, Andreas Lang and Jana Dittmann *Exemplary Automotive Attack Scenarios: Trojan horses for Electronic Throttle Control System (ETC) and replay attacks on the power window system; Automotive Security - VDI-Berichte Nr. 2016, Proceedings of the 23. VDI/VW Gemeinschaftstagung Automotive Security*, Wolfsburg, Germany; 27.-28. November 2007, VDI-Verlag.
- [C2C08] *C2C - Car 2 Car Communication Consortium*, <http://www.car-to-car.org/>
- [SE08] *SEVECOM Project - Secure Vehicle Communication*, <http://www.sevecom.org/>

# Anmerkungen zur Verwundbarkeit mobiler Web-Browser

Michael Becher

Universität Mannheim, becher@informatik.uni-mannheim.de

Die Sicherheit mobiler Endgeräte mit SIM-Karte und durch Fremdsoftware erweiterbarem Betriebssystem (*Smartphones*) ist heute ein aktives Feld, denn die mobile Welt unterscheidet sich von der heutigen PC-Welt durch die immer vorhandene Möglichkeit, schnell hohe Kosten für den Mobilfunkbenutzer zu erzeugen. Die einfachste Maßnahme zur Erhöhung der Sicherheit ist das vollständige Schließen des Betriebssystems gegenüber Fremdsoftware. Doch die Erfahrung hat gezeigt, dass dies keine durchsetzbare Lösung ist.<sup>1</sup> Ein anderer Ansatz stellt Funktionen nur über Anwendungsrahmenwerke bereit. Beispiele sind J2ME oder Dotnet, die nicht alle Funktionen des Betriebssystems zugreifbar machen. Eine weitere Maßnahme ist das kryptographische Signieren von Anwendungen, die den Zugriff auf sicherheitskritische Funktionen oder sogar deren Benutzung ohne Rückfrage an den Benutzer erlauben.

Neue Entwicklungen in der Mobilfunkwelt rücken den mobilen Web-Browser in eine exponierte Position. Einerseits nimmt die Nutzung von mobilen Datendiensten stetig zu und damit auch die Nutzung des mobilen Browsers. Andererseits werden mobile Browser zur Zeit mit vielen Funktionen erweitert, die sie selber zu einem vollständigen Anwendungsrahmenwerk machen.<sup>2</sup> Die Erfahrung zeigt, dass bei solchen Aktivitäten zumeist die Funktionalität der Sicherheit untergeordnet wird. Der mobile Browser wird damit ein weiterer Angriffsvektor für das mobile Endgerät. Beispiele für erfolgreich ausgenutzte Schwachstellen sind verschiedene Denial-of-Service-Angriffe auf den Internet Explorer Mobile<sup>3</sup> und prominent der Jailbreak des iPhones für Firmware-Version 1.1.1, der trotz Behebung der alten Umgehung die Installation von Fremdsoftware wieder ermöglichte, allein durch Aufruf einer Web-Seite.<sup>4</sup>

Es stellen sich dabei zwei Fragen: 1. Können gefundene Schwachstellen des Browsers effizient behoben werden? 2. Wie kann der Benutzer Schaden gegen seine persönlichen Schutzinteressen abwenden?

Bei aktuellen Smartphones ist die entfernte Behebung von Schwachstellen möglich. Das ist zum einen die entfernte Behebung angestoßen durch Benutzeraktion (am PC oder auf dem Gerät laden und Aktualisierung ausführen) oder angestoßen durch den Mobilfunkbetreiber (Push). Die erste Frage kann somit für aktuelle Smartphones bejaht werden.

Die zweite Frage ist nicht leicht zu beantworten. Denn der Benutzer sieht sich einer Vielzahl von Technologien mit ihren eigenen Schutzmechanismen gegenüber, aktuell die Initiative „BONDI“<sup>5</sup>, die auch zum Ziel hat, den mobilen Browser zu einem Anwendungsrahmenwerk auszubauen, allerdings mit deutlichem Fokus auf der Sicherheit des Benutzers. Vielleicht wäre eine Lösung besser, die dem Benutzer die vollständige Macht über sein Gerät gibt bezüglich seiner Sicherheitsinteressen, und zwar unabhängig von einer speziellen Technologie, geräteweit in der Art einer erweiterten persönlichen Firewall zum Schutz vor finanziellem Schaden und vor dem Verlust der Vertraulichkeit seiner persönlichen Informationen auf dem Mobiltelefon. Doch das ist nicht mehr Teil der reaktiven Sicherheit und soll deshalb an anderer Stelle vertieft werden.

<sup>1</sup>Orange SPV: <http://developer.orangews.com/orgspv/comdefq.aspx>,  
iPhone: <http://www.heise.de/newsticker/meldung/97556>

<sup>2</sup>Opera Widgets: <http://www.opera.com/products/mobile/widgets/>

<sup>3</sup>z.B. <http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-0685>

<sup>4</sup><http://jailbreakme.com>, <http://blog.metasploit.com/2007/10/cracking-iphone-part-1.html>

<sup>5</sup><http://www.omtp.org/News/Display.aspx?Id=9e9a5f27-a57e-4804-b736-379b1d98b169>

# Anomaly Detection on Smartphones

Aubrey-Derrick Schmidt\*

\* Technische Universität Berlin, DAI-Labor,  
Ernst-Reuter-Platz 7, 10587 Berlin, Germany  
aubrey.schmidt@dai-labor.de

This abstract outlines my ongoing research in the field of smartphone anomaly detection. Malware, e.g. like virus, worms, and Trojan horses, have been threats to computer systems for many years and it was only a question of time when the first malicious software writers would get interested in smartphone platforms, such as Symbian OS. In 2004, the first articles about malware for smartphones [1] appeared saying that the next generation of targets are mobile devices. Since then, the number of malwares increased every month, and variants for various platforms appeared [2].

Commercially available countermeasures to smartphone malware suffer from weaknesses since they rely on signature lists or static rules. Additionally, the currently used signature-based approaches leave user exposed to the malware threat until the signature is available where Bulygin [3] showed that in worst case a MMS worm targeting random phone book numbers can infect more than 700,000 devices in about three hours. Therefore, it is crucial to detect malware presence and activity as fast as possible. Monitoring-based anomaly detection systems can be a valuable addition to signature-based approaches for achieving this since they do not rely on signatures and can indicate suspicious activity in real-time. Although still suffering from high false-alarm and low detection quality several approaches showed the functionality of such systems, e.g. [4].

My current research focuses on creating and evaluating host-based anomaly-detection systems for smartphones. First results show that these cannot be fully independent from remote systems analyzing the monitored data since capabilities, like CPU power and available memory, are still very limited. But as these capabilities are steadily increasing, I expect *capable* devices to be released in the next two years. Especially devices running open platforms, e.g. Android or OpenMoko, are of special interest since they provide full access to the operating system, which is necessary for implementing an effective host-based anomaly detection system. Therefore, an initial monitoring client will be developed on Android for showing its usability for implementing low-level security tools. The main problems that have to be handled are: decreasing device resource usage, decreasing false positive rates of the detection, and increasing detection quality. Furthermore, applicability of already existing Linux tools. e.g. Snort, Nagios, or OSSEC, will be checked.

## References

- [1] D. Dagon, T. Martin, and T. Starner, "Mobile phones as computing devices: The viruses are coming!" *IEEE Pervasive Computing*, vol. 3, no. 4, pp. 11–15, 2004.
- [2] A. Schmidt and S. Albayrak, "Malicious software for smartphones," Technische Universität Berlin, DAI-Labor, Tech. Rep. TUB-DAI 02/08-01, Feb. 2008, <http://www.dai-labor.de>.
- [3] Y. Bulygin, "Epidemics of mobile worms," in *Proceedings of the 26th IEEE International Performance Computing and Communications Conference, IPCCC 2007, April 11-13, 2007, New Orleans, Louisiana, USA*. IEEE Computer Society, 2007, pp. 475–478.
- [4] A.-D. Schmidt, F. Peters, F. Lamour, and S. Albayrak, "Monitoring smartphones for anomaly detection," in *MOBILWARE 2008, International Conference on MOBILE Wireless MiddleWARE, Operating Systems, and Applications*, Innsbruck, Austria, 2008.

# A hierarchical approach to intrusion detection and response in MANETs

Stephan Schmidt\*,†

\*,†DAI-Labor, Technische Universität Berlin

Over the last few years, intrusion detection has become one of the most important research areas in the field of network security. In addition, network access technologies are becoming more dynamic and pervasive. Unfortunately, in contrast to cable-bound network topologies, the task of detecting intrusions is quite a bit more complicated in mobile ad hoc networks due to their inherent characteristics: they do not possess a fixed underlying infrastructure and undergo constant topology changes. This leads to a lack of fixed concentration points where classical methods of traffic analysis could be applied.

Additionally, the specific characteristics of MANETs— such as a limited physical protection layer, volatile connectivity properties and the lack of a centralized infrastructure including certification and management authorities— allow new attack types, chief among them attacks on the underlying routing mechanisms which are susceptible to manipulation. Due to the multihop communication and the on-demand character of MANET routing protocols, participating malicious nodes may try to actively interfere with the internal routing mechanisms (for example by flooding false routing requests) or passively obstruct services by intentionally dropping data packets.

We propose to employ a hierarchical approach to intrusion detection and response in MANETs to face these challenges. Due to the lack of centralized authorities, an ID hierarchy of participating nodes needs to be created, maintained, and changed at runtime as nodes leave and enter the network. Nodes at the lowest level of the hierarchy are assigned the simplest tasks: gathering and aggregating packet-level data. This may be achieved by examining data packets designated to the node itself, data packets they are relaying to another node, or data packets they have observed by listening to the packet transmissions of neighboring nodes in promiscuous mode.

The aggregated traffic is passed on to nodes higher in the hierarchy which perform analysis of the aggregated traffic received from multiple lower-layer nodes. The results of the analysis may be passed on to a higher level if necessary. The depth of the hierarchy depends on several topology characteristics such as the number of nodes and their diversity in different hardware and software attributes.

When the analysis process is complete and potential malicious activity has been detected, an appropriate intrusion response needs to be carried out depending on the type of intrusion as well as the certainty of the conclusion based on the evidence. Examples of intrusion responses are precluding nodes from the network communication which are identified as compromised, reauthenticating nodes or simply notifying the end users of suspicious activity. The necessary intrusion response information is then passed back down through the hierarchy to the leaf nodes.

In our work, we focus on creating and maintaining the intrusion detection node hierarchy. In particular, we propose to use graph theory based algorithms such as betweenness centrality of nodes to elect likely clusterhead candidates in the hierarchy. Furthermore, the challenging task is to adapt the hierarchy to the constant topology changes. Depending on the rate of change, this may require fast online optimization techniques such as time approximation schemes.

# CIMD- Collaborative Intrusion and Malware Detection

Rainer Bye\*

\* Technische Universität Berlin, DAI-Labor  
Ernst-Reuter Platz 7, D-10587 Berlin, Germany  
rainer.bye[at]dai-labor.de

Computer networks are exposed to a variety of threats: zero-day attacks leave devices connected to the Internet susceptible to attacks because there are no appropriate signatures available during the vulnerability window. On the other hand, purely anomaly-based detection schemes capable of detecting new attacks are often of limited use due to a high false-positive rate.

Hence, intrusion detection and response can be considered a complex task. From the field of sociology we learn that teams respectively groups cope with complex tasks by their inherent cooperative character. There exist cooperative intrusion detection systems bypassing shortcomings of conventional approaches, but these are often limited to very specialized scenarios and do not take the configuration of other participating nodes into account [2, 1].

In this regard, I propose CIMD (Collaborative Intrusion and Malware Detection), a scheme for the realization of collaborative intrusion detection approaches. I argue that teams, respectively *detection groups* with a common purpose for intrusion detection and response, improve the measures against malware. By enabling participants to state their *objectives* (i.e. the aim of a detection group) and *interests* (i.e. the desired properties of the team members) an intrusion detection overlay is realized.

In our ongoing work, we contribute a taxonomy-based data model reflecting relevant properties of the participants of the overlay. We discuss each category in the taxonomy with regard to the impact on detection groups and their value for collaborative intrusion detection. Additionally, we provide a group formation algorithm taking objectives, interests, and maximum group size into account. Furthermore, we introduce the notion of *homogeneous* as well as *heterogeneous detection groups*, give concrete scenarios. The scenario for homogeneous groups is based on our previous work in [3]. Finally, we conduct a vulnerability analysis of the system.

The remaining problems include the evaluation of different ontology matching techniques for the group formation, the selection of an appropriate overlay network and common data exchange format as well as a suitable trust management approach. Eventually, we plan to carry out more detailed vulnerability analysis.

## References

- [1] Vinod Yegneswaran, Paul Barford, and Somesh Jha. Global intrusion detection in the DOMINO overlay system. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2004*, 2004.
- [2] Katja Luther, Rainer Bye, Tansu Alpcan, Sahin Albayrak, and Achim Müller. A Cooperative AIS Framework for Intrusion Detection. In *Proceedings of the IEEE International Conference on Communications, ICC 2007*, 2007.
- [3] Rainer Bye, Katja Luther, Seyit Ahmet Çamtepe, Tansu Alpcan, Şahin Albayrak, and Bülent Yener. Decentralized Detector Generation in Cooperative Intrusion Detection Systems. *Stabilization, Safety, and Security of Distributed Systems 9th International Symposium, SSS 2007 Paris, France, November 14-16, 2007 Proceedings*, Lecture Notes in Computer Science, Vol. 4838. Springer, 2008.

# P2P-basierte Analyseverteilung für Intrusion Detection Systeme

Michael Vogel\*

\* Brandenburgische Technische Universität Cottbus  
mvogel{at}informatik.tu-cottbus.de

Heutige Angriffe auf IT-Systeme werden hochgradig verteilt ausgeführt, indem tausende kompromittierte und fremdkontrollierte Rechner von Angreifern zu Botnetzen zusammengeführt werden, um diese für koordinierte Angriffe auf IT-Systeme zu nutzen. Dagegen arbeiten heutige Intrusion Detection Systeme (IDS) zur Erkennung dieser Angriffe meist separat und unkoordiniert um Endsysteeme eines zugeordneten Bereichs zu überwachen. Die Möglichkeiten, die die Kooperation von Rechnern bietet, sollten auch genutzt werden, um leistungsfähigere und fehlertolerantere IDS-Systeme zu entwickeln. Situationen, in denen durch IDS überwachte Systeme unter hoher Last stehen, führen, bedingt durch die auszuführenden aufwändigen Analysen der beobachteten sicherheitsrelevanten Ereignisse, zur verzögerten Erkennung von Angriffen, so dass Gegenmaßnahmen nicht mehr rechtzeitig möglich sind. Weiterhin werden diese Systeme in Lastsituationen durch das IDS zusätzlich gebremst, was meist nicht akzeptabel ist, so dass Beobachtungen (Auditdaten) dann verworfen werden müssen, um dieser Situation zu begegnen. Angreifer können Überlastsituationen gezielt herbeiführen, um ein IDS an der Erkennung von Angriffsspuren zu hindern. Um besser mit Überlastsituationen umzugehen, sollten IDS die Auditdatenanalyse verteilt ausführen, um die Analyselast, die für eine große Anzahl beobachteter Ereignisse erzeugt wird, dynamisch über die kooperierenden Rechner zu balancieren. Natürlich darf der zusätzliche Verwaltungsaufwand den Verteilungsgewinn nicht übersteigen. Bestehende IDS nutzen bisher nur die Möglichkeiten der Kooperation zur Verteilung von Informationen, wie z.B. Angriffswarnungen. Dagegen wurde noch nicht versucht, die aufwändigen Auditdatenanalysen zu verteilen.

Um eine hohe Ausfallsicherheit zu erreichen, ist es nicht zweckmäßig statische hierarchische Konzepte zur Delegierung von Analysen an übergeordnete Analyseserver anzuwenden. Diese stellen singuläre Punkte dar, deren Ausfall die Funktion des gesamten verteilten IDS gefährden, was durch Angreifer gezielt ausgenutzt werden kann. Statt dessen werden die Vorteile von P2P-Overlaynetzen genutzt, um ein Netz kooperierender Peers aufzubauen. Diese wirken in einem verteilten IDS zusammen, um Analyseressourcen anzubieten und um selbst, falls erforderlich, Ressourcen von Partnern zu nutzen. Peers können dem verteilten System dynamisch beitreten und es wieder verlassen.

Zur Umsetzung eines solchen verteilten IDS wird ein agentenbasiertes Konzept verwirklicht. Jeder Peer verfügt über einen IDS-Agenten. Diesem sind eine Menge von Sensoren zugeordnet, die sicherheitsrelevante Ereignisse im überwachten Bereich erfassen. Weiterhin verfügt ein Agent über Analyseeinheiten, die Auditdaten der Sensoren lokal analysieren. Über eine Kommunikationskomponente hält ein Agent Kontakt zu anderen Peers des Overlaynetzes und tauscht Informationen zu Lastsituationen und verfügbaren Ressourcen aus. Basierend auf diesen Informationen zu anderen Peers kann in Überlastsituationen die Delegierung von Analysen gesteuert werden.

Ein Schwerpunkt der Arbeiten liegt in der Analyse und Anpassung von Peer-to-Peer Algorithmen an die Anforderungen der verteilten Analyse. Dies betrifft u. a. Algorithmen zum Aufbau der Netztopologie, die Verbreitung von Informationen zu angebotenen Ressourcen und aktuelle Lastsituationen der Peers im P2P-Netz. So sollten z. B. als Partner für die Delegierung von Analysen möglichst physisch nahe liegende Peers gewählt werden, um Latenzzeiten zu minimieren. Vertrauen, Datenschutz bzw. die Pseudonymisierung von Beobachtungsdaten vor der Verteilung und die Prüfung der korrekten Analyseausführung sind weitere zu betrachtende Bereiche.

# Removing Web Spam Links from Search Engine Results

Manuel Egele\*

\*International Secure Systems Lab  
Technical University Vienna  
pizzaman@iseclab.org

Web spam denotes the manipulation of web pages with the sole intent to raise their position in search engine rankings[2]. Since a better position in the rankings directly and positively affects the number of visits to a site, attackers use different techniques to boost their pages to higher ranks. In the best case, web spam pages are a nuisance that provide undeserved advertisement revenues to the page owners. In the worst case, these pages pose a threat to Internet users by hosting malicious content and launching drive-by attacks against unsuspecting victims[1].

To mitigate these threats, we propose a system that is able to detect and remove web spam pages from search engine results. This task is basically a classification problem where result pages have to be classified as either spam or no spam. Spammers will most likely tweak those features (e.g., title tag of a page, keyword frequency in the body tag) that have the maximum impact on the ranking of a site. To learn which features are the most important, an experiment is conducted. This experiment consists of a number of web pages where each page has a unique feature distribution. Four different quantities of the features are considered in the experiment. A feature is either absent, present, present in elevated quantities, or present in spam quantities on any given page. Once the pages are created, they are published on individual domains. During the experiment, the position in the results is recorded for every page on regular intervals. Based on the ranking of these experiment pages, one can derive the importance of the selected features on the ranking algorithm. The necessity of conducting this experiment arises from the fact that search engine providers do not reveal the algorithms they use for ranking the results. This step can be seen as black box testing of ranking algorithms employed by search engine providers.

Based on the outcome of this experiment, we use the features to train a classification model. So called feature extractors are used to assign a quantitative measure for the feature under consideration to each page (e.g, the number or frequency of keywords present on the page). Multiple queries are submitted to a major search engine, and the results are manually labeled as either spam or no spam. This list of data is then divided into a training data set that is used to train the classifier, and a test data set that is used to evaluate the performance of the classifier.

By running the classifier on each search result page, spam pages, that are of no value to a search engine user, can be detected and removed from the listing. This creates room for other useful pages that fit the query, and more importantly, removes pages that might harm a visitor via drive-by attacks.

## References

- [1] N. Provos, P. Mavrommatis, M. A. Rajab, and F. Monrose. All Your iFRAMEs Point to Us. Technical report, Google Inc, 1600 Amphitheatre Parkway, Mountain View, CA, February 2008.
- [2] Z. Gyöngyi and H. Garcia-Molina. Web Spam Taxonomy. In *Adversarial Information Retrieval on the Web*, 2005.

# Flow analysis in the security context

Tobias Limmer and Falko Dressler

University of Erlangen-Nuremberg, Department of Computer Science 7  
{limmer,dressler}@informatik.uni-erlangen.de

Recent statistics in the Internet show high growth rates for computers infected by malicious applications, like so-called bots. These are installed on computers owned by unsuspecting users at home or within poorly maintained networks. A few years ago, malicious applications, particularly worms, propagated by using network-wide scans to detect vulnerable hosts, and infected them afterwards (popular examples are Code Red and Nimda). These scans have been easily detected by network monitoring equipment as often this type of malware exploited known vulnerabilities using the same attack pattern. So payload-based intrusion detection systems like Snort are well suited for the detection of these worms as they used signatures for different exploits to detect traffic belonging to these malicious applications.

Later generations of malware did not depend on network-wide scans to detect and infect vulnerable hosts [1]. Instead, they relied on social engineering techniques to persuade users to execute malware, or zero-day client-side vulnerabilities like those found in Internet browsers. Especially these kinds of exploits are not easily distinguishable from normal network traffic. This explains a current trend in security-related monitoring systems: not the exploit of vulnerable systems is being detected but aftereffects of a successful exploit are detected, e.g. installed and active versions of malware generate typical traffic patterns that may be recognized by intrusion detection systems.

Another challenge of current network monitoring systems are high data rates in networks that do not allow deep packet inspection to detect traffic patterns. So, available information is often reduced to network flows that offer an aggregated view to the monitored traffic. This type of data offers reduction rates of usually more than 20:1 in the standard setting and still allows easy detection of traffic anomalies. For best performance, flow aggregation is directly performed on hardware routers, which often support the export of flows in the IPFIX format, or its predecessors Netflow v9/v5.

Current malware often causes a high number of failed connections. The limited view offered by flow-based data only allows the use of heuristic methods to determine the state of a connection, e.g. if it was successfully established, abnormally terminated or blocked by a firewall. We performed several experiments in which we analyzed results gained by analyzing flow data with a packet-based TCP defragmentation module and determined which properties describe different connection states best.

For the experiments, we used our network monitoring toolkit Vermont that offers flow-based analysis of network streams. It is based on a modular structure that allows import and export of flow data in the standardized IPFIX protocol, as well as included packet payload using the PSAMP protocol. Main goal during the development of Vermont was to maintain full reconfigurability during execution of the program. This enables adaptive reaction to new demands in dynamic distributed environments.

## References

- [1] M. Allman, V. Paxson, J. Terrell, *A brief history of scanning*. Proceedings of 7th ACM SIGCOMM Conference on Internet Measurement (IMC 2007), San Diego, CA, USA, pp. 77-82, October 2007.

# Verifikation von Signaturen - Spezifikationsfehlern auf der Spur

Sebastian Schmerl<sup>†</sup>

<sup>†</sup>Brandenburgische Technische Universität Cottbus  
sbs{at}informatik.tu-cottbus.de

*Signaturbasierte* Intrusion-Detection-Systeme sind ein wichtiges Instrument für den reaktiven Schutz informationstechnischer Ressourcen. Sie ermöglichen eine automatische Erkennung und ggf. auch eine Abwehr von IT-Sicherheitsverletzungen. Dabei hängt die Wirksamkeit der Signaturanalyse entscheidend von der Genauigkeit der verwendeten Signaturen ab. Ungenaue Signaturen schränken die Erkennungsmächtigkeit stark ein und führen u. a. zu Fehlalarmen.

Die Ursachen der Erkennungsunsicherheit sind vielfältig, so werden beispielsweise Angriffe bzw. Audit-Daten fehlinterpretiert oder Verwundbarkeiten werden falsch eingeschätzt. Allerdings sind oftmals auch Fehler in der eigentlichen Signaturspezifizierung die Ursache. Dies ist besonders bei *Multi-Step-Signaturen* der Fall. Diese Signaturen erlauben die Spezifikation von Zusammenhängen bzw. Abhängigkeiten zwischen mehreren Protokolleinträgen / Audit-Ereignissen. Durch die größere Modellierungsfreiheit wird im Gegensatz zu *Single-Step-Signaturen* (z.B. Snort-Rules) eine genauere Spezifikation des Angriffs möglich. Leider unterlaufen dem Signaturmodellierer aber durch die größeren Modellierungsfreiheiten auch mehr *Spezifikationsfehler*. Diese Fehler werden dann meist erst in der Test- bzw. Korrekturphase der neuen Signatur erkannt, oftmals kommen aber Signaturen ungetestet zum Einsatz, so dass Spezifikationsfehler erst im realen Einsatz der Signatur festgestellt werden.

Die Test- und Korrekturphase von Signaturen, in welcher die Signaturen auf Korrektheit und Präzision gegen teilweise synthetisierte Angriffe getestet werden, ist zeit- und kostenintensiv. Dem Auffinden von reinen Spezifikationsfehlern kommt somit als vorgelagerte Phase zur Testphase eine essentielle Bedeutung zu. Mittels Methoden der formalen Verifikation können in Signaturen automatisiert typische Spezifikationsfehlern erkannt werden. Beispielsweise kann u.a. festgestellt werden, ob sich geforderte Bedingungen gegenseitig ausschließen, ob alternative oder nebenläufige Aktionsfolgen korrekt modelliert wurden oder ob Analyseabbrüche für nicht erfolgreich abschließbare Angriffe spezifiziert wurden.

Zur Analyse dieser Fragestellungen bieten sich hauptsächlich zwei Möglichkeiten an. Erstens können Signaturen in gefärbte *Petri-Netze* überführt werden, um dann auf Parameter wie Erreichbarkeit oder Lebendigkeit geprüft zu werden. Hierfür muss allerdings die meist deterministische Semantik von Signaturen in ein nichtdeterministisches Konzept überführt werden, mit dem Ziel vorhandene Petri-Netz-Tools wie *CPN-Tools* zur Verifikation einzusetzen. Alternativ können die zu verifizierende Signaturen in *Promela-Modelle* überführt werden, um sie anschließend gegen LTL Formel zu prüfen. Dabei werden mittels *Linear Temporal Logic* die erwünschten oder unerwünschten Signatureigenschaften beschrieben. Für die eigentliche Verifikation wird dann der Model-Checkers *Spin* verwendet.

Der Beitrag beschreibt die skizzierten Verifikationsstrategien, deren Grenzen und offene Problemfälle.

# Anomaly-based Intelligent Intrusion Prevention

Tammo Krueger\*

\* Fraunhofer FIRST

D-12489 Berlin, Germany

tammo.krueger{at}first.fraunhofer.de

Anomaly detection techniques have been shown to be suitable for network intrusion detection: based on an automatically generated model of normality incoming traffic can be separated into anomalous and normal traffic. The majority of the proposed systems, which are based on anomaly detection, focus on the network level and therefore lack the option of real time reactive response mechanism.

Based on anomaly detection techniques developed in the MIND project the successor project ReMIND aims among other things at using these methods for intelligent adaptive defense. The project proposes two intrusion prevention mechanisms based on known and mature technologies: packet filters and application level firewalls.

Packet filters act as a network-based intrusion prevention system: by applying rules to packets the system decides based on a static repository of rules, which packets are dropped and which packets are accepted. This basic concept will be used and further refined as follows: A packet preprocessor for connection reassembling gathers meta data for a packet like connection status, payload so far, previous connections from the same origin et cetera. Since each protocol has a different model of normal usage the anomaly detection is split up for each monitored protocol.

Based on adaptive anomaly thresholding, incremental learning and further meta data from the packet preprocessor the system decides, whether this packet will be filtered, altered, redirected or dropped. This concept will be implemented in the netfilter firewall via libnetfilter\_queue as additional, self-adapting packet filtering mechanism.

The semantic level of a packet filter is limited by definition. To operate on a higher semantic level other techniques have to be exploited. A reverse proxy operates on the protocol level, i.e. incoming packets are assembled to requests and these requests are parsed and processed. As previously proposed anomaly detection can be used at this point to redirect suspicious traffic to specially hardened systems. These so called shadow systems mimic the production system and act as sensors, which process anomalous requests and can give feedback about the true nature of this request: When processing “good” requests the user gets the same result as he would get from the production system, only the time for the processing would be longer due to the lower performance of the hardened system. When processing an “evil” request, the hardened system cannot be compromised and the intrusion attempt is reported.

The feedback of the shadow system can be used as input for more differentiate preventive countermeasures: The goal of the learning task is to classify the anomalous requests into “good” and “evil” and use this information before redirecting the request to a costly shadow system. Requests which are classified as “evil” with a high confidence are dropped instantaneously. Other anomalous requests are redirected to a shadow system and the resulting feedback is used to update both the classifier and the anomaly detector.

The combination of network-based with host-based intrusion prevention mechanisms leads to a hybrid system in which both parts benefit from each other: By processing all anomalous requests the impact of false positives is diminished and by redirecting just a part of the whole traffic the performance of the shadow systems is improved.