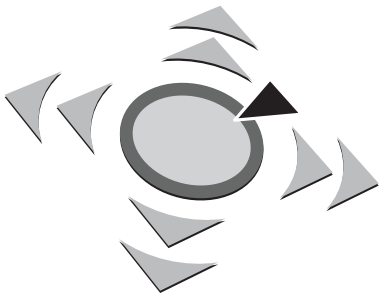
8. GI FG SIDAR Graduierten-Workshop über Reaktive Sicherheit

SPRING

Christoph Pohl, Sebastian Schinzel und Steffen Wendzel (Hrsg.)

18.-19. Februar 2013, München



SIDAR-Report SR-2013-01
ISSN 2190-846X

Diesen Bericht zitieren als:
Christoph Pohl, Sebastian Schinzel und Steffen Wendzel, editors. Proceedings of the Eight GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2013-01, GI FG SIDAR, München, Februar 2013,

Beiträge zitieren als:
Autor. Titel. In Christoph Pohl, Sebastian Schinzel und Steffen Wendzel, editors, Proceedings of the Eight GI SIG SIDAR Graduate Workshop on Reactive Security (SPRING). Technical Report SR-2013-01, page xx. GI FG SIDAR, München, Februar 2013.

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 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 (HFT Stuttgart) Ansprechpartner innerhalb der Fachgruppe SIDAR.

Nach der Premiere in Berlin fand SPRING in Dortmund, Mannheim, Stuttgart, Bonn, Bochum und 2012 erneut in Berlin statt. Wir freuten uns in diesem Jahr darüber, die Veranstaltung erstmals in München organisieren zu können, bei der wir 14 Einreichungen erhielten und in diesen technischen Bericht aufnahmen. 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 hohe Anzahl an Einreichungen für die diesjährige SPRING ermöglichte uns Sessions mit den Themenbereichen Network Security, Intrusion Detection, Data Hiding, Kryptologie sowie Malware anzubieten.

Die zugehörigen Abstracts sind in diesem technischen Bericht zusammengefasst und wurden über Hochschulbibliothek Augsburg elektronisch, zitierfähig und recherchierbar veröffentlicht. Der Bericht ist ebenfalls über das Internet-Portal der Fachgruppe SIDAR zugänglich (`http://www.gi-fg-sidar.de/`).

Besonderer Dank gebührt Ulrich Flegel und Michael Meier für ihre Unterstützung bei der Planung. Weiterhin möchten wir uns bei Collin Mulliner und Patrick Stewin für die Bereitstellung ihrer Skripte und Unterlagen des vorherigen SPRING-Events bedanken. Zudem danken wir der Hochschulbibliothek Augsburg für die Aufnahme des Reports und den freiwilligen Studenten der Hochschule München für ihre Hilfe bei der lokalen Organisation und Durchführung des Workshops. Ganz besonders danken wir der Gesellschaft für Informatik, dem Dekan der Fakultät Informatik an der Hochschule München sowie den Firmen MGM und Opitz Personalberatung für das Sponsoring dieses Workshops.

München, Februar 2013 — Christoph Pohl, Sebastian Schinzel und Steffen Wendzel

Contents

Visibility of Routing Anomalies for End Users

Matthias Wübbeling

Fraunhofer FKIE
D-53113 Bonn
matthias.wuebbeling{at}fkie.fraunhofer.de

In todays society, the Internet, as an abstract term is equated with common web applications, shopping, advertisement and interpersonally as well as business communication. The Internet as the physical internetwork connection of independent networks, so called Autonomous Systems (AS), has been pushed into the background. Current research activities focus mainly on threats to the abstract Internet, namely malware, espionage, hacktivism or fraud. The abstraction of the Internet, away from the technical to a logical view of an always up and running entity, caused unsolved problems to be faded out of the perception of science.

The Border Gateway Protocol (BGP) is used for Inter-AS routing since IP has been chosen as interconnection protocol. The simplicity of BGP has made its contribution to the growth of the Internet. However, this simplicity is a threat for worldwide secure communication over the Internet. Routing stability and BGP security has been topic in science scince the late 90's, as demonstrated in [1] and [3]. BGP alternatives and security enhancements, such as [1] and [2] could not make it to the highend routing hardware at the end. Although communication is not possible without BGP, it remains hidden from most of the users. As a matter of fact, Inter-AS routing anomalies occur regularly and are only visible for admins of Autonomous Systems - who care only, if their prefixes are affected directly. They can be informed using projects like PHAS [4] or BGPmon [5] but no such service exists for ordinary users. Incidents in the last months and years show, that it is rather easy to reroute traffic over suspicous autonomous systems.

In conjunction with security flaws of SSL certificate authorities it is not sufficient to only encrypt traffic anymore. Users must be aware of routing information in general and anomalies in particular to rely on secure Internet communication. Our goal is to show this information directly into the user's browser with a browser plugin. To achieve this, routing anomalies should be detected, based on public routing information gathered from sites like RouteViews [6] or RIPE RIS [7]. In addition to this, traceroute information directly from the users systems will be collected and used to improve existing algorithms for detecting routing anomalies. These algorithms will be based on IP prefix observation and topological changes. On a large server system, global anomalies as well as geographically local routing issues should be detected. Each routing announcement will be examined and classified by our software and reported as possibly insecure, if anomalies are detected.

Users will be informed by our browser plugin, if they browse a site of an affected IP prefix and decide on their own, whether to use a possibly insecure routed connection or suppress communication until it is secure again. Finally, our system addresses still unsolved problems concerning the detection and handling of routing anomalies and the visibility of security problems of Internet connections, that are hardly recognizable for end users.

References

[1] S. Kent, C. Lynn, and K. Seo: *Secure Border Gateway Protocol (S-BGP)* in *IEEE Journal on Selected Areas in Communications* Vol. 18, No. 4, 2000.

[2] van Oorschot, P. C., Wan, T. und Kranakis, E.: *On Interdomain Routing Security and Pretty Secure BGP (psBGP)* in *ACM Transactions on Information and System Security* Vol. 10, No. 3, 2007.

[3] Labovitz, C., Malan, G. R. und Jahanian, F.: *Internet routing instability* in *IEEE/ACM Trans. Netw.* Vol. 6, No. 5, 1998.

[4] Lad, M., Massey, D., Pei, D., Wu, Y., Zhang, B. und Zhang, L.: *PHAS: a prefix hijack alert system* in *Proceedings of the 15th conference on USENIX Security Symposium - Volume 15*, USENIX Association, 2006.

[5] http://www.bgpmon.net/

[6] http://www.routeviews.org/

[7] http://www.ris.ripe.net/

The *Missing Data Problem* in Cyber Security Research

Sebastian Abt

da/sec – Biometrics and Internet Security Research Group
Hochschule Darmstadt, Darmstadt, Germany
sebastian.abt@h-da.de

Cyber security is becoming an increasingly important field of research in the area of IT security. In contrast to other disciplines in the area of IT security, this field of research is lacking one essential and ubiquitously required resource: *data*.

Unfortunately, access to real-world data is usually restricted due to legal, regulatory or political reasons or only granted after signing non-disclosure agreements – rendering it effectively useless for rigour scientific work. Yet, unrestricted access to data is required by the scientific community for essentially two complementary reasons: First, cyber security is a very time critical field of research. Attackers' tools are constantly changing in order to be able to exploit newly identified vulnerabilities and to adapt to changing environments. To be able to identify such changes in tools and targets and to develop countermeasures, researchers at scale need access to data reflecting it. Second, one of the fundamental requirements of scientific work is reproducibility. Other researchers need to be able to reproduce experiments in order to validate or invalidate one's claims or to be able to compare competing approaches. In order to be able to reproduce experiments and compare results, however, a common data set to experiment on is required.

One approach to address the lack of real-world data is to synthetically generate traces according to an underlying model. Synthetically generating data has several advantages over real-world data. First, traces can be generated on demand and at arbitrary sizes within a deterministic period of time. Especially the latter is not guaranteed when capturing data in real-world. Second, traces can be generated tailored to the problem at hand to solve. When captured in the wild, traces do not necessarily have to show all required properties. Third, synthetically generated data can be shared within the community without restrictions as it does not interfere with any law or regulatory requirements. On the other hand, synthetically generated data, by definition, do not necessarily have to reflect all aspects of real-world data. As further disadvantage, synthetically generated data may introduce systematic errors, *i.e.* artefacts, that may especially affect its utility when it comes to training of machine learning approaches.

In this talk, availability of and issues with real-world data for cyber security research is reviewed. We especially review data sets used in recent accepted conference submissions according to their general availability to the community. As one result, we show that the vast majority of data sets recently used is not publicly accessible. We furthermore sketch approaches to data synthesis and discuss challenges arising. We especially discuss the levels of detail and abstraction of data synthesis and its impact on utility of data.

Re-Authentication Model for Mobile Devices

Matthias Trojahn*, Frank Ortmeier†

*Volkswagen AG
Wolfsburg, Germany
matthias.trojahnl{at}volkswagen.de

†Otto-von-Guericke University
Magdeburg, Germany
frank.ortmeier{at}ovgu.de

Security is a growing field while working with information. Especially in the mobility area where devices can be transported everywhere. The growing usability has its risks if the device is stolen or lost. For example, in London over 55.000 mobile devices get lost in taxis in half a year. Different steps have to be done to secure the system and the information. Hardening the system, anti-virus software and authentication system are required to secure a traditional computer. Here, we focus on the authentication part while we address a biometric keystroke authentication. For this, we presented new features in previous publications [1, 2] on the capacitive display. This display of todays mobile devices allows us to record pressure and size of the finger tip during pressing the device. In addition, the exact coordinates for the event can be localized instead of only recognizing which key was pressed. But this is only one aspect of authentication. In addition to the often used initial authentication at the beginning of the usage of a device, a re-authentication can be done during usage. If the device is unlocked, no further security measure exist which recognize who the user is at different moments. That is why text-independent keystroke authentication was introduced by different researcher [3, 4] for the computer keyboard and the 12-key mobile keyboard layout. In both cases hardware keys were used. The new features which were addressed by us earlier can be used also in this use cases.

For this, we adapted our keyboard layout (on Android OS) to record the new data from the capacitive display as well as sensor information of the interial sensors (accelerometer and gyroscope). The movements of the device during typing can be used, in addition, to authenticate a person.

We present in this publication the framework for a re-authentication containing the special aspects regarding the biometric authentication steps. In addition, we show first experimental results which we obtained.

References

[1] M. Trojahn and F. Ortmeier, "Biometric authentication through a virtual keyboard for smartphones," in *International Journal of Computer Science & Information Technology (IJCSIT)*, 2012.

[2] ——, "Toward mobile authentication with keystroke dynamics onÂ mobile phone," in *The 7th International Symposium on Security and Multimodality in Pervasive Environment (SMPE-2013)*, 2013.

[3] F. Monrose and A. D. Rubin, "Keystroke dynamics as a biometric for authentication," in *Future Generation Computer Systems*, vol. 16. Elsevier Science Publishers B. V, 2000, pp. 351–359.

[4] S. Zahid, M. Shahzad, S. A. Khayam, and M. Farooq, "Keystroke-based user identification on smart phones," in *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection*, ser. RAID '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 224–243.

Data Inconsistency Detection in Industrial Control Networks

Moritz Obermeier*, Maryna Krotofil*

* Technische Universität Hamburg Harburg
D-21079 Hamburg, Germany
{moritz.obermeier,maryna.krotofi}{at}tuhh.de

Industrial control systems (ICS) enable automation and control of complex chemical and mechanical processes. Sensors' readings (process values) and control commands for actuators are transported by control networks in real time ($\geq$ 10 ms). The primary goal of control and automation (C&A) systems is to ensure the desired quality of the process under control in a safe manner and to reduce productivity downtime to zero. Without awareness of cyber threats, the design requirements for ICS were performance, reliability, and safety. Hence, C&A systems were designed in a way to make the life of system engineers and process operators easy and efficient. This resulted in simplistic implementation of even basic security services, e.g. hardcoded passwords. Data integrity protection was not considered at all.

Programmable logic controllers (PLC) perform the primary control of the process equipment. Operators monitor the state of the process round-the-clock and take control over automated control whenever necessary. If the measured process values in the field do not match the values used in control logic of a PLC and/or values displayed on the operator's screen, the process is not under control any longer. This is the most unsafe and undesired situation possible. This type of attack was realized in Stuxnet [1].

Due to the limited number of publicly available industrial cyber accidents, the attacker model and attack scenarios for ICS are not well known. Depending on the attacker's objective, she can e.g. cause DoS or tamper with the process and/or control data to cause harm to process or equipment while hiding this information from the operator and/or control devices. Tampering can be executed through exploitation of individual components [2] or by tampering with data in transit.

Our goal is to provide operators, control hardware and supporting applications with the reliable situational awareness. We achieve this in two ways. We monitor process values consistency across the network and data storages by comparing their numeric meaning "on the wire" and in the DBs. Control data integrity cannot be verified in the same way. It is validated indirectly by monitoring the state of the process during the next execution of the control loop cycle. Our IDS platform contains a mathematical model of the physical process under control, which is used to precompute process values. The actual measurements data captured from the network are then verified against the precomputed values. Upon the deviation from the safe confidence interval an alarm will be generated.

References

[1] N. Falliere and L.O. Murchu and E. Chien *W32. Stuxnet Dossier v1.4.* In White paper, Symantec Corp., Security Responsen, February 2011

[2] T. Newman and T. Rad and J. Strauchs *SCADA & PLC Vulnerabilities in Correctional Facilities White Paper.* 2011

protectedNET: Self-Protection auf Netz-Ebene

Fabian Berner

Hochschule Furtwangen
D-78120 Furtwangen
fabian.berner@hs-furtwangen.de

Die Infrastruktur von Computernetzen hat sich in den letzten Jahren stark verändert: Ausgehend von fest miteinander verbundenen Computern hin zu lose gekoppelten, mobilen Systemen mit vielen und ständig wechselnden Endgeräten. Insbesondere durch die Verbreitung drahtloser Datenübertragungstechniken wurde diese Entwicklung beschleunigt. In drahtlosen Netzen ist der Zugang zum gemeinsamen Übertragungsmedium schwer zu kontrollieren. Dadurch sind auch die teilnehmenden Endgeräte des Netzes schwer zu überwachen.
Durch den Trend, private Geräte auch für geschäftliche Zwecke einzusetzen (Bring Your Own Device, BYOD), ist eine zentrale Administration der am Netz teilnehmenden Systeme nicht mehr gewährleistet. Das bedeutet, dass jedes Endgerät im Empfangsbereich eines Funknetzes - bewusst oder unbewusst - als potenzielles Angriffsmedium genutzt werden kann. In einem solchen Netz kommt es zu einer Vielzahl von Ereignissen die zu Gefahrensituationen bzw. die zu einer geänderten Einschätzung der Sicherheitssituation führen können. Auf eine automatische Analyse der Gefahrensituation folgt in der Regel eine manuelle Reaktion [1]. Die topologische Dynamik des Netzes erfordert aber eine möglichst weitgehende Automatisierung der Reaktionen. Um die automatische Reaktion auf Gefahrensituationen entscheidend zu verbessern, wurde das Konzept eines geschützten Netzes (protectedNET) entwickelt. Das Netz besitzt ein Sicherheitssystem, um sich und die Teilnehmer zu schützen.
Da es sich bei den Endgeräten oft um mobile Systeme mit beschränkten Ressourcen handelt, müssen große Teile des Sicherheitssystems in das gemeinsam genutzte Netzinfrastruktur integriert werden. Der Unterschied zu traditionellen Netzen ist, dass die Gewährleistung der Sicherheit von den jeweiligen Endgeräten an die Netzinfrastruktur selbst übergeben wird.
Durch eine wechselseitige Vertrauensbasis zwischen den mobilen Endgeräten und der Netzinfrastruktur können auch Informationen der Endgeräte in die Sicherheitsanalyse integriert werden. Durch den Mehrwert der Endgerätedaten können zuverlässigere Entscheidungen getroffen werden und somit die *False-positive*-Rate im Vergleich zu bisherigen Ansätzen verringert werden. Mit der Verteilung des Sicherheitssystems über alle Endgeräte im protectedNET können neben *Outsider-Attacks* auch *Insider-Attacks* erkannt werden. Endgeräte müssen bei der ersten Verbindung zum protectedNET auf ihre Vertrauenswürdigkeit hin untersucht werden. Die Verbindung zu anderen Netzen findet über ein Sicherheitsgateway statt.
Das Sicherheitsgateway und die Endgeräte erzeugen bei bestimmten Ereignissen Events, die von einem zentralen Eventprozessor [2] verarbeitet werden. Sobald eine Sicherheitslücke erkannt wurde, veranlasst der Eventprozessor entsprechende Reaktionen beim Sicherheitsgateway.

Literatur

[1] SHAMELI-SENDI, A., et. al. 2012. Intrusion Response Systems: Survey and Taxonomy. In IJCSNS International Journal of Computer Science and Network Security, S. H. LEE, Ed., 1-14.

[2] ETZION, O., NIBLETT, P. 2011. Event Processing in action. Manning, Stamford.

Botnet Detection Using General-purpose Security Sensors

Till Elsner

Fraunhofer FKIE, Bonn, Germany

The growing spread of malware in general and especially remote controllable malware, the foundation of botnets, makes botnets a more and more serious threat. Botnets massively endanger both the owners of malware-infected systems and persons or organizations becoming a target of criminals running a botnet: Users who become infected by botnet-running malware suffer theft of personal and sensitive data or abuse of their computer resources and digital identities in attacks. Victims of botnets face an adversary with massive resources of thousands of computers to carry out attacks, able to easily overburden the attacked systems and deployed security infrastructure.

An effective defense against a botnet requires its detection in the first place. Once uncovered, defensive measures and mitigation strategies can be established. In research, several approaches have been developed to detect formerly unknown botnets , such as [1], [2] and [3]. These method exploit characteristics of a botnet that distinguishes its behaviour from other security-related events in networks. The application of such botnet-specific methods requires the deployment of botnet-aware sensor system.

Because of its heavily distributed nature, monitoring a botnet from a single viewpoint might lead to an incomplete picture. Investigations may profit from a distributed network of sensors, which provides information from different viewpoints and even from the inside of non-public networks. The use of botnet-specific detection approaches makes such an setup practically difficult, since the application of botnet-aware sensors makes a deployment of those sensors in all participating sites necessary. On the other hand, organizations running a network of a certain size have their own security infrastructures deployed to protect against common and well-known security issues.

A botnet is a complex network construct, consisting of computer systems that unwillingly contribute to the botnet because of a malware infection, commanded by a bot master to act on his behalf and launching attacks like denial-of-service, email spam distribution, or data theft. Considering botnets as being only being the sum of its parts and taking its pieces apart, however, shows a botnet to consist of rather classical components, well-known to computer security research and handled by widely deployed general-purpose security sensor systems. Intrusion detection systems are able to discover the infection attempts used to acquire new bots, anti-virus software detects the malware that provides remote control of a bot, and attacks carried out by the botnet are noticed by the detection system designed for the particular kind of attack, like firewalls or spam filters. That means that the activities even of unknown botnets are already being monitored, the sensors doing so just don't interpret the data in the context of a botnet.

Our approach to utilize general-purpose security infrastructure for the detection and monitoring of botnets is a framework that accumulates data of security incidents from sensor systems deployed by different organizations. These sensors do not have to be aware of the methods of operation or even the existence of botnets and do not need to change any operational parameters. The accumulated security events are correlated by time of occurrence and behaviour patterns observable in order to connect single events to the activity of a botnet. Since we make no demands on the participating sensor systems, a broad base of data covering a wide range of security aspects is available, that enables us to follow a botnet at every turn. Botnet-specific knowledge is necessary at a central point of evaluation only. This makes way for the effortless installation of an effective botnet detection platform using infrastructure that is already in place.

References

[1] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, Wenke Lee: *BotHunter: Detecting Malware Infection Through IDS-Driven Dialog Correlation*, Proceedings of the 16th USENIX Security Symposium, 2007

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

[3] Guofei Gu, Roberto Perdisci, Junjie Zhang, Wenke Lee: *BotMiner: Clustering Analysis of Network Traffic for Protocol- and Structure-Independent Botnet Detection*, Proceedings of The 17th USENIX Security Symposium, 2008

Anomaliebasierte Intrusion Detection mit CUSUM und Generischem Normalmodell

Christoph Pohl

Hochschule München, Munich, Germany

Intrusion Detection Systeme (IDS) sind bedingt durch die mediale Betrachtung, aber auch durch die existierende Bedrohung wie die aktuellen Angriffe auf REWE, Paypal oder das LKA zeigen, ein elementarer Bestandteil in der Absicherung von Netzwerken. Ein Kernpunkt bei der Entwicklung von IDS ist dabei die Erkennungsqualität und Performance [1]. In der Angriffserkennung werden vorwiegend Algorithmen zur Anomalierkennung verwendet. Definiert wird eine Anomalie dabei durch eine Abweichung vom Normverhalten der beteiligten Ressourcen [2]. Als grundsätzliches Problem der Anomalieerkennung kann dabei die Ressourcenauslastung und die Parallelisierbarkeit der Erkennungsalgorithmen beschrieben werden [3]. Gamer begründet diesen Engpass mit hohen Datenvolumina bei begrenzten Verarbeitungsressourcen. Die Herausforderung liegt also darin einen schnellen Algorithmus zu entwickeln welcher parallelisierbar ist und eine hohe Erkennungsgeschwindigkeit bietet. Dieser Artikel beschreibt einen linear skalierenden Algorithmus zur Erkennung von Anomalien in hohen Datenvolumina. Der Algorithmus basiert auf einem modifizierten CUSUM-algorithmus [4] mit einem generischen Normalmodell. Dieses ermöglicht eine Anomalieerkennung mit einer umgebungsabhängigen Steigungsgrenze.

Literatur

[1] P. Kabiri and A.A. Ghorbani, "Research on intrusion detection and response: A survey," *International Journal of Network Security*, vol. 1, no. 2, pp. 84-102, 2005.

[2] R.R. Kompella, S. Singh, and G. Varghese, "On scalable attack detection in the network," in *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*. ACM, pp. 187-200, 2004.

[3] T. Gamer, "Dezentrale, Anomalie-basierte Erkennung verteilter Angriffe im Internet," KIT Scientific Publishing, 2010.

[4] E. Page, "Continuous inspection schemes," *Biometrika 41,* 1/2, 100-115, 1954.

JavaScript Obfuscation Detection: A Static Approach

Malte Göbel

Ruhr-Universität Bochum
D-44801 Bochum
malte.goebel{at}rub.de

JavaScript is a powerful programming language often used to create dynamic webpages. Web-based frameworks like jQuery, MooTools and Dojo use this scripting language as well. Some programmers obfuscate code to prevent plagiarism or to simply hide the sense of a code. Unfortunately, there are also use cases that do not serve a good cause. Attacks on web browsers are commonly based on obfuscated JavaScript, too. Therefore, signature-based anti-virus scanners often fail to detect malicious code [1]. Anti-virus scanners sometimes make use of dynamic analysis techniques to address this problem in a web-based context. The downside of this approach, in comparison to static analysis, is the processing overhead and therefore the time needed to complete the analysis.

A fast decision, whether a webpage is obfuscated or not, is needed and implies the call for a system that can detect obfuscated JavaScript statically. Commonly used methods are obfuscation techniques based on long strings [2] on the one hand and number-based obfuscation like CharCode or hybrid methods on the other hand. A system that concentrates on string analysis and on the source code itself, especially on common usage of numbers, should be the first choice.

To develop such a system, strings were extracted in a first step to calculate features like several entropies and character distribution. By comparing obfuscated and unobfuscated strings, a different relation between the entropy of the string itself and entropies calculated after removing character groups like numbers or special characters is conspicuous. Furthermore, replacing strings, integers and binary operators from the sourcecode using tokens, we simplified the code. Analysing this tokenized code in terms of long concatenations and numbers commonly occurring, an identification of several obfuscation techniques is efficient. Afterwards, a different frequency of assignments concerning obfuscation and unobfuscated scripts was noticed. A scoring system was set up based on these presented features. As result of the analysis, an obfuscation score indicates whether the website is obfuscated, in case the score exeeds a previously definded value, or whether it is not obfuscated.

To complete this presented analysis an interpretation of the sourcecode is not necessary and therefore this analysis is faster than a dynamic analysis. For evaluation purposes, a validation set of 7,000 obfuscated scripts and of 2,000 manually verified unobfuscated scripts was taken. The system presented in this paper efficiently detects JavaScript obfuscation techniques in the validation set. Furthermore, first tests on real world data resulted in a nominal false positive rate, without any identified false negatives. In summary, it can be stated that a fast decision whether a webpage needs to be analysed dynamically or not is implementable and reduces the processing overhead.

References

[1] W Xu, F Zhang, S Zhu The Power of Obfuscation Techniques in Malicious JavaScript Code: A Measurement Study `http://www.cse.psu.edu/~szhu/papers/malware.pdf`

[2] Y Choi, T Kim, S Choi Automatic Detection for JavaScript Obfuscation Attacks in Web Pages through String Pattern Analysis `http://www.sersc.org/journals/IJSIA/vol4_no2_2010/2.pdf`

Resource-Efficient Side Channel Mitigation in Embedded Environments

Johannes Bauer*

*University of Erlangen
D-91054 Erlangen, Germany
johannes.bauer{at}cs.fau.de

Embedded systems that rely on general-purpose microcontrollers are present in a large amount of todays electronic devices such as automotive control units, wireless routers, RFID readers or even dishwashers. With the rising connectivity of these devices, the need for secure communication has simultaneously arisen.

Since strong, well-applied cryptographic algorithms and techniques are readily available nowadays — for instance in the form of TLS, ECC and AES — it would appear as if this problem is easily solvable. The difficulty that emerges with embedded systems in particular is twofold: Firstly, the embedded appliance is usually in the hands of the customer. From a security perspective this means that the device is in permanent captivity of a potential attacker with unlimited time on her hands. The second problem is the resource limitation of embedded devices. Many attack mitigation procedures [BGNS06] are resource-intensive and are therefor not computationally feasible for use on an embedded microcontroller.

While the cryptographic foundation in use by such a system may well be considered secure, an attacker may find that the concrete implementation is vulnerable by applying techniques like differential power analysis [KJJ99] or fault injection [HTI97]. An example would be an attacker who tries to extract private cryptographic keys from a RFID door opener in order to be able to clone vendor-supplied RFID tags with higher privileges — essentially electronic lock picking.

At the same time microcontroller units — due to their general nature — have a large amount of peripherals that are idle in most use cases. For instance, it is unlikely that a microcontroller within a RFID reader will ever use the analog-digital converter, all available timers or a possibly present FPU. Our approach is to utilize the peripherals which are already present but inactive and (ab)use them in a manner that makes side channel analysis of the system more challenging.

We differentiate between active and passive techniques. Active mechanisms, for example, are the deliberate generation of noise to mask power consumption. This can be used to make differential power analysis more difficult. Passive techniques use peripherals of the device in order to detect suspicious activity. This would include power line glitches or main clock irregularities, which are all indicative of a possibly ongoing attack. The goal is to utilize resources that are already present — and therefor cost-neutral — on a microcontroller in order to improve security of the overall system by hampering side channel attacks on the device.

References

[BGNS06] Ernie Brickell, Gary Graunke, Michael Neve, and Jean-Pierre Seifert. Software mitigations to hedge aes against cache-based software side channel vulnerabilities, 2006.

[HTI97] Mei-Chen Hsueh, Timothy K. Tsai, and Ravishankar K. Iyer. Fault injection techniques and tools. *Computer*, 30(4):75–82, April 1997.

[KJJ99] Paul Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. pages 388–397. Springer-Verlag, 1999.

Similarity Preserving Hashing

Frank Breitinger

da/sec - Biometrics and Internet Security Research Group
Hochschule Darmstadt, Darmstadt, Germany

frank.breitinger AT cased.de

During a forensic investigation, an investigator is required to analyze the content of personal devices, e.g., hard disks, DVDs, sdcards. Due to huge amounts of data this resembles a needle in the haystack. Thus investigators need proceedings to enlarge the needle or minimize the haystack. One possibility is to identify files automatically and categorize them into good files (e.g., operating system) or suspect files (e.g., company secrets, child pornography).

A trivial solution to solve this problem is to have all known files in a set or database. To analyze a device we compare all inputs against the database. However, comparing inputs directly has three major problems:

1. Disk space problem as all files need to be available.
2. Comparing inputs directly is often very time consuming. For instance, considering TXT files and the Levenshtein distance, we have a quadratic complexity ($O(n^2)$).
3. There is only a little possibility to order the set (e.g., file types, size). Thus we approximately have a $n : m$ comparison to proceed a device having m files and a database having n entries.

In order to solve these problems computer forensics uses hash functions to make a fingerprint-based-comparison which has mainly three benefits, a) fixed length, very short fingerprints (e.g., 256 Bit), b) fast identification of exact duplicates and c) fingerprints could be put in hashtables which results in a lookup complexity of $O(1)$. However, this solution only allow *yes-or-no decisions* and no similarity detection, e.g., different versions of a file.

Typical working fields for similarity preserving hashing (SPH) besides digital forensics are spam/malware detection, biometrics and network traffic analysis whereby the use-cases are slightly different. For instance, in digital forensics it comes along with (suspect) file identification, in combination with malware/spam it may be usable for classification, in biometrics it is a scheme for template protection and in network traffic for packet inspection on flow data.

Comparing network packets against files is more like *fragment detection*. If SPH algorithms support fragment detection, this reveals new use-cases. Fragment detection could be used to uncover partly deleted/corrupted files, identify a virus in a 'normal' executable or find embedded objects, e.g., a JPG within a word document.

Currently we are aware of 6 different approaches all with different strength and weaknesses: `ssdeep` (2006, Kornblum), `sdhash` (2010, Roussev) and four own approaches `bbHash`, `saHash`, `mvHash` and `MRSH-v2`. The general problem is that there is *no definition* and therefore it is nearly not possible to compare/test them.

In order to overcome this drawback we are working on a) a definition, b) a test framework and c) a process model. Concerning the definition it starts right in the beginning - the naming. Next we would like to raise reasonable properties based on the well establish cryptographic hash functions properties. The test framework should be able to test existing and upcoming approaches and print a report. Within the process model we first like to discuss the limits on identifying similar files (all mentioned approaches work on the byte level) and find a reasonable proceeding how to combine binary and semantic similarity hashing.

Sicherheit des Pseudozufallszahlengenerators LAMED

Gabriele Spenger

FernUniversität Hagen
D-58097 Hagen
gabriele@spenger.org

RFID (Radio-Frequenz-IDentifikation) erlaubt die digitale automatische Erfassung von Daten. Die Datenübertragung erfolgt über eine hochfrequente Funkverbindung. Dies geschieht kontaktlos, nahezu orientierungsunabhängig und ohne Sichtverbindung zum Datenträger. Dass die Übertragung über die Luftschnittstelle stattfindet, birgt die Gefahr, dass die Daten ausgespäht werden können. Abhängig vom Anwendungsszenario muss die Datenübertragung daher mittels kryptographischer Mechanismen abgesichert werden.

Die Sicherheit kryptographischer Authentifizierungsprotokolle beruht unter anderem auf der Sicherheit der verwendeten Pseudozufallszahlengeneratoren. Da die Pseudozufallszahlen auch auf dem Transponder erzeugt werden, sind die entsprechenden Generatoren jedoch besonderen Anforderungen hinsichtlich der Komplexität unterworfen.

Ein Pseudozufallszahlengenerator, der speziell für diese Anwendung entwickelt wurde, ist LAMED, der 2009 von den Autoren Pedro Peris-Lopez et al. [1] veröffentlicht wurde.

Die Sicherheit von Pseudozufallszahlengeneratoren wird üblicherweise durch den Einsatz von standardisierten Testbatterien nachgewiesen (z. B. [2]). Hierbei wird eine begrenzte Zahl von erzeugten Pseudozufallszahlen statistisch ausgewertet.

Eine weitere Analyse lässt sich mit Hilfe von induzierten Graphen durchführen. Betrachtet man die Zahlenfolge eines Pseudozufallszahlengenerators als eine Folge von Zuständen, erhält man einen Graphen $G = (V, E)$ mit V als Menge aller möglichen Zustände und E als Menge der gerichteten Kanten der Zustandsübergänge ([3]). Im Allgemeinen ist es nicht möglich, den vollständigen Graphen zu erzeugen, da die Anzahl der Zustände eines kryptographisch einsetzbaren Pseudozufallszahlengenerators zu groß ist (bei LAMED z.B. bei Erzeugung von 16 Bit Pseudozufallszahlen 2^{81} Zustände).

Am Beispiel des LAMED wird gezeigt, wie eine solche Analyse des Zustandsbaums vorgenommen wird und welche möglichen Konsequenzen die Eigenschaften des Zustandsgraphen auf die praktische Einsetzbarkeit eines Pseudozufallszahlengenerators hat.

Literatur

[1] Pedro Peris-Lopez and Julio César Hernández Castro and Juan M. Estévez-Tapiador and Arturo Ribagorda, *LAMED - A PRNG for EPC Class-1 Generation-2 RFID specification*, Computer Standards & Interfaces, Volume 31, p. 88-97, 2009.

[2] George Marsaglia, *The Marsaglia Random Number CDROM including the Diehard Battery of Tests of Randomness*, http://www.stat.fsu.edu/pub/diehard/, Department of Statistics and Supercomputer Computations Research Institute, Florida State University, 1995.

[3] Jörg Keller, *Parallel Exploration of the Structure of Random Functions*, Proc. PASA 2002, Karlsruhe, April 2002, pp. 233-235, VDE Verlag, 2002.

Behaviour-Based Malware Clustering

Stefan Hausotte
Ruhr-University Bochum
D-44801 Bochum, Germany
stefan.hausotte{at}rub.de

Classification of novel malware is one of the main challenges in applied IT-Security [OECD, cf.]. Thousands of new vermin are detected every day, too many for manual analysis. In this talk we present a fully automated dynamic malware analysis, based on malware behaviour. The system we introduce is capable of the analysis of thousands of new malware samples every day. The behaviour of the malware is represented by API-traces, which are extracted from the malware samples via API-hooking. We used a customized version of the Cuckoo-Sandbox engine and in comparison a specialized hooking engine, which was solely developed for the purpose of behaviour based clustering. The API-traces are converted into a binary format for easier parsing and weighting of the API parameters.

We took 20 different malware families and created a mixed sample set with around 2600 samples from them. We extracted the API-traces from the malware samples and used a nearest-prototype clustering to search for similar behaviour in the samples. Nearest-prototype clustering [RT11, cf.] is a technique where a prototype for every malware family is searched in a sample set and the samples are assigned to this prototypes. For this, the malware behaviour is mapped into a n-dimensional vector space and the distances between points, which represent malware samples, are computed. The clustering algorithm searches for scatter plots in the vector space and tries to assign labels to them.

To check if our clustering worked correctly and the malware samples are sorted by their family we used internal and external evaluation methods to compute how good our results are. We used the triple of precision, recall and f-measure for the external evaluation and the Davies-Bouldin index for the internal evaluation. The results of the cluster evaluation show that results with more than 90% correct classification are accomplished.

We used differed feature-vectors to represent multiple aspects of malware behaviour and compared their results with each other. The evaluation shows that the feature-vector has the main impact on the clustering result, but also on speed and memory usage. We show the different results for the feature-vectors and a motivation why we developed a new hooking engine to overcome some drawbacks of the Cuckoo-Sandbox engine.

References

[RT11] Konrad Rieck and Philipp Trinius and Carsten Willems and Thorsten Holz (Hrsg.): *Automatic Analysis of Malware Behavior using Machine Learning* 2011

[OECD] OECD (Hrsg.): *Malicious Software (Malware): A Security Threat to the Internet Economy: Ministerial Background Report* 2008

Malware Clustering Based On Generic API Hooking

Andre Waldhoff

Ruhr-University Bochum
D-44801 Bochum, Germany
andre.waldhoff{at}rub.de

The generic processing of Windows functions and their arguments within a hooking engine allows complete logging of calls from both, the *Win32 API* and the *Native API*. Unfortunately the currently available solutions for dynamic malware analysis are not able to create their traces in such detail as they are only hooking a small subset of all possible functions. The motivation to develop a generic engine is the improved accuracy of the resulting logfiles. This gain is expected to improve the results of clustering algorithms deployed in malware classification as well as the comprehensibility for human analysts.

An important aspect when hooks are applied in malware analysis is how easy it is to detect them. Often hooks are set at the beginning of the program's execution, for example in *Export hooking* by modifying the export table. This however has several disadvantages: One is that these techniques can be detected relatively easy by comparing the file on disk and in memory when the hooks are set [BLS11]. Furthermore this technique requires to set the hooks every time a sample is executed.

In order to avoid these shortcomings we decided to manipulate the DLLs before they are loaded into memory. Instead of altering the file in memory after it was loaded by Windows, we modify the DLL on disk by enlarging each executable section to the double of its size. In the first half of the enlarged section we place our hooks at the original exported addresses, in the second one we place the original code. In this way we can alter the DLLs and replace the original files during the initial setup of the analyzing machine.

To avoid noise within the API traces we need to detect *call chains* which occur when a function which is called by the analyzed program calls other API functions. As these *subcalls* do not represent the behaviour of the sample itself, we only want to log those functions which are called directly by the executable. An example for such a callchain is the *DeleteUrlCacheEntry* function which clears the cached browser data. Therefore it needs to call native API functions which, for example, read and write to registry entries and the filesystem to access the URL cache database. The noise which is created can have significant impact on the clustering results and on the readability of the logfiles. Using a marker in unused space allocated for the *Thread Environment Block* (TEB) we are able to detect call chains and exclude them from the logfiles.

In the presentation we introduce the important aspects of the design and implementation of our concept of a generic hooking engine which is suitable for use by classification of malware using clustering algorithms, as well as for the assistance of manual analysts.

References

[BLS11] Buescher, Armin and Leder, Felix and Siebert, Thomas: Banksafe Information Stealer Detection Inside the Web Browser, Lecture Notes in Computer Science 262-280, 2011

Praktischer Einsatz von Hardware-Honeypots im Deutschen Forschungsnetz

Stefan Metzger, Wolfgang Hommel

Leibniz-Rechenzentrum
D-85748 Garching b. München
{metzger,hommel}@lrz.de

Betreiber großer Netzinfrastrukturen stehen heutzutage häufig vor der Herausforderung, ihren Kunden IT-Dienste überall und jederzeit nutzbar bereitzustellen. Eine weltweite Öffnung sensibler Netzbereiche gilt es andererseits zum Schutz dort gespeicherter, sensitiver Daten weitestgehend einzuschränken. Gezielte Freischaltungen auf Firewalls, Intrusion Detektion Systeme, die Signatur-basiert bekannte Angriffe detektieren sowie eine strukturierte, prozessorientierte Reaktionsmöglichkeit, um Auswirkungen des Angriffs schnellstmöglich einzudämmen, bilden einen gesamtheitlichen, integrierten Ansatz einer Security-Monitoring-Strategie. Wichtig dabei ist, die Detektionsmöglichkeiten auf einem aktuellen Stand zu halten und adäquat an die Bedrohungssituation anzupassen.
Aus diesem Grund betreibt das Leibniz-Rechenzentrum (LRZ) in Kooperation mit dem Center for Advanced Security Research Darmstadt (CASED) einen dort entwickelten Hardware-Honeypot, der im Gegensatz zu Software-basierten Lösungen nicht das Risiko birgt, als Opfer eines erfolgreichen Angriffs kompromittiert zu werden.
Die Malware Collection Box (MalCoBox) [MBRK10] basiert auf einer Multi-FPGA-Architektur. Ein FPGA-basierter NetStage-Kern übernimmt die Netzwerkpaketverwaltung und unterstützt gängige Protokolle wie ARP, IP und TCP/UDP. Die restlichen FPGAs werden als Vulnerability Emulation Handler (VEH) Module zur Simulation bestimmter Dienste oder deren Schwachstellen verwendet. Aufgrund beschränkter Ressourcen und des Implementierungsaufwands lassen sich darauf jedoch nicht beliebige Dienste bereitstellen. Zu berücksichtigende Kriterien, die in die Bewertung eines Dienstes, der als VEH implementiert werden soll, einfließen, sind dessen Funktionsumfang, die Anzahl und Varianz der Kommunikationspartner sowie jegliche Art von Sicherungsmechanismen.
Aktuell sind drei VEHs im Einsatz: HTTP, SMTP und MySQL. Der MySQL-VEH beispielsweise emuliert die Schwachstelle CVE-2012-2122. Diese ermöglicht Angreifern, trotz Eingabe falscher Login-Daten erfolgreich authentifiziert zu werden. Zusätzlich kann der Angreifer eine ebenfalls programmierte Schwachstelle für das weit verbreitete MySQL UDF Dynamic Library Exploit ausnutzen, um Datenbanken und Tabellen zu erstellen, beliebige eigene Daten dort einzuspeisen und eigenen Code auf dem vermeintlichen MySQL-Server auszuführen.
Am Leibniz-Rechenzentrum werden derzeit rund 250 IPv4-Adressen auf die VEHs der MalCoBox umgeleitet. Die Protokollierung der Ereignisse erlaubt, anhand der Quell-IP-Adressen und Korrelation mit anderen Alarmen anderer Security-Monitoring-Mechanismen die Erstellung eines Gesamtbildes und eine lückenlose Nachvollziehbarkeit selbst neuer Varianten des Angreifervorgehens.
Im Vortrag werden die MalCoBox, die hardwarenahe Implementierung simulierter Schwachstellen, die Auswertung bislang gesammelter Angriffsdaten und die Einbettung in unsere Intrusion Detection Infrastruktur vorgestellt.

Literatur

[MBRK10] Sascha Mühlbach, Martin Brunner, Christopher Roblee und Andreas Koch: *MalCoBox: Designing a 10 Gb/s Malware Collection Honeypot using reconfigurable Technology* in *Proceedings of the 2010 International Conference on Field Programmable Logic and Applications*, Washington DC, USA, 2010. IEEE Computer Society.