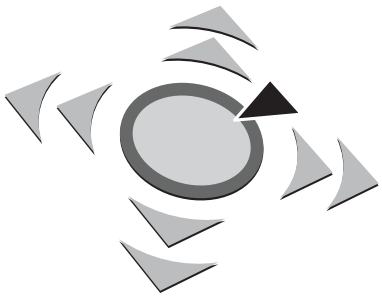
4. GI FG SIDAR Graduierten-Workshop über Reaktive Sicherheit

SPRING

Ulrich Flegel, Sandra Frings (Hrsg.)

14.-15. September 2009, Stuttgart



SIDAR-Report SR-2009-01

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 zwei Tage und es werden keine Gebühren für die Teilnahme erhoben. SPRING findet einmal jährlich 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 Abschlußarbeit 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 zu den folgenden Themen: neue Bedrohungen und Verwundbarkeitsanalyse, Malwareanalyse, Smartphone-Malware, Intrusion und Fraud Detection, und Datenflußkontrolle.

Wir danken allen, die mitgeholfen haben, besonders Cristina Fortu und Stanislaus Stelle.

Stuttgart, September 2009 — Ulrich Flegel, Sandra Frings

Contents

Countering Lifetime Kernel Code Integrity Protections
Ralf Hund 4
Botnetzmonitoring – Waledac
Ben Stock, Markus Engelberth, Jan Göbel 5
ConFlood - a non-distributed DoS-Attack
Kjell Witte und Florens Wasserfall 6
Visualisierung von Malware-Verhalten
Philipp Trinius 7
EDL als Ereigniskorrelationssprache in Multi-Sensor Intrusion Detection Systemen
Christoph Leuzinger 8
Smartphone Malware Evolution Revisited
*Aubrey-Derrick Schmidt** 9
Android Application Sandbox
Thomas Bläsing 10
Context-Awareness in Security Solutions
Leonid Batyuk 11
Continuous User Verification through Behavior Biometrics
*Arik Messerman** 12
Using entropy-analysis for shellcode detection
*Michael Gröning *† Jan Kohlrausch ** 13
Detecting Bots with Automatically Generated Network Signatures
Peter Wurzinger 14
Evaluation von Lernverfahren zur Bewertung der Gefahrenlage im Internet
Simon Hunke 15
Generierung von Signaturen mittels statischer Kontrollflussanalyse
René Rietz 16
Modeling Fraud Scenarios in a RETE-based Stateful Rule Engine
Cristina Fortu and Ulrich Flegel 17
Automatic Generation of Separation of Duty Fraud Scenrios
Stanislaus Stelle and Ulrich Flegel 19
Design and Application of a Security Analysis Method for Healthcare Telematics in Germany (HatSec)
Ali Sunyaev 20
Kann Data Leakage Prevention vor Datenoffenlegung schützen?
Matthias Luft 21

Diesen Bericht zitieren als:
Ulrich Flegel, Sandra Frings, editors. Proceedings of the Fourth GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2009-01, GI FG SIDAR, Stuttgart, September 2009,

Beiträge zitieren als:
Autor. Titel. In Ulrich Flegel, Sandra Frings, editors, Proceedings of the Fourth GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2009-01, page xx. GI FG SIDAR, Stuttgart, September 2009.

Countering Lifetime Kernel Code Integrity Protections

Ralf Hund
University of Mannheim
Laboratory for Dependable Distributed Systems
D-68159 Mannheim, Germany
hund{at}uni-mannheim.de

Attacks against the very core of an operating system, the so-called *kernel*, have become a significant threat nowadays. Such attacks often come in the form of a *rootkit*, a piece of malware that tries to hide system entities, such as files or sockets, to all, or certain, processes. Therefore, protecting the kernel of an operating system against attacks, especially injection of malicious code, is an important factor for implementing secure operating systems.

To address this problem, several kernel code integrity protection mechanisms were proposed recently that aim to *prevent* malicious programs from being executed with elevated privileges. Programs such as NICKLE [1] achieve this by means of code authentication and enforcing code integrity at all times. Therefore, a special technique called *memory shadowing* is introduced. By doing so, NICKLE enforces permanent code integrity, meaning that an attacker may not overwrite existing code or smuggle-in new code and subsequently execute it with elevated privileges. Other solutions achieve the same goal by enforcing the WX property on all relevant memory pages [2]. However, all code integrity based protections share a common shortcoming: they are based on the assumption that an attacker must always execute own code to carry out her attack.

While the infamous *return-to-libc* attack has already proven this assumption to be problematic, a recently introduced technique called *return-oriented programming* [3] pushes this kind of attack to a new level. Thereby, an attacker may execute *arbitrary* computations by chaining together several code-chunks of already existing instructions without ever executing new code.

To assess the feasibility of such an attack in the given domain, we have implemented a framework that fully automates the process of spotting and incorporating instruction sequences within the existing kernel code that can be used by an attacker for malicious computations. With the aid of this tool, we created two rootkits for Windows operating systems that do not have a single own instruction.

We evaluated the system on different commodity operating systems to show the portability, universality, and limitations of our approach. On the whole, we show that it is insufficient to protect the operating system core only by means of code integrity and that an attacker does not need own code in order to do arbitrary computations.

References

[1] Ryan Riley et al.: Guest-Transparent Prevention of Kernel Rootkits with VMM-Based Memory Shadowing (RAID '08)

[2] Arvind Seshadri et al.: SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes (ACM SIGOPS '07)

[3] Hovav Shacham: The Geometry of Innocent Flesh on the Bone: Return-into-libc without Function Calls (on the x86) (CCS '07)

Botnetzmonitoring – Waledac

Ben Stock, Markus Engelberth, Jan Göbel

Universität Mannheim
(stock|engelberth|goebel)@informatik.uni-mannheim.de

Seit Ende 2008 hat sich ein neuer Bot in die Riege der Internetschädlinge eingereiht – *Waledac*. Von einigen Seiten wurde bereits vermutet, dass es sich bei Waledac um den Nachfolger des bekannten *Storm*-Wurms handelt. Um diese Hypothese zu belegen und genauere Zahlen zur Größe des Botnetzes zu liefern, wurde im Rahmen einer Abschlussarbeit eine detaillierte Untersuchung des Waledac-Netzes durchgeführt. Anders als Storm setzt Waledac auf HTTP als Kommunikationskanal. Dadurch ist Waledac auch auf solchen Rechner funktionsfähig, auf denen lediglich Verbindungen zu Standardports erlaubt sind.

Die dynamische Analyse ergab, dass es sich beim Waledac nicht um ein reines Peer-to-Peer-Botnetz handelt, sondern dass im Hintergrund weiterhin zentrale Server vorhanden sind. Bei den infizierten Rechnern wird zwischen *Spammern* und *Repeatern* unterschieden. Repeater zeichnen sich dadurch aus, dass sie über eine öffentliche IP-Adresse erreichbar sind. Diese Rechner bilden das Herzstück des Botnetzes und agieren als Proxies für die Spammer. Die Spammer, die nicht über eine eigene öffentliche IP-Adresse erreichbar sind, werden dabei zum Versenden von Spam und Ausführen von Denial-of-Service-Attacken genutzt.

Da der größte Teil der Rechner im Waledac-Netz Spammer sind, musste eine Lösung gefunden werden, die Zahl dieser Rechner zu bestimmen. Mit Hilfe der in der statischen Analyse gewonnen Informationen wurde daher ein Klon von Waledac entwickelt, der lediglich die Kommunikationsfunktionen des Bots implementiert: *Walowdac*. Zusätzlich wurde ein Crawler programmiert, der die Repeater im Botnetz rekursiv arbeitet. Um zu jedem Zeitpunkt aktuelle Repeaterlisten zu haben, tauschen Spammer und Repeater diese Listen regelmäßig untereinander aus. Dazu sendet ein Spammer eine POST-Anfrage an den Repeater, deren Inhalt die dem Spammer bekannte Liste ist. Diese enthält ebenfalls den aktuellen Zeistempel sowie den der letzten erfolgreichen Verbindung zum jeweiligen Repater in der Liste. Der Repeater zieht diese Liste und seine eigene heran, um sie – anhand der Zeitstempel – zu einer aktualisierte Liste zu verschmelzen. Diese speichert der Repeater ab und sendet sie ebenfalls zurück an den Spammer. Dies wurde genutzt, um mit Hilfe des Crawlers die IP-Adressen von Walowdac im Netz zu propagieren. Diese Methode erwies sich als äußerst erfolgreich. Im Rahmen der Beobachtungen wurden am 22. Juli insgesamt 129.449 Bots gezählt. Die Untersuchung zeigte auch, dass dabei verhältnismäßig wenige Verbindungen aus Nordamerika eingingen, obwohl die Verbreitungskampagnen teilweise direkt auf Benutzer aus den USA abzielten (z.B. durch eine Kampagne zum 4. Juli) und somit die Zahl der Bots als untere Schranke angesehen werden muss.

Zusätzlich werden vom Waledac – ähnlich wie beim Storm – Berichte über den Erfolg der Spam-Läufe an das Netz gesendet. Die Auswertung der vom Walowdac erhaltenen Daten zeigte, dass ca. 10 Prozent der versendeten E-Mails vom empfangenen Mailserver angenommen wurden. Ein Versuch von ESET (`https://www.eset.de/news/botnet-waledac-macht-keinen-sommerurlaub/`) zeigte, dass ein Waledac-infizierter Rechner etwa 6500 Spam-Mails pro Stunde verschickt. Im Rahmen der Beobachtungen wurden zu jeder Uhrzeit mindestens 10.000 aktive Spammer beobachtet, was dem Waledac-Netz eine Spamkapazität von 1,5 Milliarden Spam-Mails pro Tag verleiht.

ConFlood - a non-distributed DoS-Attack

Kjell Witte und Florens Wasserfall

Universität Hamburg, Department Informatik
6witte{at}informatik.uni-hamburg.de 6wasserf{at}informatik.uni-hamburg.de

Dass die Verfügbarkeit von Diensten ein ebenso zu schützendes Gut ist, wie Vertraulichkeit und Integrität, ist unumstritten. Während virtuelle Einbruchswerkzeuge in den letzten 15 Jahren aber zunehmend an Komplexität gewannen, setzten die Angreifer bei DoS-Attacken vor allem auf eine große Anzahl verteilter Systeme, die koordiniert an den Angriffen beteiligt sein müssen. Die Angriffstechniken sind dabei größtenteils gleich geblieben [BSI07].

Die Erforschung neuartiger DoS-Angriffe darf trotzdem keinesfalls vernachlässigt werden, denn es gibt immer wieder Schwachstellen in den heute eingesetzten Protokollen. Ziel der Arbeit war es deshalb, zu prüfen, ob DoS-Angriffe auch mit einem einfachen DSL-Anschluss möglich sind, und damit - aus Sicht des Angreifers - ein günstiges Kosten/Nutzen-Verhältnis haben.

Ein DoS-Angriff gegen einen Server zielt im allgemeinen auf vollständige Ausnutzung einer Ressource, etwa der Netzwerkkapazität, oder einer Hardwarekomponente ab. Unter Beachtung der Anforderung, den Angriff von nur einem Computer mit verhältnismäßig schlechter Netzwerkanbindung durchführen zu können, bot es sich an, eine künstlich beschränkte Ressource auszunutzen. Tatsächlich existiert eine bisher nicht dokumentierte Angriffsmöglichkeit. Verletzbar ist als Ergebnis der Forschungen die Beschränkung vieler Programme, nur eine maximale Anzahl gleichzeitiger Verbindungen zuzulassen, indem man die im TCP-Protokoll gegebenen Möglichkeiten nutzt, Daten sehr langsam zu übertragen. Fordert man von einem Server einen Datensatz (etwa eine Webseite) an, verwendet im TCP-Protokoll sehr kleine rwnd Werte und forciert eine möglichst hohe Round Trip Time, so lässt sich die Verbindung über einen sehr langen Zeitraum geöffnet halten. Baut man nun so viele Verbindungen gleichzeitig auf, wie der Server zulässt, nimmt er keine legitimen Anfragen mehr an und ist somit nicht mehr verfügbar.

Das Erkennen und Abwehren des skizzierten Angriffes ist, zumindest, solange er nicht modifiziert wird, relativ einfach. Solange der Angriff von nur einem Computer durchgeführt wird, ist eine hohe Anzahl gleichzeitiger, sehr langsamer Verbindungen zu einer Internetadresse ein sehr sicheres Zeichen für einen Angriff. Auf Netzwerkebene sollten moderne Sicherheitssysteme einen solchen Angriff bereits aufgrund dieser Symptome erkennen und abwehren, ohne explizit für diesen Angriff konfiguriert zu werden. Auf Anwendungsebene sind Erkennung und Abwehr ebenfalls möglich, müssen aber durch die eingesetzen Serverprogramme aktiv unterstützt werden.

Generell ist davon auszugehen, dass die meisten, momentan im Internet erreichbaren (Web–)Server für diesen Angriff verwundbar sind. Größere, professionell verwaltete Netzwerke, verfügen bereits über Schutzmechanismen, die auch einen solchen Angriff verhindern können. Es bleibt aber eine Vielzahl kleiner und mittlerer Netzwerke, für die ein veritables Risiko besteht, da der Angriff von Einzelpersonen ohne nennenswerte Infrastruktur durchgeführt werden kann.

ConFlood stellt somit nicht unbedingt eine neue Qualität bei DoS-Angriffen dar, zeigt aber drastisch auf, welche Möglichkeiten auch in einem Gebiet, das viele Jahre erforscht wurde und im kommerziellen Bereich zu Produkten geführt hat, immer noch existieren, Schaden anzurichten.

Literatur

[BSI07] Bundesamt für Sicherheit in der Informationstechnik (Hrsg.): *Erkennung und Abwehr von DDoS-Angriffen im Internet - Einschätzung und Bewertung aktueller Bedrohungen*, Bonn, 2007.

Visualisierung von Malware-Verhalten

Philipp Trinius

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

Bei der dynamischen Malwareanalyse wird die Malware in einer kontrollierten Umgebung – einer Sandbox – ausgeführt und die von ihr durchgeführten Operationen werden aufgezeichnet. Die Ausgabe stellt ein detaillierter Report dar, der aus hunderten Einträgen bestehen kann. Die Länge und die Vielzahl der darin enthaltenen Details kann einen menschlichen Analysten dabei leicht überfordern. Abbildung 1 zeigt mit der *Treemap*- und der *Threadgraphen*-Darstellung zwei Visualisierung, die die Analyseergebnisse in für den Analysten schneller fassbaren Formen aufbereiten und ihn bei der Bewertung des Malwareverhaltens unterstützen können.

Die Treemap-Darstellung soll einen schnellen Überblick über die in dem Report aufgezeichneten Ereignisse und deren Frequenz geben. Auf der x-Achse sind hierzu die Sektionen (Registy, Filesystem, Network, usw.) aus denen Ereignisse aufgezeichnet wurden aufgelistet. Die einzelnen Sektionen sind zusätzlich entlang der y-Achse in die jeweils beobachteten Operationen unterteilt.

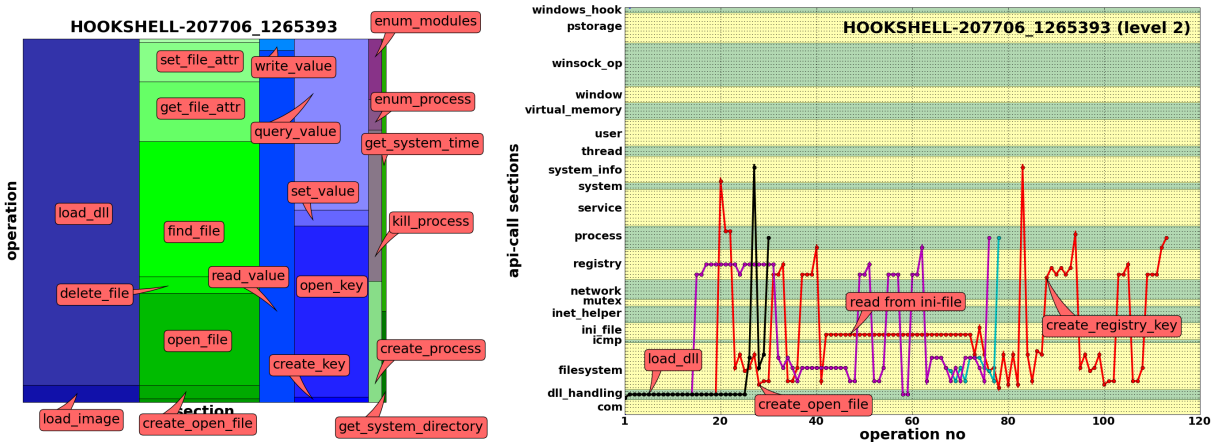


Abbildung 1: Treemap- und Threadgraph-Darstellung eines CWSandbox-Reports

Der Threadgraph liefert eine Detailsicht auf die beobachteten Threads. Entlang der x-Achse chronologisch aufgereiht werden die einzelnen von den Threads ausgeführten Operationen auf der y-Achse aufgetragen. Ein Analyst kann damit den Ablauf und den Zusammenhang zwischen den einzelnen API-Calls auf den ersten Blick erfassen. Zusätzlich lässt sich am Threadgraph ablesen, welche Operationen nicht ausgeführt werden. Ein Analyst kann somit auf den erstem Blick erkennen ob beispielsweise Netzwerkkommunikation stattgefunden hat. Auch das Wissen um nicht duchgeführte Operationen kann das weitere Analysevorgehen stark beeinflussen.

Neben der Unterstützung die die graphische Darstellung dem Analysten bietet, ermöglicht die Visualisierung mehrerer Reports, mit Bilderkennungs- und Clusteringalgorithmen, bereits existierende Verfahren auf die Malwareanalyse zu portieren. Erste Versuche mit knapp 2000 Samples aus 13 Familien lieferten hier viel versprechende Ergebnisse. Auch bei der Analyse infizierter Officedokumente mit Hilfe der Visualisierung konnten die von den Dokumenten zusätzlich angestoßenen Threads und Operationen sichtbar gemacht werden.

EDL als Ereigniskorrelationssprache in Multi-Sensor Intrusion Detection Systemen

Christoph Leuzinger

TU Dortmund
Informationssysteme und Sicherheit (ISSI)
Lehrstuhl VI, Fakultät für Informatik
christoph.leuzinger@cs.tu-dortmund.de

Die Abhängigkeit unserer Gesellschaft von Rechnernetzen und entfernten Diensten ist in den letzten Jahren stets größer geworden. Dies erfordert eine umfassende Absicherung des Betriebs von verteilten Systemen sowohl durch präventive als auch durch reaktive Maßnahmen. Voraussetzung für das Ergreifen reaktiver Sicherheitsmaßnahmen ist, dass Angriffe auf Komponenten des Systems erkannt und dem Sicherheitsverantwortlichen möglichst detailliert dargestellt werden.

Mit der zunehmenden Vernetzung von Rechensystemen beschränken deren Benutzer – und damit auch jene mit bösartigen Absichten – ihre Aktivitäten immer weniger auf ein einzelnes System, sondern nutzen verschiedene Komponenten eines verteilten Systems. Gleichzeitig eröffnen Audit-Komponenten in vielen Softwareprodukten Möglichkeiten, Beobachtungsdaten aus dem verteilten System zu Überwachungszwecken zu extrahieren. So können einerseits verschiedene Aktionen eines Benutzers im System in einem gemeinsamen Kontext beobachtet werden und andererseits einzelne Aktionen anhand von Überwachungsdaten mehrerer Komponenten unter verschiedenen Aspekten betrachtet werden. Dies erlaubt eine umfassendere Analyse von Nutzerverhalten zur Erkennung von missbräuchlichem Verhalten, als punktuell eingesetzte IDS sie leisten, denen der Kontext von Nutzeraktionen nur in eingeschränktem Maß zur Verfügung steht.

Auditdaten aus verschiedenen Überwachungskomponenten in einem verteilten System überschneiden und ergänzen sich. Die Aufgabe der Ereigniskorrelation ist es, Zusammenhänge zwischen beobachteten Aktionen sichtbar zu machen. Dazu werden die Auditdaten der Überwachungskomponenten gesammelt und einem mehrstufigen Prozess zugeführt. In dieser Arbeit werden die Stufen eines Korrelationsprozesses in Anlehnung an Vorschläge aus der Literatur [1, 2] beschrieben und untersucht, ob und wie sie sich als Ereignisbeschreibungen in der Event Description Language (EDL) [3, 4] modellieren lassen.

Literatur

[1] Christopher Krügel, Fredrik Valuer und Giovanni Vigna. *Intrusion Detection and Correlation: Challenges and Solutions*. Santa Clara, CA, USA: Springer-Verlag TELOS, 2004.

[2] Peng Ning, Sushil Jajodia und X. Sean Wang. *Intrusion Detection in Distributed Systems: An Abstraction-Based Approach*. Advances in Information Security. Kluwer, 2004.

[3] Sebastian Schmerl. *Entwurf und Implementierung einer effizienten Analyseeinheit für Intrusion-Detection-Systeme*. Diplomarbeit. Brandenburgische Technische Universität Cottbus, 2004.

[4] Michael Meier. *Intrusion Detection effektiv! Modellierung und Analyse von Angriffsmustern*. Springer Verlag, 2007.

Smartphone Malware Evolution Revisited

Aubrey-Derrick Schmidt*

*Technische Universität Berlin - DAI-Labor
10587 Berlin, Germany
aubrey.schmidt{at}dai-labor.de

This abstract gives an short summary on the paper [Schmidt09c]. The paper extends past research on smartphone malware from a perspective not having the same access to malware databases as most anti-virus product vendors have. We noticed a great discrepancy between published malwares and corresponding available descriptions on their behaviors. Especially in the last two years, descriptions on the bahaviors got scarce without obvious reason.

For statistical purpose, we gathered all published malware *descriptions*[1] from various web pages, e.g. from F-Secure, Kaspersky, McAfee, Symantec, Sophos, and similar, for identifying key aspects of mobile malware. One obvious aspect is their appearance in time. Figure 2 shows mobile malware evolution from January 2004 to December 2008 based on published mobile malware with available behavior description. We found 288 smartphone malwares until end of 2008 where peaks in new appearing malware can be found at the end of 2005 and in the middle of 2006.

For comparison, we requested the numbers from F-Secure Research in Helsinki. Comparing the numbers from Figure 2, you can see that F-Secure counted 418 malwares, 130 more than we found, showing that there are several malwares without publicly available descriptions. Additionally, following the F-Secure numbers, in the middle of 2006, more than 100 new malwares appeared. There is no obvious reason for this discrepancy but our estimation is that most of the missing malwares have only very little modifications in their code, compared to the other existing malwares.

As a conclusion on our paper [Schmidt09c], we can state that smartphone malware did not get the attention it deserved. Therefore our main intention was to update related researchers for getting improved understanding on this topic.

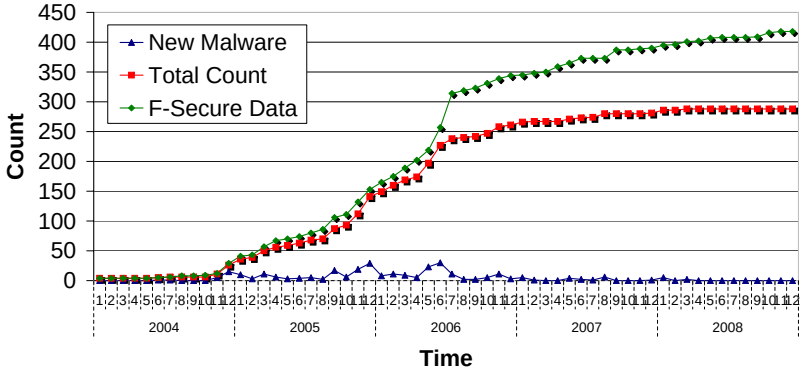


Figure 2: Mobile malware evolution basing on published malware including descriptions on their behavior. F-Secure data was added for comparison.

References

[Schmidt09c] Aubrey-Derrick Schmidt, Hans-Gunther Schmidt, Jan Hendrik Clausen, Leonid Batyuk, Seyit Ahmet Camtepe, Sahin Albayrak and Can Yildizli: Smartphone Malware Evolution Revisited: Android Next Target?, *4th International Conference on Malicious and Unwanted Software (Malware 2009)*, Montreal, Quebec, Canada - to appear

[1] malwares lacking descriptions were ignored

Android Application Sandbox

Thomas Bläsing

Technische Universität Berlin
DAI-Labor
10587 Berlin, Germany
thomas.blaesing{at}dai-labor.de

This article gives a short introduction on my diploma thesis which focuses malicious software detection on the Android platform.

The usage of smartphones in everyday life is getting more popular. While former mobile devices were mainly used only for phone calls, modern devices often have the ability to install third-party software and are connected to the internet. Along with these new possibilities there are also new attack vectors on the security of mobile devices.

One of the newest platforms for mobile devices is Android which is developed by Google and the Open Handset Alliance. The Android operating system is open source and based on a Linux kernel.

Detection of malware on this platform is a new but important aspect also for the mobile security sector. Considering [Schmidt09] shows that there is an unspecified number of malicious software already existing for Android as well as for other mobile platforms.

Although there are still certain parallels to commonly used Linux environments, Google engineer Patrick Brady said "Android is not Linux". This might indicate that some special characteristics of Android and also the limitations of the used mobile hardware makes it harder to detect and react on malware attacks if using the same techniques as on personal computer environments.

The common way of developing software for Android is to write the source code in Java and afterwards build an APK package which is afterwards uploaded to the phone and then executed by the user. Google distributes an SDK which assists the developer and also makes sure that the source code complies to the public Android API. But there are different methods, *e.g.*the Java Native Interface (JNI, see [Batyuk09]), where a developer can subvert the restrictions of an APK and therefore perform unusual, potentially malicious, actions on the mobile phone.

The aim of the diploma thesis is to detect such malicious software packages, warn the user and then maybe disarm them completely as a future work. As a first step, the aim is to develop a sandbox environment where a customer can submit an APK to get a deeper analysis of the application. Corresponding on this there are several ways of going on in the process which will be specified in the thesis itself.

References

[Schmidt09] Aubrey-Derrick Schmidt, Hans-Gunther Schmidt, Jan Hendrik Clausen, Leonid Batyuk, Seyit Ahmet Camtepe, Sahin Albayrak and Can Yildizli: Smartphone Malware Evolution Revisited: Android Next Target?, *4th International Conference on Malicious and Unwanted Software (Malware 2009)*, Montreal, Quebec, Canada - to appear

[Batyuk09] Leonid Batyuk, Aubrey-Derrick Schmidt, Hans-Gunther Schmidt, Seyit Ahmet Camtepe, Sahin Albayrak: Developing and benchmarking native linux applications on android. In mobileWireless-Middleware, Operating Systems, and Applications, 2009.

Context-Awareness in Security Solutions

Leonid Batyuk

Technische Universität Berlin
DAI-Labor
10587 Berlin, Germany
leonid.batyuk{at}dai-labor.de

In today's world, everyone is connected, anywhere and anytime. Increasing ubiquity of technology inevitably leads to a more complex IT ecosystem. While smartphones have been seldom some five years ago, their reach in today's market has risen dramatically. More and more vendors introduce new devices which allow third party software and have significant computational resources.

The advent of handheld computing devices has lead to an emerging scientific interest in new capabilities which these class of hardware provide. Hardware sensors have become commonplace in the mobile world, including geopositioning, motion, etc. A solid research field has been established around the concept of context-awareness, where the raw sensor data is being collected in order to detect the user's current context. This status information is being used to assist the user and improve the overall device experience [Baker09].

Ubiquity of mobile devices in the enterprise field brings security implications with it. In contrast to traditional wired networks with fixed workplaces, a mobile device is being used in untrusted places and insecure, potentially hostile networks.

In my PhD thesis, I am proposing a solution which would provide a significant security increase through consideration of context in access control scenarios. The focus lies not only on physical sensors, but also software sensors, *e.g.*running applications, opened data streams, etc. The main contribution will be an expandable system which would be able to recognize users' personal contexts, prevent data leaks and react to abnormal actions. The core of the framework will be comprised of a set of AI-based components for context learning and recognition, and a policy engine allowing deployment of a set of rules defining reactions upon certain security threats or potential misuse.

The system will be able to recognize a threat or a misuse taking place and *react* upon the incident in an appropriate way, *e.g.*if a user performs a series of unusual and potentially harmful actions, she is being forced to authenticate again to make sure the device has not been stolen. Another possible scenario is automatically turning on a VPN connection to the company headquarters if the device is in a potentially insecure untrusted network and access to sensitive data has been requested by the user.

The aimed platform for a proof-of-concept implementation will be Android, the open source mobile OS driven by Google, since we already have gained expertise in it [Batyuk09], and it is the only open source mobile OS currently on the market. As an appropriate test setting, the system combined with several back-end services typical for an enterprise environment.

References

[Baker09] Nigel Baker, Madiha Zafar, Boris Moltschanov and Michael Knappmeyer: Context-Aware Systems and Implications for Future Internet, *Future Internet Conference and Technical Workshops*, 2009

[Batyuk09] Leonid Batyuk, Aubrey-Derrick Schmidt, Hans-Gunther Schmidt, Seyit Ahmet Camtepe, Sahin Albayrak: Developing and Benchmarking Native Linux Applications on Android, *Mobile Wireless Middleware, Operating Systems, and Applications (Mobilware'09)*, Berlin, 2009.

Continuous User Verification through Behavior Biometrics

Arik Messerman*
* Technische Universität Berlin, DAI-Labor
Ernst-Reuter Platz 7, D-10587 Berlin, Germany
arik.messerman{at}dai-labor.de

Most computerized systems usually authenticate users by means of a user name and a password. Following a successful authentication process, the user is generally granted access rights, so the singular authentication process can be regarded as an essential prerequisite for access control.

There are different ways to authenticate humans to a system. An authentication process consists of the validation of the authorization by at least one of the following three aspects, but they can be also combined: (i) the user knows a secret (e.g. a password), (ii) the user is in the possession of a physical item (e.g. a smart card) or (iii) the user has something that unambiguously represents himself (e.g. biometric characteristics). Each of these categories has both advantages and disadvantages. Mechanisms based on the knowledge of a secret will transfer the risk to the user himself. The user is responsible for ensuring that secrecy is not compromised. With the abundance of user accounts of a typical internet user, he will often use simple user passwords, which can be guessed or spied out. While the possession of a physical item seems to be a promising alternative, it has the disadvantage that it can be stolen or lost and thus an account can be compromised. Depending on the type of procedure, the use of biometric features (e.g. fingerprints, iris scans, voice recognition or any combination thereof) can be considered an innovative and promising method. Unfortunately, most biometric techniques require the employment of rather expensive devices, so their area of application is rather limited. Independent of the type of authentication mechanism, in fact the authentication process is usually performed only once initially.

In recent years various keystroke dynamic behavior-based approaches were published, which authenticate humans based on their typing behavior. The major part focuses on so-called static text approaches, where users are requested to type a previously defined text. Relatively few techniques are based on text not previously defined, so-called *free text* approaches, which allow a transparent monitoring of user activities and with that a *continuous* verification. Unfortunately only few solutions are deployable in real application environments under real conditions, because of scalability reasons, too high response times or error rates. The aim of this work is the development of a behavioral-based verification solution, which can be deployed under real conditions in order to allow a transparent and free text continuous verification of active users with low error rates and response times. Regarding this, I intend to implement a solution, which guarantees that the verified object represents in truth the current acting human. Since the user's behavior will be analyzed during the interaction trough a standard input devices such as the keyboard, no additional hardware equipment is required.

References

[1] Moskovitch R., Feher C., Messerman A., Kirschnick N., Mustafic T., Camtepe A., Löhlein B., Heister U., Möller S., Rokach L., Elovici Y. Identity Theft, Computers and Behavioral Biometrics. In *IEEE Intelligence and Security Informatics, 2009. ISI '09.*, 2009

Using entropy-analysis for shellcode detection

Michael Gröning *† Jan Kohlrausch *

*DFN-CERT Services GmbH
Sachsenstrasse 5-7
20097 Hamburg
{groening,kohlrausch}@dfn-cert.de

† Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Technik und Informatik
Berliner Tor 7
20099 Hamburg

The operation of low-interaction honeypots creates a huge amount of data, for which efficient techniques of analysis have to be developed.Especially honeypots covering bigger IP-ranges, are in need of automatic methods that deliver good results. Medium interaction honeypots like Nepenthes [BK06] are very efficient to capture malware which is spread by known worms or attacks. However, other techniques are requiered to deal with the unknown attacks, misrouted peer-to-peer traffic, or polymorphic shellcode. In this contribution, we propose an approach for a preclassification of this traffic which supports a subsequent finer analysis.
We apply principles originated in information theory like entropy[SW63][LH07] combined with other characteristics of the data. For our project we use a low-interaction honeypot[W07] to collect netflows for which certain characteristic are calculated. Aspects of the overall characteristics include average and maximum entropy, variance and standard deviation, which were determined with the help of a sliding window algorithm.
For the following evaluation we selected datasets which exhibit characteristics implying the existence of shellcode, like a significant difference between average and local entropy of our dataset or a high variance of the calculated entropy values of the sliding window.
The evaluation of our results showed, that in direct comparison to an other method[BK05], analysis of entropy characteristics delivers comparable results in spite of reduced complexity of implementation.
The achieved results show that the entropy analysis is a promising method and can provide a basis for further analysis methods.

References

[SW63] Shannon, Claude E. ; Weaver, Warren: *A Mathematical Theory of Communication*. Champaign, IL, USA : University of Illinois Press, 1963. – ISBN 0252725484

[BK06] Baecher, Paul ; Koetter, Markus ; Dornseif, Maximillian ; Freiling, Felix: The nepenthes platform: An efficient approach to collect malware. In: *In Proceedings of the 9 th International Symposium on Recent Advances in Intrusion Detection (RAID*, Springer, 2006, S. 165–184

[LH07] Lyda, Robert and Hamrock, James: Using Entropy Analysis to Find Encrypted and Packed Malware. In: *IEEE Security and Privacy* 5 (2007), Nr. 2, S. 40–45. – ISSN 1540-7993

[W07] Werner, Tillmann: Honeytrap - Ein Meta Honeypot zur Identifikation und Analyse neuer Angriffstechniken. In: *Sicherheit in vernetzten Systemen - 14. Workshop in Hamburg am 7. und 8. Februar 2007* DFN-CERT Services GmbH (Veranst.), Christian Paulsen, 2007, S. H1–H22. – ISBN 978-3-00-020162-2 *http://honeytrap.carnivore.it*

[BK05] Baecher, Paul and Koetter, Markus: libemu - x86 shellcode detection and emulation 2008-11-30 *http://libemu.carnivore.it/*

Detecting Bots with Automatically Generated Network Signatures

Peter Wurzinger

Vienna University of Technology,
International Secure Systems Lab
pw@seclab.tuwien.ac.at

Botnets have gained popularity among Internet criminals as tools to carry out their malicious tasks, such as sending spam emails, or participating in distributed denial of service attacks. Bot-infected machines are usually under the complete control of the attacker, who can establish a communication channel (command and control, C&C) with the infected hosts, and provide them with new instructions. Bots are therefore a much more powerful instrument in the hands of an attacker than traditional malware programs.

Existing counter-measures against the botnet threat include host-based virus scanners, and network intrusion detection systems (NIDSs). Virus scanners rely on a set of signatures that each binary is matched against. This strategy fails in the long run, due to the increased use of polymorphic malware programs. Moreover, each host-based solution shares the disadvantage that it requires users to actively install additional software on their computers. NIDSs commonly run on the Internet gateway machine, and protect users without requiring end-host installations. However, the effectiveness of NIDSs strongly depends on the set of signatures that the traffic is matched against. Creating these signatures requires costly manual work by human experts. Additionally, these signatures must be constantly updated for the NIDS to correctly detect new bot variants.

In our work, a strategy for detecting bot-infected machines is presented that tackles the mentioned shortcomings of existing approaches. Our system works solely on the network level. No end-host installation is required. Also, the models that the system checks the network traffic against are produced automatically, without the need for human interaction.

The system is given a set of executables from one bot class as input. Each sample is run in a controlled virtual environment with Internet access for a limited period of time, and all network activity is recorded. The collected network traces are then searched for sudden changes in the bot's network activity, using change point detection techniques. The underlying assumption is, that these sudden changes were triggered by a command the bot had previously received from the botmaster. Consequently, by extracting the commonalities between various occurrences of an incoming bot command, it is possible to produce detection models that characterize, first, the command the bot receives via the C&C channel, and second, the response the bot shows in the network traffic while executing that command. Since bots are used for various purposes, multiple models are generated for each bot class, reflecting different types of bot activity. No assumptions are made about the nature of the tasks a bot performs, the syntax of the commands, or the protocol used to implement the C&C channel. The generated models can be deployed on standard NIDSs and they reliably detect bot-infected machines with very few false positives.

A paper describing this work is published and presented at ESORICS 2009 [1].

References

[1] Peter Wurzinger, Leyla Bilge, Thorsten Holz, Jan Goebel, Christopher Kruegel, and Engin Kirda. Automatically Generating Models for Botnet Detection, *14th European Symposium on Research in Computer Security (ESORICS)*, 2009.

Evaluation von Lernverfahren zur Bewertung der Gefahrenlage im Internet

Simon Hunke

Forschungsinstitut für Kommunikation, Informationsverarbeitung und Ergonomie (FKIE)
hunke{at}fgan.de

In gemeinsamen Aktivitäten der Universität Bonn und dem BSI sind anhand von *honeytrap*–Sensoren [Wer07] täglich mehrere tausend Angriffsversuche aus dem Internet pro Tag und Sensor aufgezeichnet worden. Vor dem Hintergrund des korrespondierenden Daten– und Informationsvolumens und der Unmöglichkeit, sämtlichen Gefahren vorbeugend zu begegnen oder diese zu identifizieren, ist eine automatisierte Bewertung der Gefahrenlage im Internet in verschiedenen Dringlichkeitsstufen ein hilfreiches Werkzeug, um kritische Situationen schneller erkennen und Gegenmaßnahmen einleiten zu können. Ein solches Kategorisierungsverfahren bildet eine potentielle Grundlage für automatisierte Internet–Alarmsysteme.

In diesem Zusammenhang existieren zwei grundlegende Ansatzmöglichkeiten: Die Konzipierung von Berechnungsvorschriften und der Einsatz von maschinellen Lernverfahren. Berechnungsvorschriften sind unter der Annahme ihrer Korrektheit zuverlässig und liefern stets eine fehlerfreie Einschätzung. Jedoch ist ihre Bestimmung insbesondere in komplexen Problemdomänen schwierig. Maschinelle Lernverfahren hingegen sind oftmals einsetzbar, wenn Experten zu einer Problemstellung intuitive Einschätzungen, aber keine explizite Herleitung angeben können. Zudem können Lernverfahren leicht auf neue Gegebenheiten angepasst werden, indem ihr Training auf einer veränderten Datengrundlage vollzogen wird. Berechnungsvorschriften müssen für diesen Zweck neu konzipiert werden.

Mit Blick auf die hohe technologische Entwicklungsgeschwindigkeit und die Komplexität der Problemstellung sind maschinelle Lernverfahren aus den genannten Gründen ein vielversprechender Ansatz zur Kategorisierung der Gefahrenlage im Internet. In verwandten Problemstellungen (IDS [RLM98] und Spam–Filter [And00]) konnten neuronale Netze beziehungsweise Bayes Netze bereits erfolgreich eingesetzt werden. Im Rahmen einer Diplomarbeit wurden diese beiden Lernverfahren anhand eines konkreten Bewertungsszenarios evaluiert.

Den durchgeführten Untersuchungen liegen drei verschieden komplexe Modelle zur Gefahrenbewertung im Internet auf Basis von Sensordaten und in Anlehnung an das menschliche Beurteilungsvermögen zugrunde. Die Modelle stammen, ebenso wie die verfügbaren Trainingsdaten, aus der gemeinsamen Aktivität der Universität Bonn und dem BSI. Der Fokus der Evaluation liegt insbesondere im Umgang der Lernverfahren mit der Komplexität des Lernproblems. Es werden jedoch auch Lernverfahren bezogene Anforderungen an die Beschaffenheit und das Volumen der Trainingsdaten abgeleitet.

Literatur

[RLM98] J. Ryan, M. Lin und R. Miikkulainen. *Intrusion Detection with Neural Networks.* Proc. of the 1997 conference on Advances in neural information processing systems 10, MIT Press, 1998.

[And00] I. Androutsopoulos. *Learning to Filter Spam E–Mail: A Comparison of a Naive Bayesian and a Memory–Based Approach.* Proc. of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD 2000), Springer, 2000.

[Wer07] T. Werner. *honeytrap — Ein Meta–Honeypot zur Identifikation und Analyse neuer Angriffe.* 14. DFN-CERT Workshop "Sicherheit in vernetzten Systemen", 2007.

Generierung von Signaturen mittels statischer Kontrollflussanalyse

René Rietz
BTU Cottbus
D-03013 Cottbus
rrietz{at}informatik.tu-cottbus.de

Die derzeit eingesetzten Intrusion Detection Systeme (IDS) realisieren häufig eine Signaturanalyse, bei der protokollierte Ereignisse mit vordefinierten Mustern (Signaturen) verglichen werden. Der Entwicklungsprozess einer entsprechenden Signatur umfasst eine Reihe von zeitaufwendigen und fehleranfälligen Teilschritten. Der Signaturmodellierer muss zunächst den neuen Angriff ausführen und die auftretenden Ereignisse protokollieren. Anschließend identifiziert er die sicherheitsrelevanten Aktionen und erzeugt schrittweise eine Signatur zur Beschreibung der ausgenutzten Sicherheitslücke. Diese Signatur wird in einer Testphase auf ihre Genauigkeit und Korrektheit überprüft. Ungenaue oder nicht korrekte Signaturen führen zu Fehlalarmen und nicht erkannten Angriffen.

Im Bereich der Host-basierten Intrusion Detection wurden verschiedene Konzepte zur Vermeidung dieses aufwendigen Prozesses untersucht. Die meisten Konzepte basieren auf der Anomalieerkennung, bei der das Programmverhalten mit einem Modell des Programmes verglichen wird. Vom Programmmodell abweichendes Verhalten wird als potenzieller Angriff gewertet. Das zu diesem Zweck benötigte Modell muss zunächst aus dem Quelltext der zu untersuchenden Anwendung (statische Analyse) oder dem laufenden Programm (dynamische Analyse) hergeleitet werden. In das Modell fließen ebenfalls alle Anwendungslogikfehler ein. Infolgedessen sind logisch fehlerhafte Programmabschnitte von korrekten Programmabschnitten nicht zu unterscheiden.

Für die Erkennung der Ausnutzung von Anwendungslogikfehlern müssen die wesentlichen Spuren eines Angriffes auf den betroffenen Programmabschnitt beschrieben werden. Die Modellierung einer Signatur zur eindeutigen Identifizierung dieses Abschnittes ist besonders zeitaufwendig und fehleranfällig. Zur Minimierung des Verwundbarkeitsfensters eines Programmes ist jedoch die zeitnahe Bereitstellung einer Signatur (vor der Veröffentlichung der Fehlerbehebung) notwendig. Für die Beschleunigung der Signaturmodellierung sollen die einzelnen Teilschritte automatisiert werden. Im Gegensatz zur Anomalieerkennung ist eine vollständige Automatisierung des Signaturgenerierungsprozesses nicht möglich. Infolgedessen wird ein Ansatz vorgestellt, der eine Teilautomatisierung dieses Prozesses realisiert.

Voraussetzung für den Generierungsprozess ist die Spezifizierung der Programmstellen, an denen ein Angriff erfolgreich ausgenutzt werden kann (Punkt der Verwundbarkeitsausnutzung). Diese Programmstellen dienen als Ansatzpunkt für die Generierung einer minimalen Signatur zur Beschreibung verwundbarer Kontrollflüsse im Programm. Der Signaturgenerierungsprozess erzeugt aus dem Quelltext der verwundbaren Anwendung ein vollständiges Modell des Programmverhaltens (Kontrollflussgraph), das häufig eine hohe Komplexität aufweist. Dieses Modell wird im weiteren Verlauf des Generierungsprozesses auf die Beobachtungsmöglichkeiten eines IDS (bspw. Betriebssystemaufrufe und Bibliotheksaufrufe) reduziert. Anschließend wird ein Ausschnitt des Modells ermittelt, der den von der Sicherheitslücke betroffenen Programmabschnitt eindeutig identifiziert. Die generierte Signatur dient dem Signaturmodellierer als Ausgangsbasis für die weitere Beschreibung der konkreten Sicherheitslücke.

Modeling Fraud Scenarios in a RETE-based Stateful Rule Engine

Cristina Fortu and Ulrich Flegel

SAP Research Karlsruhe
D-76131 Karlsruhe, Germany
cristina.fortu{at}sap.com, ulrich.flegel{at}sap.com

The auditing of high volumes of business transactions even today involves painstaking manual labor. There is a strong need for real-time tools selecting only the suspicious activity for further scrutiny. The goal of this bachelor thesis is constructing such a tool based on an existing general rule engine and validating its real-world applicability based on real fraud scenarios and real business transaction data.

Accurately specifying scenario patterns is a key ingredient for a practically relevant solution. Thus an appropriately expressive specification language is required. For this project a RETE-based stateful rule engine is used for modeling fraud scenarios, employing the specification language *WANF*. The current status of this project includes the documentation of the expressiveness of the WANF language with respect to the semantics model provided by Meier [1] for modeling frauds and provides constructive proof of concept by offering working WANF constructions for each of the supported semantic concepts. The results show that the WANF language supports most of the semantic aspects described by Meier [1], with the exception of the following: simultaneity from event pattern, and the modes "last" and "all" from step instance selection. This allowed us to translate the reference fraud scenarios into WANF and giving detailed descriptions of the red flag alerts, similar to the Intrusion Detection Message Exchange Format.

One of the frauds encountered in a company, where an employer intends to make purchases which are much higher than the imposed limit, without asking for supplementary approval is the "order splitting" example. This is how the WANF rule for detecting the fraud would look like:

```
   rule orderSplitting
if exists SRM:PurchaseOrder po ( exists SRM:PurchaseRequisition pr
and po.poNumber==pr.poNumber and po.amount>pr.limit)
enable rf = new SRM:RedFlag(‘‘rf11’’, ‘‘Purchase Order Splitting’’, ‘‘Intention of
making purchases for amounts higher than approved, without management approval’’);
```

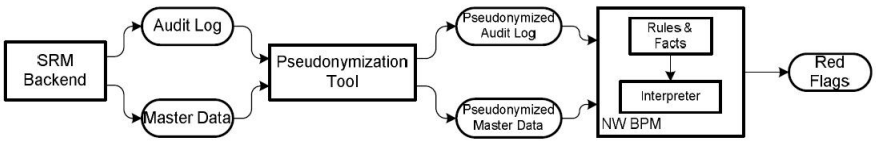


Figure 3: System Architecture

The diagram depicts the architecture of the system where the two main components are the SAP Supply Relationship Management System and the SAP NetWeaver BPM. The future work includes the implementation of I/O adapters for testing the rules using real data, and the pseudonymization of sensitive data for compliance with the Federal Data Protection Act.

References

[1] Michael Meier. *A Model for the Semantic of Attack Signatures in Misuse Detection Systems.* In *Proceedings of the 7th International Information Security Conference (ISC 2004)*, LNCS 3225, pages 158 - 169, Palo Alto, CA, USA, September, 2004, Springer.

Automatic Generation of Separation of Duty Fraud Scenrios

Stanislaus Stelle and Ulrich Flegel

SAP Research Karlsruhe
D-76131 Karlsruhe , Germany
stanislaus.stelle{at}sap.com, ulrich.flegel{at}sap.com

According to research conducted by the American Association of Certified Fraud Examiners (ACFE), American companies lose on average 5% of their revenues to fraud [1]. Most fraud cases are committed by regular employees to make a profit for themselves, by misusing work privileges granted to them. To address this issue, it is important to separate duties by implementing the four-eye principle. This means having a process or workflow which causes money or goods to change hands it is important to have at least two employees who are involved in this process. In the SUPER project [2] was a composer developed which orchestrates web services according to their pre- and postconditions. This composer is a backend algorithm which can determine a correct sequence of adequately annotated web services to achieve a desired workflow. It is possible to use this composer to generate critical paths which are sequences of actions which lead to goods or money changing hands. The discovered critical paths can be used in different ways. Once there is a sequence of actions which would empower a single user to run a workflow or parts of it which would cause money or goods to change hands, the sequence can be transformed into rules which can be interpreted by a fraud detection system - such as described by Fortu and Flegel [3] - and used to supply a repository of critical paths. The critical paths can afterwards be dynamically used to analyse web service orchestrations on the fly and raise red flags if needed.

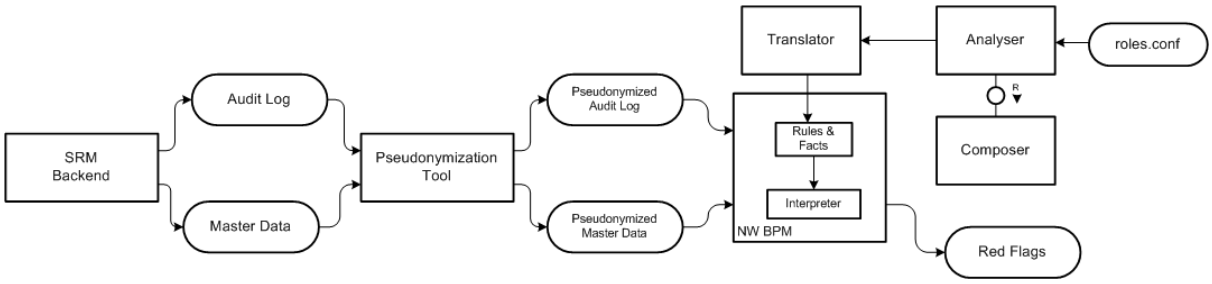


Figure 4: Architecture

References

[1] Association of Certified Fraud Examiners : *Report to the Nations*, 2006, `http://www.acfe.com/documents/2006-rttn.pdf`

[2] Maximo Casas: *Semantic Utilized for Process management within and between Enterprises*, in *Project IST 026850 SUPER*, November 2008. `http://www.ip-super.org/res/Deliverables/M30/D6.9.pdf`.

[3] Cristina Fortu and Ulrich Flegel : *Modeling Fraud Scenarios in a RETE-Based Stateful Rule Engine with First-Order Capabilities*, 2009

Design and Application of a Security Analysis Method for Healthcare Telematics in Germany (HatSec)

Ali Sunyaev

Technische Universität München
D-85748 Garching, Germany
sunyaev{at}in.tum.de

Purpose: The goal of this work is to provide a method for organisational and technical analysis of security issues in health care (using tools, methods and processes in a structured and traceable way). On the basis of this method the current security status of health care telematics in Germany is evaluated and valuable hints for future developments in the health care sector are derived.

Findings: During the planning stage of designing such an IS security analysis method specific to healthcare industry [Ka04], it is advisable to base the design procedure on established standards and best practice approaches, so that the security analysis method relies on previously approved frameworks [SBH04]. Based on the PDCA (Plan/Do/Check/Act) model [De86] the HealthcAre Telematics SECurity - HatSec - analysis method is built in a compositional manner. This means that the HatSec method was designed from existing IS security analysis approaches (like ISO 27001 and IT-Grundschutzhandbuch), which were subdivided into method fragments. These method fragments were used to construct the HatSec security analysis method. The identified method fragments of selected IS security analysis approaches were methodically composed into the following seven steps: (1) scope identification, (2) asset identification, (3) basic security check, (4) threat identification, (5) vulnerability identification, (6) security assessment and (7) security measures. These steps represent at least one part of an IS security approach that fits best to the current situation.
The application of the HatSec method identified 24 deficiencies around the current status of the German health care telematics (including weaknesses, inconsistent and conflicting development documents and violation of security demands) and provided solutions for discovered vulnerabilities accordingly.

Practical implications: Based on the outcome of this research project, a broader understanding of analyzing healthcare security is expected. The created method is designed for chief information security officers (CISO) to analyze forthcoming or already implemented healthcare information systems. A further contribution to practice is the identification of security problems in the current concept of the German healthcare telematics.

References

[Ka04] Katsikas, S.K.: *Health care management and information systems security: awareness, training or education?* International Journal of Medical Informatics, Vol. 60 (2000), pp. 129-135.

[SBH04] Siponen, M., Baskerville, R. and Heikka, J.: *A Design Theory for Secure Information Systems Design Methods.* Journal of the Association for Information Systems, Vol. 7 (2006) Nr. 11, pp. 725-770.

[De86] Deming, W. E.: *Out of the Crisis.* MIT Center for Advanced Engineering Study, Mit Press (1986).

Kann Data Leakage Prevention vor Datenoffenlegung schützen?

Matthias Luft
Universität Mannheim
D-68159 Mannheim
mluft{at}informatik.uni-mannheim.de

Data Leakage Prevention ist der allgemeine Ausdruck für ein neues Konzept, das die unautorisierte Offenlegung von Daten verhindern soll. Da immer wieder Fälle weitgreifender Offenlegung auftreten [DBR09], werden seit einigen Jahren Implementierungen veröffentlicht [MQ09], die verschiedene inhalts- und kontextbasierte Untersuchungen auf abgefangenen Daten durchführen.

Diese Analysen basieren auf Policies, die schützenswerte Daten beschreiben. Es existieren verschiedene Möglichkeiten, diese Policies zu definieren und Daten abzufangen um so die Analyse erst zu ermöglichen.

Diese Arbeit untersucht exemplarische Lösungen, um enthaltene Sicherheitslücken aufzudecken. Diese Sicherheitslücken können verschiedene Auswirkungen nach sich ziehen: So kann beispielsweise die Offenlegung von Daten nicht wirksam verhindert werden oder es können sogar neue Möglichkeiten für das Auftreten von Datenverlust hinzukommen [MM07]. Dieser Prüfungsprozess ist ein essentieller Schritt im Lebenszyklus jeder neuen Software oder jedes neuen Konzepts. Bevor eine Lösung als zuverlässig betrachtet werden kann, sollte ein kontinuierlicher Zyklus an Testphasen und Untersuchungen existieren und durchlaufen werden. Für das relativ neue Gebiet *Data Leakage Prevention* muss zunächst eine Reihe von Anforderungen definiert werden, die durch entsprechende Tests überprüft werden können.

Die entwickelte Testreihe orientierte sich dabei hauptsächlich an den folgenden Fragen:

- Ist zufällige Offenlegung immer noch möglich?
- Ist es möglich, die installierten Schutzmechanismen zu umgehen?
- Sind Schwachstellen in der Software enthalten?

Werkzeuge für die Beantwortung der Fragestellung sind unter anderem Verschlüsselung, Verschleierung, Metadaten sowie forensische Untersuchungsmethoden.

Das Ergebnis der durchgeführten Tests erlaubt allgemeine Rückschlüsse auf den Reifegrad aktueller Produkte und zeigt prinzipielle Probleme des Konzepts *Data Leakage Prevention* auf. Die initiale Fragestellung kann daher zumindest für die untersuchten Lösungen beantwortet werden: Unautorisierte Offenlegung kann nicht verhindert werden. Dabei kann leider keine Unterscheidung zwischen bösartig motivierten Zugriffen und fahrlässigem Umgang mit Daten gemacht werden.

Literatur

[DBR09] Wade H. Baker, David Hylender, Andrew Valentine: *2009 Data Breach Investigations Report, Verizon Business RISK Team.*

[MQ09] E. Quellet und P. Proctor: *Magic Quadrant for Content Aware Data Loss Prevention, Gartner RAS Core Research, 2009.*

[MM07] E. Monti und D. Moniz: *Defeating Information Leak Prevention, Matasano Labs, 2007.*