ICIMP 2012

The Seventh International Conference on Internet Monitoring and Protection

ISBN: 978-1-61208-201-1

May 27- June 1, 2012

Stuttgart, Germany

ICIMP 2012 Editors

Arno Wagner, Consecom AG, Switzerland

Petre Dini, Concordia University, Canada / China Space Agency Center, China

ICIMP 2012

Forward

The Seventh International Conference on Internet Monitoring and Protection (ICIMP 2012) held on May 27 - June 1, 2012 - Stuttgart, Germany, continued a series of special events targeting security, performance, vulnerabilities in Internet, as well as disaster prevention and recovery. Dedicated events focused on measurement, monitoring and lessons learnt in protecting the user.

Internet and Web-based technologies led to new frameworks, languages, mechanisms and protocols for Web applications design and development. Interaction between web-based applications and classical applications requires special interfaces and exposes various performance parameters.

The design, implementation and deployment of large distributed systems are subject to conflicting or missing requirements leading to visible and/or hidden vulnerabilities. Vulnerability specification patterns and vulnerability assessment tools are used for discovering, predicting and/or bypassing known vulnerabilities.

Vulnerability self-assessment software tools have been developed to capture and report critical vulnerabilities. Some of vulnerabilities are fixed via patches, other are simply reported, while others are self-fixed by the system itself. Despite the advances in the last years, protocol vulnerabilities, domain-specific vulnerabilities and detection of critical vulnerabilities rely on the art and experience of the operators; sometimes this is fruit of hazard discovery and difficult to be reproduced and repaired.

We take this opportunity to thank all the members of the ICIMP 2012 Technical Program Committee as well as the numerous reviewers. The creation of such high-quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to the ICIMP 2012. We truly believe that, thanks to all these efforts, the final conference program consists of top quality contributions.

This event could also not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the ICIMP 2012 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that ICIMP 2012 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in Internet Monitoring and Protection

We are convinced that the participants found the event useful and communications very open. The beautiful city of Stuttgart surely provided a pleasant environment during the conference and we hope you had a chance to visit the surroundings.

ICIMP 2012 Chairs
Go Hasegawa, Osaka University, Japan
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Jaime Lloret Mauri, Polytechnic University of Valencia, Spain
Richard Chow, PARC, USA
Constantion Paleologu, University 'Politehnica' Bucharest, Romania
Michael Grottke, University of Erlangen-Nuremberg, Germany

ICIMP 2012

Committee

ICMP Advisory Committee

Go Hasegawa, Osaka University, Japan
Sathiamoorthy Manoharan, University of Auckland, New Zealand
Jaime Lloret Mauri, Polytechnic University of Valencia, Spain
Richard Chow, PARC, USA
Constantion Paleologu, University 'Politehnica' Bucharest, Romania
Michael Grottke, University of Erlangen-Nuremberg, Germany

ICIMP 2012 Technical Program Committee

Jemal Abawajy, Deakin University - Victoria, Australia
Ejaz Ahmed, Queensland University of Technology (QUT)- Brisbane, Australia
Manos Antonakakis, Damballa Inc., USA
Javier Barria, Imperial College London, UK
Lasse Berntzen, Vestfold University College Norway
Jonathan Blackledge, Dublin Institute of Technology, Ireland
Matthias R. Brust, University of Central Florida, USA
Christian Callegari, University of Pisa, Italy
Eduardo Cerqueira, Federal university of Para, Brazil
Richard Chow, PARC, USA
Denis Collange, Orange-tfgroup, France
Christopher Costanzo, U.S. Department of Commerce, USA
Jianguo Ding, University of Luxembourg, Luxembourg
Matthew Dunlop, Virginia Polytechnic Institute and State University,USA
Mohamed Eltoweissy, Pacific Northwest National Laboratory, USA
William Enck, Penn State University, USA
Nicolas Fischbach, COLT Telecom, Germany
Ulrich Flegel, SAP Research - Karlsruhe, Germany
Alex Galis, University College London, UK
João Gomes, University of Beira Interior, Portugal
Stefanos Gritzalis, University of the Aegean - Karlovassi/Samos, Greece
Michael Grottke, University of Erlangen-Nuremberg, Germany
Go Hasegawa, Osaka University, Japan
Terje Jensen, Telenor Corporate Development - Fornebu / Norwegian University of Science and Technology - Trondheim, Norway
Naser Ezzati Jivan, Polytechnique Montreal University, Canada
Andrew Kalafut, Grand Valley State University, USA
Florian Kammueller, Middlesex University - London, UK
Ayad Ali Keshlaf, Newcastle University, UK
Daniel Kumar, University of North Carolina - Chapel Hill, USA

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Design and Implementation of an Active Warden Addressing Protocol Switching Covert Channels

Steffen Wendzel[1,2], Jörg Keller[1]
[1]*Department of Mathematics and Computer Science, University of Hagen, Germany*
[2]*Department of Computer Science, Augsburg University of Applied Sciences, Germany*
steffen.wendzel1@hs-augsburg.de, joerg.keller@fernuni-hagen.de

***Abstract*—Network covert channels enable a policy-breaking network communication (e.g., within botnets). Within the last years, new covert channel techniques occurred which are based on the capability of protocol switching. There are currently no means available to counter these new techniques. In this paper we present the first approach to effectively limit the bandwidth of such covert channels by introducing a new active warden. We present a calculation method for the bandwidth of these channels in case the active warden is used. Additionally, we discuss implementation details and we evaluate the practical usefulness of our technique.**

***Keywords*-Protocol Switching Covert Channel; Protocol Channel, Active Warden**

I. INTRODUCTION

The term *covert channel* refers to a type of communication channel defined as *not intended for information transfer* by Lampson [1]. While covert channels can occur on local systems, we focus on covert channels within networks. The goal of using covert channels is to transfer information without rising attention while breaking a security policy [2].

Covert channels have been a focus of research for decades. A number of publications (e.g., [3], [4], [5]) describe how to implement covert channels in network packet data and packet timings. Techniques were developed to deal with the problem of covert channels, like the pump [6], which is a device that limits the number of acknowledgement messages from a higher to a lower security level and thus affects covert timing channels based on ACKs. Other well-known techniques are for instance covert flow trees [7], the shared resource matrix (SRM) methodology [8] the extended SRM [9], and steganalysis of covert channels in VoIP traffic [10].

A well-known technology to counter covert channels is the active warden, i.e. a system counteracting a covert channel communication. While passive wardens monitor and report events (e.g., for intrusion detection), active wardens (e.g., traffic normalizers [11]) are capable of modifying network traffic [12] to prevent steganographic information transfer.

Recently, the capability to keep a low profile resulted in a rising importance of network covert channels because of their use cases. For instance, covert channels can be used to control botnets in a hidden way [13]. Covert channel techniques can also be used by journalists to transfer illicit information, i.e., they can generally contribute to the free expression of opinions [14].

A covert channel able to switch a network protocol based on a user's command called *LOKI2* was presented in 1997 [15]. Within the last decade, different new covert channel techniques occurred and not all of them are already addressed by protection means. However, these new techniques enable covert channels to switch their communication protocol automatically and transparently, as well as they were enabled to cooperate in overlay networks by using internal control protocols as presented in [16].

In this paper, we focus on two new covert channel techniques, *protocol switching covert storage channels* (also known as *protocol hopping covert channels*, PHCC) as well as so called *protocol channels* (PCs). PHCC were presented in [17] and were improved in [16]. These channels transfer hidden information using different network protocols to raise as little attention as possible due to peculiar protocol behaviour. For instance, a simple PHCC could use the "User-Agent" field in HTTP as well as the message number in POP3 "RETR" requests to transfer hidden information.

PCs were introduced in [18] and signal information solely by transferring network protocols of a pre-defined set in an order that represents hidden information. Protocols are therefore linked to secret values, e.g., a HTTP packet could represent the value "1" and a POP3 packet could represent the value "0". To transfer the message "110" via this PC, the sender is required to send two HTTP packets followed by a POP3 packet. The bandwidth of a PC is usually limited to a few hundred bit/s, however, for an attacker this is fast enough to transfer passwords, selected records or tweets.

A first detection algorithm for PC (but not for PHCC) was implemented in [19], but there is no work done to limit the bandwidth of PC and PHCC or to prevent them.

We present the first concept as well as an implementation of an active warden able to significantly reduce the bandwidth of both, PHCC and PC. While we do not present a universal solution countering all covert channels, this is the first work discussing means against PHCC and PC. The limitation of these advanced covert channels is more challenging in comparison to single-protocol covert

 ISBN: 978-1-61208-201-1

channels. We evaluate our results by simulation experiments. We demonstrate that our approach is useful for the practical operation in organizations and that the active warden decreases the attractiveness of both channel types for attackers.

The remainder of this paper is structured as follows. Section II introduces the idea and implementation of our active warden, while Section III discusses results and Section IV focuses on the practical aspects of the presented approach. Section V concludes.

II. Design and Implementation

The idea of an active warden addressing PHCC and PC focuses on one aspect both covert channel types share: the protocol switches. For both channel types, it is a necessity to ensure that network packets using different network protocols reach their destination in the same order as they were sent. Our approach of an active warden for countering PHCC and PC monitors the protocol switching behaviour of network hosts and introduces delays in network packets if a protocol switch occurs.

Like the network pump, we have no explicit detection capabilities in our active warden but aim on limiting the channel's bandwidth nevertheless. In comparison to the pump, we do not limit ACK messages but alter protocol occurrences. Using the presented technique, we can prevent performance decreases for downloads and uploads and minimize practical implications (cf. Section IV) between (autonomous) systems. The system only affects those network packets which change the last occurring network protocol.

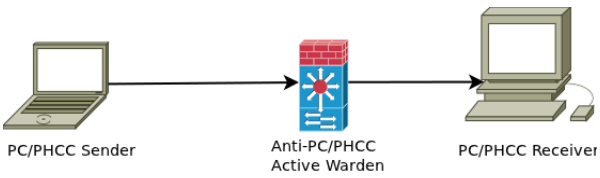


Figure 1. Location of the Anti-PC/PHCC active warden

The active warden must be located between a sender and a receiver (Figure 1). To prevent PC/PHCC-based data leakage for enterprises, a suitable location would be the company's uplink to the internet. The active warden's delay is configurable and thus makes our approach adjustable, i.e. an administrator can choose the individual optimum between maximized protection and maximized throughput. Formally, this creates a multi-criterion optimization problem. For a given delay d, data leakage can occur at a maximum rate $R(d)$, that is decreasing with increasing d. On the other hand, the side-effects for legitimate users will increase with increasing d, i.e. can be modeled by a function $S(d)$. Ideally, one would like to minimize both, R and S, which is however not possible. One can combine the two in a target function

$$\text{penalty}(d) = \epsilon \cdot R(d) + (1 - \epsilon) \cdot S(d) \ ,$$

which is then to be minimized, i.e. after fixing a suitable $\epsilon \in [0; 1]$ the minimization results in an optimal d which represents the administrator's priorities. As both R and S are assumed to be monotonous, a Pareto optimum can be found in the sense that a further reduction of R by increasing d cannot be achieved without increasing S. Typically, instead of using R and S directly, they are normalized to a certain range such as interval $[0; 1]$, and they might be adapted by linear or non-linear functions that reflect e.g. the severeness of an increased leakage.

Imagine a PC using ICMP (1 bit) and UDP (0 bit) and the goal to transfer the message "0110001". In this case, the sender would need to send UDP, ICMP, ICMP, UDP, UDP, UDP, ICMP. If our active warden is located on a gateway between both hosts and can delay packets which probably belong to a PC or PHCC, the successful information transfer will be corrupted. At the beginning, the sender sends a UDP packet which is forwarded by the active warden. Afterwards, the sender sends the ICMP packet, which is delayed for a time d because a protocol switch happened. The next packet is an ICMP packet again and therefore not delayed but forwarded. Afterwards a UDP packet occurs which is delayed for a time d, too. The next two UDP packets do not change the last protocol and are therefore forwarded. The last ICMP packet results in an additional packet switch and is therefore delayed for time d again. If d is 1 second, then all delayed packets will arrive after the non-delayed packets if the sender did not introduce synthetic delays itself. The resulting packet order at the receiver's side will be UDP, ICMP, UDP, UDP, ICMP, UDP, ICMP, or "01**00**101" (containing two incorrect bit values).

The situation is similar for PHCC where the hidden information is not represented through the protocol itself but through alternations of a protocol's attributes (such as the IPv4 TTL or the HTTP "User-Agent"). If the active warden modifies PHCC transmissions sent via different protocols, the reassembled payload will be jumbled.

In order to prevent the covert channel users to take advantage of learning about the value of d, it might also be randomized, cf. Section III.

A. Bandwidth Calculation

Tsai and Gligor introduced the formula $B = b \cdot (T_R + T_S + 2 \cdot T_{CS})^{-1}$ for calculating the bandwidth of covert channels using the values T_R (time to receive a covert message), T_S (time to send a message), T_{CS} (time required for the context switch between processes), and b (amount of transferred information per message) [20]. While the formula addresses local covert storage channels, all parameters are adaptable to a network covert channel as well.

We use a modification of this formula (cf. formula 1) to calculate the bandwidth of a PC in case our active warden is located between sender and receiver. We introduce the active warden's delay d multiplied with the probability p of

 ISBN: 978-1-61208-201-1

a protocol switch per packet. Instead of T_R, T_S and T_{CS} we use T to represent the transmission time (i.e., the time difference between receiving and sending a packet including the processing time required by the active warden).

The amount of transferred data per packet (b) is 1 bit/packet in case two protocols are used for a PC since $b = \log_2 n$, where n is the number of protocols used. Thus, p as well as n will rise if more than two protocols are used (more information can be transferred per packet but the switching rate will also increase). In case of a PHCC, b depends on the amount of storage data per packet and not on the amount of protocols used. Therefore, p will rise if more protocols are used but since the amount of protocols is not linked to the amount of information transferred per packet, b will *not* rise if more protocols are used.

$$B = b \cdot (p \cdot d + T)^{-1} \quad (1)$$

Theoretically, p is 0.5 if randomized input, a uniform coding and a set of two protocols is used since the next packet is either using the same protocol as the last (no protocol switch) or the other protocol (a protocol switch is taking place). In our experiments, the average protocol switching value for a typical protocol switching covert channel using only two protocols was $p = 0.4738806$. Thus, to transfer information without risking a corruption through a delay, a PC/PHCC user is forced to send packets with protocol switches in a way that the delay d cannot corrupt the packet order.

As mentioned earlier, the value p depends on the amount of protocols used as well as on the channel's coding. If a uniform coding was used (as with optimized channels) and if two protocols are used p is approx. 0.5. In case four protocols are used, p is approx. 0.75. In general, for n protocols used p is $(1 - 1/n)$. Thus, formula 1 can be modified to the following version:

$$B = b \cdot ((1 - 1/n) \cdot d + T)^{-1} \quad (2)$$

Protocol Channel: As also already discussed, $B = \log_2 n \cdot ((1 - 1/n) \cdot d + T)^{-1}$ in case of a PC. Taking d as well T into account using formula 1, we calculated the maximum useful bandwidths for an uncorrupted PC transfer using a set of two protocols (Figure 2). According to our calculations, a PC's bandwidth can be reduced to less than 1 bit/s if the active warden introduces a delay of 2.0 sec for protocol switches using realistic values for T.

Protocol Hopping Covert Channel: For a PHCC, the parameter b varies more than parameter T (T is, as in the case of a PC, usually very low). Therefore, we created a different plot where we set T to the static value 0.005 which we measured in experiments. Figure 3 shows the bandwidth of a PHCC dependent on the delay d as well as the amount of transferred bits per packet b. Obviously, the result of the PHCC is equal to the result of a PC if $b = 1$. However, PHCCs can carry more information and are therefore harder to limit, i.e., they require higher delays.

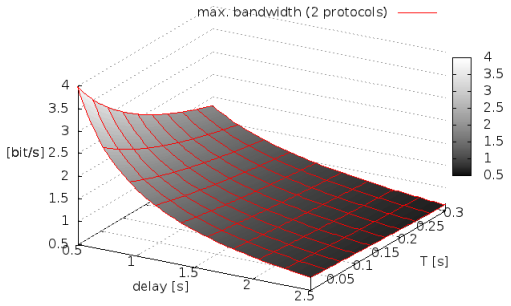


Figure 2. A PC's bandwidth (B) using a set of two protocols dependent on the delay and the transfer value.

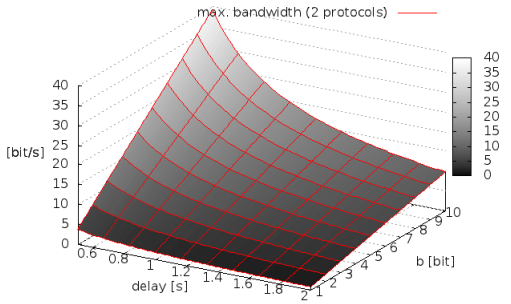


Figure 3. A PHCC's bandwidth using *two* protocols, $T = 0.005$ and delays between 0.5 and 2s as well as the capability to transfer between 1 and 10 bits per packet.

As shown in Figure 4, the bandwidth of a PHCC decreases if the number of protocols used increases, since more protocol switches ($p = 1 - 1/4$) occur, i.e., the active warden's efficiency for PHCCs is positively correlated with p.

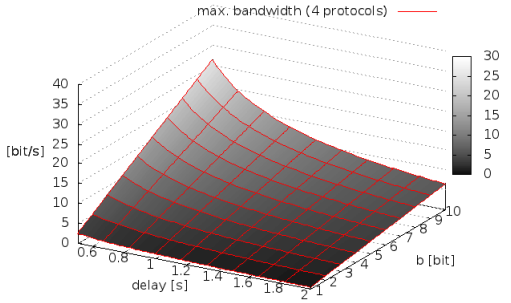


Figure 4. A PHCC's bandwidth using *four* protocols, $T = 0.005$ and delays between 0.5 and 2 as well as the capability to transfer between 1 and 10 bits per packet.

B. Implementation

For our test implementation, we set up a virtual network between two virtualized Linux 3.0 systems (a covert channel sender as well as a receiver with a local active warden

instance) using *VirtualBox* (*www.virtualbox.org*). Both hosts were connected using a virtual Ethernet interface and IPv4. Our proof of concept code focused on layer 4 protocols identifiable by the IPv4 "protocol" field only. Therefore we modified the *protocol channel tool* (pct) [21] that used ARP and ICMP to use UDP and ICMP instead. Additionally, we implemented the functionality to generate randomized input and to adjust the channel's bitrate.

To implement the network delays on the receiver system that is acting as both the active warden gateway and the protocol switching covert channel receiver at the same time, we made use of the firewall system *netfilter/iptables*. Netfilter/iptables provides a "QUEUE" feature which can be used to redirect data packets to a userspace program. Berrange implemented the Perl-based program *delay-net* [22] that enforces configurable network delays using the *IPQueue* module [23] which is based on the iptables QUEUE feature. We modified delay-net in a way that it only delays packets after a protocol switch happened. We also implemented a third program to evaluate the correct transmission at the receiver's side to verify formula 1 (cf. Section III).

Since this test focuses on the protocol switching capability of both the PC and the PHCC at the same time, the testing method is valid for both covert channel types.

III. RESULTS

In our test configuration, the value T is quite small (we measured 0.005 in average) in comparison to the delay time d. As mentioned in the previous section, we were able to determine $p = 0.4738806$ through observing the behaviour of the modified *pct* in our virtual test network. However, p turned out to be slightly higher (0.53) in a real network environment, where protocols occur which do not belong to the PC, and therefore result in few additional protocol switches.

In our test setting we sent PC data using different bitrates and monitored the correctness of the received packet order at the receiver's side. Using this method, we were able to find out the maximum bitrate able to work error-free dependent on the delay introduced by the active warden. Figure 5 shows the results in comparison to our calculation of B (formula 1). The differences between B and our recorded values are small. An active warden with a delay value $d = 2.1\ s$ ($T = 0.005$) reduces the bandwidth limit required for a successful transmission of data to $1bit/s$. If $d = 1.0\ s$, the bandwidth limit is reduced to a maximum of $2.088bit/s$.

PC with improved coding: If a PC uses a coding that requires to send new packets only if a value unequal to the current value is required to be transferred, it can overcome the active warden, if sender and receiver are synchronized. This is possible if the sender only transfers a network packet if a protocol switch occurs, i.e. two packets of the same protocol are never transferred after another. The timing intervals between the protocol switches represent the amount

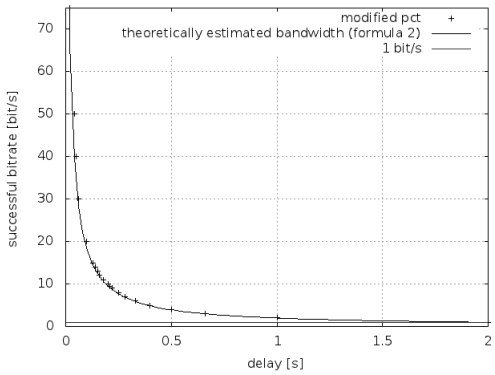


Figure 5. Measured maximal bandwidth of the modified *pct* dependent on the active warden's delay. In comparison, we show formula 1 using the estimated protocol switching value.

of bits to transfer. Thus, such a covert channel would be a hybrid version of a timing channel and a PC.

Example: The sender sends a packet of protocol P_1. The active warden delays it for time d and forwards it. Three more bits as represented by P_1 shall be transferred. Therefore, the sender waits for three time slots. Afterwards, a bit represented by P_2 shall be transferred and the sender sends one such a packet. The active warden delays the packet for d and forwards it. If the sender sends P_1 again, it will also be delayed for d. The receiver will receive P_1, three waiting slots, P_2, P_1, i.e. the same input as was sent by the sender. The only disadvantage introduced by the active warden is the delay of d for all packets but this is a minor consequence.

To overcome this problem, an improved version of the active warden can be developed. In our previous experiments, we focused on an active warden with a constant delay d. If d varies from packet to packet, or in other words, d is randomized (e.g., $d \in [0.1; 2]$ sec), previous packets are likely to overrun newer ones if the timing interval of the sender is too small. Thus, the sender is forced to use big waiting times and thus, will be forced to decrease its bandwidth.

Besides the previously mentioned coding, a PC could also use other codings to improve the amount of bits transferred per protocol switch (b/p). For the default PC coding and two protocols, $p = 0.5$ but when a run length limited (RLL) coding (as used for hard disks [24]) is implemented, p can be decreased.

In case of geometrically distributed symbols, an optimized coding (Huffman coding) can help the covert channel's users to minimize the amount of packets to transfer, but – as usual for covert channel research – we focus on an optimized coding using a uniform distribution (e.g., the covert channel is used to transfer encrypted input).

PHCC: A drawback of our approach is linked to a feature only available for PHCCs but not for PCs. PHCCs provide usually enough covert space to contain sequence numbers in

 ISBN: 978-1-61208-201-1

their payload [16]. Using these sequence numbers, the receiver can reassemble network packets even if their ordering was disturbed [17]. While the active warden is not able to completely solve this problem, it forces a PHCC user to use a sequence number. Such a storage channel-internal sequence number usually consists of only 2-3 bits and thus, the active warden can break the receiver-side sorting nevertheless, if d is large enough. Additionally, since these channels do only provide a few bits per packet, the active warden decreases the available space per packet in this way. Thus, a user is forced to send more packets to transfer the same amount of data than in the case where no active warden would be located on the path between PHCC sender and PHCC receiver.

Receiver-side re-calculation attack for PCs: In case a constant delay is used, it is possible for the attacker recalculate the original sequence of received packets of a PC. However, due to the jitter, it is possible that the attacker is forced to use error-correcting codes. The active warden can implement the previously mentioned randomized d to overcome this problem.

IV. Discussion of Practical Aspects

A goal of the presented active warden approach is to design the system for a practical use. The requirement for only small delays is – even if a user's initial request to a website will be delayed – an acceptable limitation for legitimate traffic in high-security environments since delays of only around 2s can reduce the useful bandwidth of PCs to a maximum of 1 bit/s. For PHCC, the value can differ if the channel provides high values for b. However, to achieve the goal of practical usefulness, it is necessary to implement additional functionality because of the following reasons:

DNS requests: Typically, a user sends DNS requests to a DNS server and, after receiving a response, connects to a system using another protocol. It is required to take care of this typical effect (and similar effects such as using HTTPS right after a user clicked on a link of a HTTP-based website). Protocol switches occur in both cases (DNS→HTTP respectively HTTP→HTTPS). The DNS server and the system a user connects to (usually a web server) will in almost all cases have different addresses, thus it is easy to address this problem if the active warden distinguishes destination hosts.

Different protocols on a single host: However, situations are thinkable in which a user is connected to one host using two different protocols, e.g. to an SMTP server and an IMAP server on the same system. In such cases, whitelisting (e.g., defining trusted hosts) can reduce this problem.

Multiple senders: In an enterprise network, there are usually a number of different computers with Internet access. If the active warden is located on the uplink, it will notice many protocol switches since different systems use different protocols to achieve different tasks. The active warden should distinguish source addresses to solve this problem.

However, some companies run a *network address translation* (NAT) service *within* their network. These systems would appear as a single system to the active warden although the systems as well as the active warden are located inside the company network. Thus, the NAT'ed systems would face delays. A whitelisting is no sufficient solution since these address translated systems are required to be protected from data leakage too. A thinkable limitation for that problem would be to use *remote physical device fingerprinting* [25] to count the number of NAT'ed systems and apply fewer delays per packet switch if the number of hosts behind NAT increases (and vice versa).

Multiple Receivers: If one host sends PC packets to different receivers and each receiver is associated with only a single protocol, no protocol switches occur and no bandwidth is limited on a direct way but in an indirect way: If the receiver is forced to be a distributed system, it has to implement a distributed coordination mechanism (sorting packets and extracting all hidden information on a single system that finally computes the whole hidden message). If the receiver-side network is monitored as well, the coordination itself must be covered too, and thus can probably be detected or at least raise attention. Also, the bandwidth is limited since multiple receivers can receive messages via different performing network access points and the network packets for the coordination can differ in their timings, too. Therefore, the sender must introduce pause intervals (which limit the bandwidth) between new packets to prevent a scrambled result at the receiver.

Redundancy: As all normalizer and firewall-like systems, our prototype of an active warden can result in a single point of failure if not operated on a redundant installation. Modifications of existing redundancy protocols (e.g., CARP) are thinkable to solve this problem. However, as any firewall-like system, the active warden introduces side-effects, i.e., delays, into network traffic.

End-User Limitation: As explained, the effect for end-users is low if the active warden is used. While an extensive end-user study was not part of this work, we measured different HTTP request-response times for 10MByte downloads over the active warden. The standard download time in our network was 0.41-0.57s. After we installed the active warden, we ran a HTTP download as well as a 0.25 bit/s PC to simulate a number of protocol switches as they occur for modern websites (multiple DNS requests for a whole site are normal since they can include script sources from other domains). Our simulation increased the download-time to 0.43-3s. We observed that the basic limitation for connections in the establishment phase where a new protocol (HTTP over TCP) occurred. In the context of the 4s-rule for website rendering [26], we can assume that our active warden is valuable for practical use-cases.

To summarize, all mentioned problems (excepting the network address translation) are solvable by adding the men-

 ISBN: 978-1-61208-201-1

tioned simple features. The configurable delay parameter d provides administrators a way to adjust the efficiency of the active warden to their requirements. Since only protocol switching packets are affected by the active warden, most of a network's traffic is not affected, i.e., download rates and upload rates will not decrease notably.

V. Conclusion

In this paper, we present the first active warden designed to counter both types of protocol switching covert channels: PC as well as PHCC. We limit the useful bandwidth of these covert channels by disturbing the protocol switches through synthetically introduced delays. Therefore, we implemented an active warden and verified its practical usefulness.

Future work will include to find solutions for the problem of network address translation inside a protected network as well as to find solutions for effects of large network environments where load balancing and redundancy protocols are required; the presented prototype was not designed for such environments. Additionally, research must be done to provide an exact bitrate controlling for PHCC using internal sequence numbers since we do only provide a loose bandwidth reduction for this channel type.

References

[1] B. W. Lampson, "A note on the confinement problem," *Commun. ACM*, vol. 16, no. 10, pp. 613–615, 1973.

[2] S. J. Murdoch, "Covert channel vulnerabilities in anonymity systems," Ph.D. dissertation, University of Cambridge, 2007.

[3] C. H. Rowland, "Covert channels in the TCP/IP protocol suite," *First Monday*, vol. 2, no. 5, May 1997, retrieved: Mar, 2012. [Online]. Available: http://firstmonday.org/htbin/cgiwrap/bin/ojs/index.php/fm/article/view/528/449

[4] T. G. Handel and M. T. Sandford, II., "Hiding data in the osi network model," in *Proc. First Int. Workshop on Information Hiding*. London, UK: Springer-Verlag, 1996, pp. 23–38.

[5] S. Cabuk, C. E. Brodley, and C. Shields, "IP covert timing channels: design and detection," in *ACM Conference on Computer and Communications Security*, V. Atluri, B. Pfitzmann, and P. D. McDaniel, Eds. ACM, 2004, pp. 178–187.

[6] M. H. Kang, I. S. Moskowitz, and S. Chincheck, "The pump: A decade of covert fun," in *ACSAC*, 2005, pp. 352–360.

[7] P. A. Porras and R. A. Kemmerer, "Covert flow trees: A technique for identifying and analyzing covert storage channels," in *IEEE Symp. on Security and Privacy*, 1991, pp. 36–51.

[8] R. A. Kemmerer, "Shared resource matrix methodology: an approach to identifying storage and timing channels," *ACM Trans. Comput. Syst.*, vol. 1, no. 3, pp. 256–277, 1983.

[9] J. McHugh, "An information flow tool for gypsy - an extended abstract revisited," in *Proc. 17th Annual Computer Security Applications Conference*, 2001, pp. 191–201.

[10] C. Krätzer and J. Dittmann, "Früherkennung von verdeckten Kanälen in VoIP-Kommunikation," in *IT-Frühwarnsysteme*, ser. BSI-Workshop. BSI, 2006, pp. 209–214, (In German).

[11] M. Handley, V. Paxson, and C. Kreibich, "Network intrusion detection: Evasion, traffic normalization, and end-to-end protocol semantics," in *10th USENIX Security Symposium*, vol. 10, 2001, pp. 115–131.

[12] A. Singh, O. Nordström, A. L. M. dos Santos, and C. Lu, "Stateless model for the prevention of malicious communication channels," *Int. Journal of Comp. and Applications*, vol. 28, no. 3, pp. 285–297, 2006.

[13] G. Gu, R. Perdisci, J. Zhang, and W. Lee, "Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection," in *USENIX Security Symp.*, 2008, pp. 139–154.

[14] S. Zander, G. Armitage, and P. Branch, "Covert channels and countermeasures in computer network protocols," *IEEE Comm. Magazine*, vol. 45, no. 12, pp. 136–142, Dec 2007.

[15] Daemon9, "Loki2 (the implementation)," *Phrack Magazine*, vol. 7, no. 5, September 1997, retrieved: Mar, 2012. [Online]. Available: http://www.phrack.org/issues.html?issue=51&id=6

[16] S. Wendzel and J. Keller, "Low-attention forwarding for mobile network covert channels," in *12th IFIP Comm. and Multim. Security*, ser. LNCS, 2011, vol. 7025, pp. 122–133.

[17] S. Wendzel, "Protocol hopping covert channels," *Hakin9*, vol. 08, no. 03, pp. 20–21, 2008, (in German).

[18] ——, "Protocol channels as a new design alternative of covert channels," *CoRR*, vol. abs/0809.1949, pp. 1–2, 2008.

[19] ——, "Analyse der Präventions- und Detektionsmethoden für verdeckte Kanäle," Master's thesis, Augsburg University of Applied Sciences, June 2011, (in German).

[20] C.-R. Tsai and V. D. Gligor, "A bandwidth computation model for covert storage channels and its applications," in *Proc. IEEE Conf. on Security and Privacy*, 1988, pp. 108–121.

[21] S. Wendzel, "pct," 2009, retrieved: Mar, 2012. [Online]. Available: http://www.wendzel.de/dr.org/files/Projects/pct/

[22] D. Berrange, "Simulating WAN network delay," 2005, retrieved: Mar, 2012. [Online]. Available: http://people.redhat.com/berrange/notes/network-delay.html

[23] J. Morris, "IPTables::IPv4::IPQueue module for Perl," 2002, retrieved: Mar, 2012. [Online]. Available: http://search.cpan.org/~jmorris/perlipq-1.25/IPQueue.pm

[24] C. D. Mee and E. D. Daniel, *Magnetic Storage Handbook*, 2nd ed. McGraw Hill, 1996.

[25] T. Kohno, A. Broido, and k. claffy, "Remote physical device fingerprinting," *IEEE Transactions on Dependable and Secure Computing*, no. 2, pp. 93–108, 2005.

[26] Akamai, "Retail web site performance," 2006, retrieved: Mar, 2012. [Online]. Available: http://www.-akamai.com/dl/reports/Site_Abandonment_Final_Report.pdf

 ISBN: 978-1-61208-201-1

Quantifying the Value of SSL Certification with Web Reputation Metrics

Jani Suomalainen
VTT Technical Research Centre of Finland
Espoo, Finland
Jani.Suomalainen@vtt.fi

Abstract— **Protection in the Internet and World Wide Web is based on the Socket Secure Layer (SSL) protocol and certification authorities, who verify the identities of servers with SSL certificates. Trust in the Web is based on users' perception of sites' trustworthiness and privacy as well as knowledge of servers' monitored behavior. Community-based reputation systems enable users to share their views on servers' trustworthiness. In this paper, we provide a large-scale empirical analysis on the correlation of SSL certification and community-based reputation evaluations. By using publicly available global certificate and reputation databases, we study how availability of SSL support and properties of certificates correlates to users' perception of trust, dependability, and privacy. The paper proposes a metric for revealing the benefits that service providers gain from SSL certification in general, from authority selection, and from extended validation. The proposed reputation metric could provide a mean to quantify the users' valuation of security measures. Hence, it can be utilized when selecting and designing new web security mechanisms.**

Keywords-Web security; Web reputation; Web of Trust; SSL; HTTPS; certification; correlation analysis

I. INTRODUCTION

Authentication and confidentiality of communication in the World Wide Web (WWW) is based on HTTPS (Hypertext Transfer Protocol Secure) [1] and SSL (Secure Socket Layer) [2] protocols as well as X509 public key certificates [3, 4]. The authentication model is scalable and capable of preventing most masquerading attacks when used properly. The model has, however, been criticized due to large amount of equally trusted certification authorities (CAs) and loose certification processes, which make acquiring of phishing certificates possible for attackers. Extended validation [5] certificates and additional visual trust indicators in browsers have been proposed as a more secure certification alternative. However, there have not been large scale studies on the benefits that the service providers gain from SSL certification in general and from extended validation.

Trust in WWW is based on users' perception on the trustworthiness of web sites as well as on reputation of services and service providers. To ease users to decide whether to trust a site or not, reputation services, e.g., Web of Trust (WOT) [6], have emerged. These services enable browsers to show visual warning or block access when the user tries to access a web site with poor reputation. The reputation is a measure determined by monitoring the behavior and content of servers. It can be based on automated analysis or on ratings shared by users.

Servers' support for SSL correlates with servers' security related reputation. SSL makes phishing and other masquerading attacks as well as confidentiality breaches harder. Therefore, it should increase reputation of servers when considering trustworthiness and privacy. The correlation and the causal relation between reputation and SSL are not straightforward or direct. In addition to SSL, other factors affect to the users perception of trust. A service provider that invests to security may also invest other factors increasing the reputation. Nevertheless, the correlation can be used as a one metric when evaluating the usefulness of SSL certification.

This paper contributes by providing a large-scale empirical analysis on the correlation of SSL certification and crowd-based reputation evaluations. Existing work studying effectiveness of SSL certification and warnings in browsers has concentrated on experiments with restricted amount of participants. In this study, we analyze real world data in much larger scale. In our study, the data comes from real deployments and thus cannot be distorted due to laboratory arrangements. The study has two implications. Firstly, we measure the benefits that reputations of web services gain from SSL certification and extended validation as well as from the selection of more reputable certification authorities. Secondly, we introduce a metric that can be used when analyzing impacts and visibility of web security solutions.

The paper is organized as follows. Section II presents related work and motivates our research. In Section III, we describe what data was collected for the analysis. Section IV presents results of statistical analysis on the correlation of SSL certificates and reputation ratings. A discussion on the results and their potential exploitation is provided in Section V. Section VI concludes and summarizes the paper.

II. BACKGROUND AND RELATED WORK

A. SSL Certification

Authentication of web servers is based on X.509 certificates, which have been granted to servers by a trusted CA. In typical browsers (including Mozilla Firefox, Internet Explorer, Google Chrome etc.) the amount of accepted root certificates is large. The acceptance criteria depend on the trustworthiness of CA but also on business and politics. If one of these CAs has been compromised and certifies bogus

 ISBN: 978-1-61208-201-1

servers, the end-users web transactions are in jeopardy. Browser's security identifiers will not warn on bogus servers certified by trusted CA even if it would have been a different CA that actually had signed the victim service. Attacks demonstrating the weaknesses of CAs have already been reported, including the recent DigiNotar and Comodo incidents [7, 8].

Large scale studies on how the certificates are used has been performed by Eckersley et al. [9], who scanned public Internet for certificates and reported several vulnerabilities. Vratonjic et al. [10] analyzed certificates with the million most popular web sites and reported that most HTTPS servers do not use certificates properly. Typical problems are domain mismatch, certificate expiration and untrusted (self-signed) certificates.

Dhamija et al. [11] studied users ability to distinguish real web sites from spoofed sites using SSL warnings. They found that 23% of participants did not check browser's passive security indicators at all when evaluating the trustworthiness of the site. Sunshine et al. [12] performed a survey and a laboratory test to examine users' reactions to different active SSL warnings. They noted that users' behaviour depends on the actual message as well as on the service type. Tests revealed that more than the half of the hundred participants ignored the warnings of the main stream browsers and proceeded to the web sites anyhow. A bit more moderate results were gained by Egelman et al. [13] who found that 21% of sixty study participants ignored active warnings and fell to phishing attacks. When the security indicators and warnings are ignored, the credibility of a web site depends on various other factors. These factors were studied by Fogg et al. [14]. Their study, made with 1400 participants, reveals that real-world feel, ease of use and expertise are the most important categories affecting to credibility.

SSL certificates are assigned to service providers through diverse certification processes. Typically, it is enough that the requester has an access to email, which has been registered for the domain name holder. This makes acquirement of phishing certificates possible for attackers. Some certification authorities may have more trustworthy processes in use but the large amount of equally trusted authorities means that end-users do not have practical means to separate real and trustworthy certifications from bogus certification received from a compromised or careless authority.

Extended Validation Certificates [5] and additional visual trust indicators in browsers have been proposed as a more secure certification alternative. EV certificates are given for servers, which have gone through stricter authentication processes. Browsers identify servers with EV certificates as more trusted by displaying additional trust indicators, notably green address bar. See Figure 1 and Figure 2 for examples of address bar in Mozilla Firefox 8 and Internet Explorer 8 looks when browser connects to services with either unsecure HTTP, (ignored) invalid certificate on HTTPS server, valid regular certificate on HTTPS server, or EV certificate on HTTPS server. EV trust indicators have been supported for a couple of years in the main stream browsers including Microsoft Internet Explorer (since version 7, released October 2006), Mozilla Firefox (version 3, June 2008), Opera (version 9.5, June 2008), Google Chrome (September 2008) and Safari (version 3.2, November 2008).

Figure 1. Security indicators in address bar of Mozilla Firefox 8 (from top to bottom: unsecured HTTP, ignored certificate error, regular certificate, extended validation certificate)

Figure 2. Security indicators in address bar of Internet Explorer 8

The question whether the extended validation increase the security and trustworthiness has been considered by few researchers. Sobey et al. [15] studied whether users notice the additional trust indicators by tracking eye movements of 28 untrained test participants who were making online shopping decisions. They concluded that the validation indicators in Mozilla Firefox 3's address bar went unnoticed for all participants and proposed, as an alternative, more visible and obtrusive trust indicators. Similar results were gained by Jackson et al. [16] studied whether extended validation would help users to detect phishing attacks more easily with a test group of 27 participants and whether security trained users, who had read a help file, are capable to use these indicators. They noted that the trained users did not outperform the untrained users as extended validation did not help users to detect control attacks.

Some researchers have addressed the problems of weak certification by proposing means to determine certificates' trustworthiness and to **limit certificate issuers' authorities**. Marlinspike presented [17] a solution called Converge for turning off all untrusted CAs in a browser. The idea includes a trust management scheme, where other users' views and consensus on particular CAs can be queried from notaries. Another solution called CertLock, presented by Soghoian and Stamm [18], tries to detect suspicious CA changes in certificates. They focus particularly on CA's country of origin and in the prevention of governmental attacks. CertLock uses browsers history information on certificates and warns end-users if CA's country of origin has been changed. In Perspectives [19], presented by Wendlandt et al., a trusted party collects issuer identity information frequently from TLS servers. The browser plugin may then query

 ISBN: 978-1-61208-201-1

whether the issuer has been changed and warn end-user accordingly. A related certificate transparency proposal was made by Laurie and Langley [20]. They proposed that end-users would accept only those certificates, which are available from trusted and public source. The approach would prevent long-life attacks, as service providers could to monitor this public source and suppress fake certificates, claiming their domain names.

B. Web Reputation

SSL certification provides mechanisms for checking that web servers belong to the legitimate entities. However, it does not address whether the server acts inappropriate and expected manner and thus whether the site can be trusted. Untrustworthy web sites can be avoided by using blacklists, containing sites with bad reputation, and whitelists, containing sites with good reputation. Black- and whitelisting can be based either on automated techniques, where server's content is checked against malware fingerprints, or manual techniques, where users evaluate sites' trustworthiness. Human based evaluation is extensive only when a large number of people, a community or a crowd, are participating.

One of the crowd based reputation information providers is WOT. WOT is a company, which collects information from the open community of volunteers. These volunteers evaluate the web sites they visit by using browser add-ons, which are available for Firefox, IE, Chrome, Safari, and Opera. The WOT company was founded July 2006. In November 2011 they reported that their database contains ratings from over 33 million servers.

The strength of WOT is in the detail of information. Evaluation is based on collecting users' subjective ratings, which vary from very poor (numeric values 0-19), poor (20-39), unsatisfactory (40-59) and good (60-79) to excellent (80-100). Ratings are given to four different categories:

1. Trustworthiness – whether the site is safe to use and free of malware and phishing attacks
2. Vendor dependability – whether the commercial actor (e.g., a web shop) behind the server can be trusted and provides good shopping experience
3. Privacy – whether the server is trusted to protect users information appropriately and does not collect private information for vague purposes
4. Child safety – whether the server contains material such as adult content, violence or hateful language, not suitable for the children

In addition to the ratings, WOT provides confidence information for each rating. Confidence is presented by using six different categories and numeric value from 0 to 100. A rating is more credible when large amount of contributors have given similar ratings and when these contributors itself have high individual confidence rating. Individual confidence ratings grow among time when users contribute. WOT does not reveal how the confidence ratings and reputation ratings are exactly calculated to make misuse harder.

Reputation systems are vulnerable for manipulating attacks as discussed by Moore et al. [21] who analyzed a phishing focused service called PhishTank [22]. They noted that the service is dominated by most active users and there is a risk of manipulation by small number of people. The accuracy, completeness and vulnerabilities of the WOT metrics have been analyzed by Chia et al. [23]. They found that WOT was more comprehensive than the compared automated services (Google's Safe Browsing, McAfee's SiteAdvisor and Norton's Safe Web) in detecting malicious domains. They also argued that WOT may be resistant against manipulation attacks due to advanced statistical analysis on the contributors' behavior but that it is still vulnerable for determined malicious gamers. However, as manipulation is likely to affect only restricted amount of servers, it is not likely to distort our large scale statistical studies.

Accuracy of crowd-based reputation systems and black lists has been enhanced by combining results from various heterogeneous sources. For instance, WOT utilizes blacklisting information from PhishTank. Use of quantitative web traffic information was proposed by Sharifi et al. [24], who automated information collection from various web services, including traffic ranking and search engine hits, and analyzed how well this information supports scam detection.

III. Combining SSL Certificate, Web Reputation and Web Rank Data

We collected, combined, and analyzed data from three different repositories as illustrated in Figure 3. First we received SSL certificate database collected in SSL observatory project of Electronic Frontier Foundation (EFF). Secondly, information on web server's popularity was received in form of a list of top million servers produced by Alexa. Then, for the all valid certificates and for all top servers, we requested Web reputation ratings from WOT.

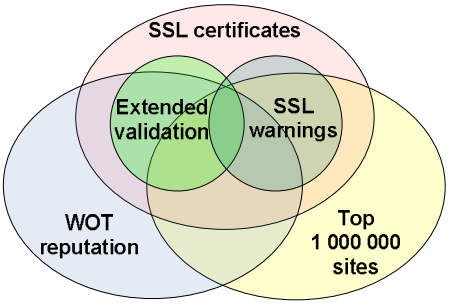


Figure 3. Composition of analysis data

SSL certificates available in the public Internet have been collected in EFF's SSL observatory project [9, 25]. The database contains almost 4 million certificates, including both 'regular' certificates as well as extended validation certificates. We used certificates, which were collected December 2010 and classified as valid by EFF. For the analysis we resolved and selected those HTTPS servers, which had complete domain name (certificates with wild cards in domain names were ignored), were active and fully

working in November 2011. Services were classified as active if the request (to the root directory of the SSL (443) port) resulted a reply larger than 1kB. This limit filtered most servers were HTTPS port is used only for redirection to HTTP port or for some other limited purpose.

List of top million servers, collected by Alexa [26], was used to get domain names of servers, which are really used and frequently visited. This enables comparison between HTTP only servers and servers with HTTPS support. For each server in the list, we collected HTTPS status information indicating whether the HTTPS port was open and whether the connection succeeded without warnings.

WOT reputation metrics were collected for all HTTPS sites as well as for HTTP only sites among top million servers in order to enable comparisons. The confidence limit does not affect substantially to counted averages but it filters out some suspicious ratings. In our analysis, described later, we used only those ratings with reasonable confidence value (12 or higher). Data was collected and analyzed with Linux shell and Perl scripts. SSL status queries and certificate verifications were done on a client based on OpenSSL. Certificates of contacted servers were verified against root certificate list used by Mozilla. MySQL was used as database software. For EFF dataset we found 201,099 active and reputed HTTPS servers and for Alexa dataset we found reputation information for 132,533 HTTP only servers, for 68,961 HTTPS servers, and for 34,985 broken HTTPS servers (showing security warnings when connected).

IV. CORRELATION BETWEEN WEB REPUTATION AND HTTPS SUPPORT

A. *Does the HTTPS Support Increase Reputation?*

The effect of HTTPS support to reputation rankings was studied by calculating average and distribution of reputation values from the Alexa dataset, which contained information from top million servers. The results for trustworthiness and privacy reputation are given in Table 1 and Table 2, respectively. For both metrics the rating for errorless HTTPS support gives around six additional points. Similarly, the amount of poor and very poor rates drops from around 9% to 4% when HTTPS was supported. Additionally we studied how the security warnings, such as domain mismatch or self-signed certificate, affects the ratings. We noted that HTTPS increases trustworthiness only when used correctly. However, even misused SSL based cryptography increases privacy ratings with one point.

TABLE 1. TRUSTWORTHINESS REPUTATION OF SERVERS WITH AND WITHOUT SSL SUPPORT AND WITH BROKEN SSL SUPPORT SHOWING WARNINGS

Server type / count	Avg	Distribution (%)				
		Ex.	G	***Uns.***	Poor	***VP***
HTTPS / 13,497	84,7	84,5	9,5	1,8	1,0	3,1
Broken HTTPS / 9,483	78,7	73,1	13,4	4,1	2,5	7,0
HTTP only / 41,250	78,6	72,1	13,8	5,0	2,5	6,5

TABLE 2. PRIVACY REPUTATION OF SERVERS WITH AND WITHOUT SSL SUPPORT AND WITH BROKEN SSL SUPPORT SHOWING WARNINGS

Server type / count	Avg	Distribution (%)				
		Ex.	G	***Uns.***	Poor	***VP***
HTTPS / 13,001	84,9	86,0	8,1	2,0	1,1	2,8
Broken HTTPS / 8,776	80,0	73,7	13,1	4,9	2,4	5,8
HTTP only / 37,197	78,9	73,4	13,0	6,6	2,8	6,2

The servers in HTTPS category may have also the HTTP port open. Hence, we cannot say whether the user evaluations were done in the HTTPS secured connection or not. From the larger EFF dataset, we found servers that had only HTTPS port active. For 431 servers the average trust value was 86,6 (when the average value for all HTTPS servers in 'EFF dataset' was 85,8). The privacy ratings for 371 servers were 87,9 (and 87,1 for all). This small sample indicates that reputation of servers supporting only HTTPS would be even larger.

We studied also how trustworthiness and privacy reputations correlate with the popularity of server. Sliding averages presented in Figure 4 illustrate that the better ranking Alexa increases trustworthiness and privacy value. The difference of reputation between secured and unsecured is visible despite the popularity, though the difference is smaller with more popular servers.

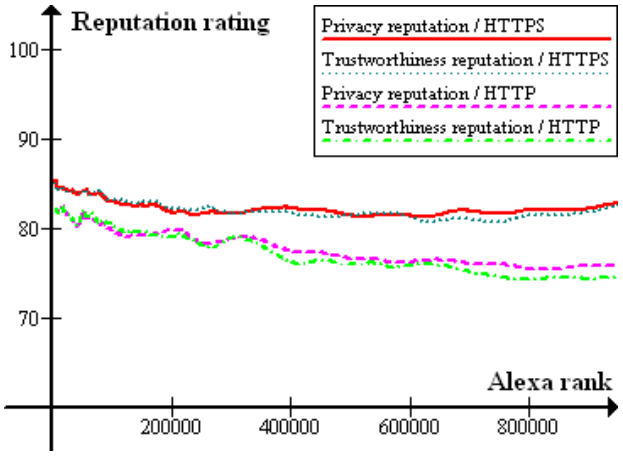


Figure 4. Dependency between reputations and popularity

B. *Differences between CAs*

There are clear differences between the reputation of servers certified by different CAs. The following table presents results of CAs, which all had more than thousand valid certificates used by active and trustworthiness ranking with reliability at least 12 points servers within 'EFF dataset'. The results show a difference of over 10 points between the averages of the best and the worse CAs. The difference is even significant when looking at the ratio of poor and very poor sites: increase from close zero to 7,4%. Different CA brands provided by one company have not been combined in the table. For example, Comodo is also the provider of The Usertrust Network and Terena certificates, Symantec is the owner of Verisign and Thawte.

 ISBN: 978-1-61208-201-1

TABLE 3. TRUSTWORTHINESS REPUTATION OF SERVERS CERTIFIED BY DIFFERENT CAS

CA / certificate count	Avg	Distribution (%)				
		Ex.	G	***Uns.***	Poor	***VP***
Cybertrust / 1061	89,3	96,6	3,0	0,2	0,0	0,2
Verisign / 9993	88,7	92,1	6,0	0,8	0,4	0,7
Terena / 1410	88,6	95,7	4,3	0,0	0,0	0,0
Entrust / 1747	88,1	92,8	4,6	1,4	1,0	0,2
Thawte / 5506	85,9	85,3	10,7	1,6	1,0	1,3
Usertrust N. / 1994	83,9	77,4	18,7	1,0	1,0	2,0
Equifax / 4828	82,0	74,0	19,0	1,9	1,3	3,8
Comodo / 1557	81,9	75,8	16,2	2,1	0,7	5,3
GoDaddy / 2973	79,0	67,5	22,7	2,5	1,8	5,6
Total / 39482	*85,8*	*84,6*	*11,6*	*1,2*	*0,8*	*1,8*

C. The Value of Extended Validation Certificates

Extended validation provides only small or no increase of reputation at all. Table 4 compares trustworthiness and privacy values of EV certificates to non-EV certificates within the EFF dataset. Trustworthiness average is 0,7% higher and privacy value is 0,5% smaller.

TABLE 4. TRUSTWORTHINESS AND PRIVACY REPUTATION OF SERVERS WITH REGULAR OR EXTENDED VALIDATION CERTIFICATES

Certificates / count	Avg	Distribution (%)				
		Ex.	G	***Uns.***	Poor	***VP***
Trustworthiness						
Regular / 36297	85,7	84,5	11,7	1,2	0,7	1,9
EV / 3185	86,4	85,8	9,9	1,8	1,2	1,3
Privacy						
Regular / 32166	87,1	88,2	7,3	1,5	0,7	2,3
EV / 2839	86,6	87,1	7,6	2,3	1,2	1,7

Table 5 describes CA specific trustworthiness ratings for CAs with more than 100 EV certificates. When comparing to CA specific numbers to generic CA results in Table 3, there is a small increase of reputation all CAs except for the largest EV provider. For Verisign the EV rate is 0,7% smaller than the rate for all Verisign certificates.

TABLE 5. TRUSTWORTHINESS REPUTATION OF SERVERS EV CERTIFIED BY PARTICULAR CAS

CA / certificates	Avg	Distribution (%)				
		Ex.	***G***	***Uns.***	***Poor***	***VP***
Cybertrust / 255	89,9	100	0	0	0	0
Verisign/ 1688	88,0	91,0	5,3	1,9	3,5	0,9
Thawte/ 183	86,2	85,2	8,7	3,3	33,9	0,0
Comodo / 226	83,2	81,0	11,5	0,9	6,6	4,9
Globalsign/ 366	83,1	70,2	25,7	1,9	2,2	1,1

V. DISCUSSION

A. The Value of SSL and the Limitations of the Metric

Our intuition was that the support for HTTPS affects to reputation in two manners: Visibility of security indicators may increase it and security warning indicators and dialogs as well as published security problems will decrease the reputation. However, service providers who are willing to invest more on HTTPS are typically also willing to invest on other factors increasing reputation. The reputation is not a result of HTTPS support. Instead, they are both results of security efforts. However, even though the correlation does not imply causality, it indicates possible causes. Future research is needed to understand, in more detail, what is the value of SSL certification and what is the value of other factors contributing to reputation.

The results show that there is a clear correlation between HTTPS support and Web reputation. The reputation average of valid SSL certificates was significantly higher than the average of servers with broken certificates. Hence, it seems to pay off to have a working HTTPS support.

The difference of reputation average between the best CA and the worst CAs was significant. Certification authorities are not typically selected from the security perspective, instead price, compatibility with browsers and easiness are likely to be more important factors. Hence, the correlation may not be used to indicate of weak certification procedures but it can be used to characterize attackers' probable selections.

The difference between regular and extended validation certificates was insignificant. Since EV certificates are more expensive it would be likely that these service providers would had invested also in other factors contributing sites trustworthiness. For that reason we expected the trustworthiness ratings for EV certificates to be higher. Detected correlation seems to indicate that the additional trust indicators in browsers (Figure 1 and Figure 2) are undetected by the users. This result confirms the previous small scale end-users studies that trust indicators are ignored. Hence, according to these results we could ask why to pay an extra for extended validation.

We did not analyze differences between applications and business sectors. It may be likely that HTTPS and extended validation are typically used in more critical services, such as banks, and that WOT contributors valuate these services differently or more carefully. In the future, it should be studied how the application field affects to the reputation.

Reputation systems may utilize information on the SSL certification. Currently, WOT collects information on newly discovered phishing attacks from PhishTank and adjust reputations accordingly. Similarly, knowledge that a server has a valid certificate may increase trustworthiness and privacy reputation values of the domain name.

B. How to Utilize the Reputation Metrics?

Reputation metrics provide us a mean to quantify users' perception of security. These metrics provide researchers a better understanding on the effectiveness and impact of security mechanisms. Hence, the metrics can be valuable when developing new useful security solutions. Also, the information on the correlation can be used by decision makers, when analyzing which security mechanisms are needed and provide enough benefits to justify the investments.

Metrics may be used also to enhance applications for existing web security solutions. Specially, they are usable in notary based CA selection approaches. For instance, in Convergence [17], the browser trusts only those SSL certificates which have been certified by CAs, which are accepted by particular notaries. However, it may be difficult

 ISBN: 978-1-61208-201-1

for notaries to know which CAs to trust. Reputation gives notaries a tool, formal metric, which can be used when evaluating CAs' trustworthiness. This would act as an incentive for CAs to verify services more thoroughly, as root certificates with bad trustworthiness averages could be considered as untrusted in some browsers.

VI. Conclusions and Future Work

In this paper, we studied the correlation between web reputation metrics (particularly WOT trustworthiness and privacy values) and SSL certification. Web reputation metrics provide researchers a statistical mean to quantify users' perception of trust and privacy and, hence, impact and effectiveness of security solutions. The results of our-large scale HTTPS/SSL correlation analysis reinforce the doubts on the inefficiency of the extended validation in SSL certification. They also reveal the differences between servers certified by different authorities.

In the future, more studies and analysis is needed to fully understand the causal relation between security mechanisms and end-user's perception of security. We need to study differences between particular web service categories and within selected services in order to understand all the contributing factors.

References

[1] E. Rescorla. HTTP Over TLS. IETF Specification. 2000. http://www.ietf.org/rfc/rfc2818.txt.

[2] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) Protocol. Version 1.1. IETF Specification. 2006. http://www.ietf.org/rfc/4346.txt.

[3] International Telecommunication Union. ITU-T Recommendation X.509. 2008. http://www.itu.int/rec/T-REC-X.509-200811-I/en.

[4] R. Housley, W. Ford, W. Polk, and D. Solo. Internet X.509 Public Key Infrastructure, Certificate and CRL Profile. IETF Standard. 1999. http://www.ietf.org/rfc/rfc2459.txt.

[5] CA/Browser Forum. Guidelines For The Issuance And Management Of Extended Validation Certificates. Version 1.3. 2010. http://www.cabforum.org/Guidelines_v1_3.pdf.

[6] Web of Trust . WWW site. Available: http://www.mywot.com/. [Accessed March 3rd 2012].

[7] G. Keizer . Hackers may have stolen over 200 SSL certificates. Computerworld. 2011. http://www.computerworld.com/s/article/9219663/Hackers_may_have_stolen_over_200_SSL_certificates. [Accessed March 3rd 2012].

[8] Mills, E. Comodo: Web attack broader than initially thought. CNET, 2011. Available: http://news.cnet.com/8301-27080_3-20048831-245.html;. [Accessed March 3rd 2012].

[9] P. Eckersley and J. Burns. An Observatory for the SSLiverse. Presentation at DEFCON 18. 2010.

[10] N. Vratonjic, J. Freudiger, V. Bindschaedler, and J. Hubaux. The Inconvenient Truth about Web Certificates. The Workshop on Economics of Information Security (WEIS). Fairfax, Virginia, USA, 2011.

[11] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. Proceedings of the SIGCHI conference on Human Factors in computing systems. Montreal, Quebec, Canada, 2006. CHI '06, pp. 581-590.

[12] J. Sunshine, S. Egelman, H. Almuhimedi, N. Atri, and L. F. Cranor. Crying wolf: an empirical study of SSL warning effectiveness. Proceedings of the 18th conference on USENIX security symposium. Montreal, Canada, 2009. SSYM'09, pp. 399-416.

[13] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: an empirical study of the effectiveness of web browser phishing warnings. Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems. Florence, Italy, 2008. CHI '08, pp. 1065-1074.

[14] B.J. Fogg, et al. What makes Web sites credible?: a report on a large quantitative study. Proceedings of the SIGCHI conference on Human factors in computing systems. Seattle, Washington, United States, 2001. CHI '01, pp. 61-68.

[15] J. Sobey, R. Biddle, P. C. Oorschot, and A. S. Patrick. Exploring User Reactions to New Browser Cues for Extended Validation Certificates. Proceedings of the 13th European Symposium on Research in Computer Security: Computer Security. Malaga, Spain, 2008. ESORICS '08, pp. 411-427.

[16] C. Jackson, D. R. Simon, D. S. Tan, and A. Barth. An evaluation of extended validation and picture-in-picture phishing attacks. Proceedings of the 11th International Conference on Financial cryptography and 1st International conference on Usable Security. Scarborough, Trinidad and Tobago, 2007. FC'07/USEC'07, pp. 281-293.

[17] M. Marlinspike. SSL and the future of authenticity. Presentation at BlackHat USA . 2011. http://www.youtube.com/watch?v=Z7Wl2FW2TcA. [Accessed March 3rd 2012].

[18] C. Soghoian and S. Stamm. Certified Lies: Detecting and Defeating Government Interception Attacks Against SSL. 3rd Hot Topics in Privacy Enhancing Technologies, 2010. HotPETs 2010, pp. 50-61.

[19] D. Wendlandt, D. G. Andersen, and A. Perrig. Perspectives: improving SSH-style host authentication with multi-path probing. USENIX 2008 Annual Technical Conference. Boston, Massachusetts, 2008. ATC'08, pp. 321-334.

[20] B. Laurie and A. Langley. Certificate Authority Transparency and Auditability, 2011. http://www.links.org/files/CertificateAuthorityTransparencyandAuditability.pdf.

[21] T. Moore and R. Clayton. Evaluating the Wisdom of Crowds in Assessing Phishing Websites. Proceedings of the Financial Cryptography and Data Security, 2008. FC'08, pp. 16-30.

[22] PhishTank. Web site. Available: http://www.phishtank.com/. [Accessed March 3rd 2012].

[23] P.H. Chia and S. J. Knapskog. Re-Evaluating the Wisdom of Crowds in Assessing Web Security. Proceedings of the Financial Cryptography and Data Security. 2011.

[24] M. Sharifi, E. Fink, and J. G. Carbonell. Detection of Internet scam using logistic regression. Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics. 2011. Pp. 2168-2172.

[25] Electronic Frontier Foundation. The EFF SSL Observatory. Available: https://www.eff.org/observatory. [Accessed March 3rd 2012].

[26] Alexa. Top 1,000,000 sites. 2011. Available: http://s3.amazonaws.com/alexa-static/top-1m.csv.zip. [Accessed November 28th 2011].

 ISBN: 978-1-61208-201-1

Parameter Estimation for Heuristic-based Internet Traffic Classification

Michael Finsterbusch, Jean-Alexander Müller, Chris Richter
Dept. of Communication and Computer Science
Hochschule für Telekommunikation Leipzig (University of Applied Sciences)
Leipzig, Germany
{*finsterbusch|jeanm|richter*}*@hft-leipzig.de*

***Abstract*—Accurate traffic classification is necessary for many administrative networking tasks like security monitoring, providing Quality of Service and network design or planning. We apply 18 machine learning algorithms to classify network traffic based on six classes of statistical parameters. In contrast to other studies, we use a per-packet approach instead of a per-flow approach to make it possible to use the classification results for real-time network interception. In this paper we illustrate the accuracy of the algorithms with different parameter combinations with the goal to reduce the amount of necessary parameters needed for high accuracy traffic classification. Our results indicate that some parameter combinations can be used to classify a large number of protocols. We identified algorithms with good and worse classification accuracy and algorithms which need much time for classification, so that they cannot be used for real-time classification.**

Keywords-flow classification, Internet traffic, traffic identification.

I. INTRODUCTION

Network traffic classification or particularly application classification and identification is the process of identifying the type of application (or the protocol) that generates a particular network flow. There is a growing need for traffic classification. Many tasks that are necessary for network operation and management as well as for business models based on providing network access can benefit from traffic classification. Traffic classification can be used for QoS (quality of service) mapping, traffic shaping, access control, content control/filtering, intrusion detection and prevention, trend analysis, monitoring, lawful interception, content optimization, billing and metering, load balancing, traffic engineering, network planning, etc.

In general, there are four kinds of traffic classification methods. The oldest and most common method is the *port based* approach. This uses the well-known-port numbers of the TCP/UDP protocols assigned by the IANA. Many client-server applications or protocols use asymmetric port numbers for client and server, which means that client port and server port differ. The port based method mostly refers to the server port to identify an application. But the server port can be set to any port number by the server administrator. Not every protocol own well-known port numbers or they use dynamic ports like P2P (peer-to-peer) protocols. By using tunnels, this method fails too. Therefore, we cannot trust in this method.

Another method used is *protocol decoding*. It is based on stateful reconstruction of sessions and application information from packet content. It identifies protocols by their characteristic protocol headers (magic numbers, incrementing counters, session identifiers, etc.), packet sequences, etc. so it avoids needing to trust in port numbers. This method provides high accuracy but it is very expensive. Every protocol detection must be implemented manually and in-depth knowledge of the entire protocol and the network environment is necessary. Problems of this method are the amount of network protocols and keeping it up to date. Furthermore proprietary protocols must be reverse engineered and encrypted traffic is out of scope for protocol decoding. Therefore this method is often used only for dedicated popular protocols, e. g., HTTP and mail protocols like in Cisco's Network Based Application Recognition (NBAR) [1].

The third method is the *pattern* or *signature based* approach [2]. This method uses application specific signatures and searches for those in the protocol header and content to identify the application. The problem of this method is to find good pattern. Furthermore pattern matching for many patterns across all network traffic can become very expensive especially for higher data rates. Additionally, the protocol detection can take up to 100 seconds or more [2].

The fourth method is based on the *machine learning* approach. This method uses machine learning algorithms as used in data mining to identify applications by characteristic packet or flow statistics. The advantage of this approach is that the algorithms can be trained with real network traffic. If a protocol changed or a new protocol appeared, it is easy to repeat the training to update the protocol identifier. Probably it can be used to identify some encrypted protocols. A problem of this method is to find the proper parameters and effective machine learning algorithms.

The machine learning approach has been discussed in numerous papers [3], [4], [5], [6], but with focus on just one algorithm. In this paper we evaluate a wide range of candidate algorithms and parameters. Furthermore, we evaluate which of these algorithms are suitable for real-time traffic classification. In this context, real-time means that traffic management (e. g., traffic shaping) can be done

 ISBN: 978-1-61208-201-1

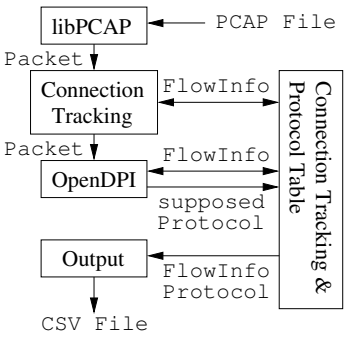


Figure 1: Block diagram of automatic flow labelling

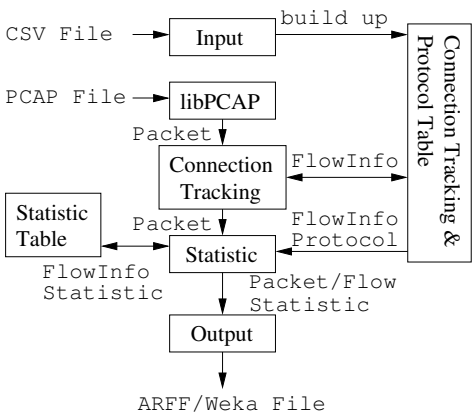


Figure 2: Block diagram of statistics computation

immediately. This approach is mostly used for non real-time or offline network traffic analysis [3], [4], [6].

The remainder of this paper is structured as follows: Section 2 contains a description of the traffic classification used as the basis for our research. The statistical parameters applied and the classifying algorithms examined can be found in the succeeding Sections 3 and 4. Section 5 describes the experimental setup of our research and Section 6 covers the parameter reduction. Finally, Section 7 provides a conclusion and the direction for our future work.

II. Traffic classification Tools

To test and train the different machine learning algorithms (see Section IV) we used different network traffic traces in PCAP (packet capture library) format [7]. To extract the necessary information from these capture files, we developed a tool to generate adequate input data to the algorithms. We had to develop our own tool because available tools like GTVS (Ground Truth Verification System) [8] do not fulfil our constraints for the automatic traffic labelling and we used many different statistical parameters (see Section III), which were not included in GTVS.

The processing and labelling of the PCAP file is done in two consecutive steps: First, the automatic protocol detection and labelling and second, computation of statistics. The result of the first step is a text file in CSV (comma separated values) format. This CSV file is used for manual post-processing to check for correctly labelled flows or to label unknown — not automatically detected traffic — manually. To detect and label the flows automatically we use OpenDPI [9], which is a library that performs protocol detection by stateful reconstruction of session and application information from packet content. Other tools, e. g., GTVS [8] use heuristics to classify and label the traffic. But we decided not to use this tool because it is based on port numbers, which are not aware of, e. g., tunnels etc. (see Section I).

The structure and function of the automatic traffic labelling is shown in the block diagram on Fig. 1. The libPCAP-API [7] is used to read single packets from a PCAP file. A connection tracking is needed to work on a per-flow basis. The connection tracking contains a TCP state machine. As a result the connection tracking provides only those packets belonging to a correct TCP flow that was already seen from beginning of the flow (3-way-handshake). Many protocols may only be detected if the first packets of the flow — which in many protocols contain a handshake — are present (e. g., SSL/TLS). Therefore the connection tracking is very important. To deal with the stateless nature of UDP we used timeouts to split the UDP traffic for the same 5-tuple into different flows. The necessary flow information (e. g., current state machine state) is saved to the connection tracking table with the 5-tuple (source/destination IP address, source/destination port, the IP protocol field value) as key.

The second step in PCAP processing is to compute the statistical parameters. Fig. 2 shows the block diagram of the flow based statistic generation. The CSV-File from first step is used to build up a flow table that contains the detected protocol, the flow start timestamp and the 5-tuple as key. The start timestamp is prerequisite to differentiate flows with the same 5-tuple. The computation of the statistical parameters is done for every packet of a flow. A detailed description of the parameters and their computation is in Section III.

The output of the statistics processing is in ARFF (Attribute-Relation File Format)/WEKA file format [10], see Section IV. The output contains separate training and testing data. Furthermore there are two kinds of statistical data: (1) per-packet data (2) per-flow data. The per-packet data can be used to determine if the classifiers/algorithms can be used for real-time classification tasks. The per-flow data can be used to determine the quality of the classifiers/algorithms applied to offline data or non real-time data like NetFlow (NetFlow is an industry standard protocol for traffic monitoring by collecting IP traffic information) data used for monitoring or forensics.

Traffic classification on a per-packet basis makes it possible to intercept the network traffic for management and

 ISBN: 978-1-61208-201-1

traffic engineering in real-time. On a per-flow basis the flow has finished when it is classified. So, the classification can only be used for monitoring or statistics.

III. Statistical Parameters

The basis for traffic classification using heuristic algorithms are objects which can be classified by the algorithms used. For this approach, we are using traffic flows as the foundation for our classification. We define a traffic flow as one or more related IP-packets between two connected hosts. Each flow is characterized by a 5-tuple consisting of the source and destination IP-address, the network layer protocol number and the source and destination port (referring to TCP and UDP). For TCP based flows we are using only complete TCP flows. This implies that the investigated TCP flows have a 3-Way-Handshake for the connection establishment and a connection termination, for example with the 4-Way-Handshake or the reset flag (RST). To collect those complete TCP flows we used a TCP state machine we implemented in the tool described in Section II. Incomplete or fragmented TCP flow traces are subject of our further ongoing research.

A. Flow parameters

Besides the 5-tuple as the unique qualifier, each flow is described by additional parameters. The most parameters are differentiated for the whole flow, for upstream and for downstream. The reason for this differentiation can be found in the particular differences in upstream and downstream behaviour of various protocols, e. g., HTTP. Our selection of flow characterizing parameters is shown in Table I.

According to [11] we describe each flow with three modes:

- **idle**: A flow is idle when no packets were sent between the two hosts for more than two seconds.
- **bulk**: The bulk mode is defined as the time when there are more than three packets being successfully delivered in the same direction without any packet with data in the other direction.
- **interactive**: When there are packets sent in both directions, the flow is in the interactive mode.

Because the interactive mode is also defined as the time when the flow is not in the idle or in the bulk mode — which means that the interactive mode correlates with the other two modes, we decided to gather only the duration and quota for idle and bulk mode. Furthermore we provide information about the interarrival time, whole flow duration, packet count and the payload size.

B. Parameter computation

As shown in [11], there are more possible flow characterizing parameters than we present in Table I. But, a lot of these parameters are not correlated with the used protocol. Instead, they are only influenced by the transport network, e. g., the Internet. For example, the total number of duplicated SACK packets ([11]: parameter no. 39 and no. 40) only indicates packet loss in the network and is a result of TCP congestion control. These network influences are independent from the protocol characteristics.

Also, often the transport layer (UDP and TCP) port numbers are used as flow characteristics [4], [5], [11], [12]. Because of the use of network address translation (NAT) or by intentional administrative changes of server port numbers, these characteristics are not reliable for traffic classification. Other parameters, like the median of bytes on wire, are depending on the used network stack implementation. The number of bytes on wire, e. g., Ethernet, are influenced by the different possible options in the IP and TCP header or the use of IPv4 and IPv6. Thus, there will be a packet with the same transport layer payload with a different number of bytes on the wire and this is not dependent on the classifying protocol.

The target of our research is the usage of heuristic algorithms for real-time traffic classification. Therefore, we cannot compute the flow characterizing parameters at the end of the flow. Instead, we compute all parameters after each packet of the flow is received. Hence, we compute moving averages, like the median interarrival time or the average byte count of the network layer payload. Required aggregate values are also computed as moving values for each received packet of the flow.

IV. Algortihms

To classify the network traffic we are using different machine learning algorithms, which are also called classifiers. All the classifiers we used are off the shelf machine learning algorithms and we treat them as black-box classifiers. For this work we used the WEKA software suite [10]. The WEKA suite is written in Java language and contains a collection of machine learning algorithms for data mining tasks. WEKA provides a uniform interface for all classifiers, which made it easy to automate training and testing of different classifiers.

A. Train and Test suite

The records, generated by the tool described in Section II and used as training and testing data for the classifiers, are provided as ARFF files. An ARFF file is an ASCII text file that contains (1) a header Section, which describes the used parameters/attributes and (2) the data Section, which contains the records. Each record is represented by one line containing all parameters in a comma separated list. The training record contains all statistical parameters and the associated protocol. Therefore, we can use the supervised learning approach for the machine learning algorithms. The test records contain only the statistical parameters.

If a classifier is trained by WEKA it generates a classifier model that can be saved to a file. In a test case this model can be used by the classifier to make a prediction of the

 ISBN: 978-1-61208-201-1

Table I: Used flow and packet parameters and their description

No.	Class	Short Name	Description
1	1	packet_cnt	packet count whole flow
2	1	packet_cnt_us	packet count upstream
3	1	packet_cnt_ds	packet count downstream
4	2	intarv_time_med	median interarrival time whole flow
5	2	intarv_time_max	maximum interarrival time whole flow
6	2	intarv_time_min	minimum interarrival time whole flow
7	2	intarv_time_med_us	median interarrival time upstream
8	2	intarv_time_max_us	maximum interarrival time upstream
9	2	intarv_time_min_us	minimum interarrival time upstream
10	2	intarv_time_med_ds	median interarrival time downstream
11	2	intarv_time_max_ds	maximum interarrival time downstream
12	2	intarv_time_min_ds	minimum interarrival time downstream
13	3	bytes_payload_l4_med	median byte count of L4 payload whole flow
14	3	bytes_payload_l4_max	maximum byte count of L4 payload whole flow
15	3	bytes_payload_l4_min	minimum byte count of L4 payload whole flow
16	3	bytes_payload_range	(maximum - minium) byte of L4 payload whole flow
17	3	bytes_payload_l4_med_us	median byte count of L4 payload upstream
18	3	bytes_payload_l4_max_us	maximum byte count of L4 payload upstream
19	3	bytes_payload_l4_min_us	minimum byte count of L4 payload upstream
20	3	bytes_payload_range_us	(maximum - minium) byte of L4 payload upstream
21	3	bytes_payload_l4_med_ds	median byte count of L4 payload downstream
22	3	bytes_payload_l4_max_ds	maximum byte count of L4 payload downstream
23	3	bytes_payload_l4_min_ds	minimum byte count of L4 payload downstream
24	3	bytes_payload_range_ds	(maximum - minium) byte of L4 payload downstream
25	4	duration_flow	whole connection duration
26	4	duration_flow_us	connections duration on upstream
27	4	duration_flow_ds	connections duration on downstream
28	5	changes_bulktrans_mode	number of transitions between bulk- and transfermode
29	5	duration_bulkmode	time spent in bulk transfer mode
30	5	duration_bulkmode_us	time spent in bulk transfer mode for upstream
31	5	duration_bulkmode_ds	time spent in bulk transfer mode for downstream
32	5	quota_bulkmode	percentage of quota of bulk transfer mode (per mille)
33	5	quota_bulkmode_us	percentage of quota of bulk transfer mode (per mille) on upstream
34	5	quota_bulkmode_ds	percentage of quota of bulk transfer mode (per mille) on downstream
35	6	time_idle_sum	time spent in idle mode for whole flow
36	6	time_idle_sum_us	time spent in idle mode for upstream
37	6	time_idle_sum_ds	time spent in idle mode for downstream
38	6	quota_time_idle	percentage of quota of time spent in idle mode (per mille)
39	6	quota_time_idle_us	percentage of quota of time spent in idle mode (per mille) on upstream
40	6	quota_time_idle_ds	percentage of quota of time spent in idle mode (per mille) on downstream

protocol which was the origin of the statistical record. Due to the generation of training and testing data from one PCAP file by the tool in Section II, we can evaluate the quality of prediction by comparing the classifier results with the training records containing the protocol. This PCAP file must be a different file than the one which was used to generate the classifier model. To automate training and testing of different classifiers we wrote a test suite on top of WEKA. Fig. 3 shows a block diagram of the test suite. The test suite provides parallel WEKA instances to speed up training and tests. Furthermore, the test suite can measure the memory and time consumption the classifiers need for training and testing.

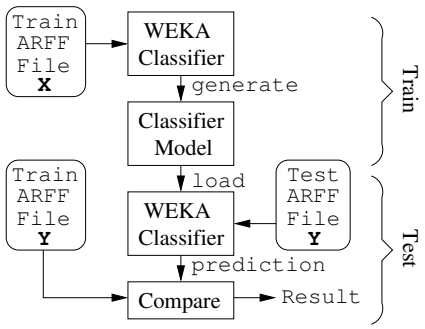


Figure 3: Block diagram of the test suite

B. Preselection of Algorithms

The WEKA software suite contains more than 100 classifiers. So, it was necessary to choose the best candidates of these classifiers to reduce the amount of time to train, test and evaluate the classifier results.

In a first preselection of the classifiers, we used a training-set that was also used as test-set. This means a good classifier should correctly predict a preponderant part of data. We dismissed classifiers with accuracy below 90%. The second

criterion for preselection was the time needed for training and testing. The third criterion was the memory consumption of the classifiers. If a classifier needed more than 2 GByte of RAM for this test, it was also dismissed. The preselected classifiers are shown in the left row of Table II.

We used the classifiers as black-box with the default parameters as proposed by WEKA.

V. Experimental Setup

For training and testing we used the test suite described in Section IV. The used data and the results of our experiments are shown below in this section.

A. Used Data

During this study we have used data collected from different sources. We manually generated network traffic and captured network traffic from the campus network. Additionally we used trace files from [13], [14].

All captured network traffic was automatically classified with the tool described in Section II. The whole traffic was verified by hand. False or not classified traffic was completed by hand. Then, we split this data into two parts and generated the training and testing data in ARFF-format. Table III contains the composition of network packets for training and testing data.

Many of the investigated protocols use cryptographic encryption to protect their data, making it difficult to classify these protocols. Bittorrent, eDonkey, IMAP, POP3 and SMTP use encryption. IMAP, POP3 and SMTP can cryptographically protect their communication before they start their session or they can request encryption during the session (STARTTLS). In the second case the establishment of the session is not encrypted. Our data contains both kinds of IMAP, POP3 and SMTP traffic. The other protocols do not use encryption. Also, the HTTP traffic contains only unencrypted traffic; it does not contain HTTPS (TLS) traffic.

The results of the test run for all classifiers with the computed classifier-models and the particular test data are deterministic.

B. Results

Table II contains the results of this study. It shows if it is possible for an algorithm with any parameter combination to detect a protocol with an accuracy greater than or equal to 90%.

Additionally, Table II contains the time needed for training and testing. These times are measured by using the HPROF tool for heap and CPU profiling, which is part of the Java Virtual Machine. We measured the CPU usage time for every algorithm and used the fastest algorithm (OneR) as reference to scale the timing results. As we can see, in Table II the amount of time spent for training is much higher than for testing. But, this is not a problem in general. Training is done only once but testing is done permanently for protocol classification. All the algorithms have very similar CPU time consumption, but four algorithms (BayesNet, NaiveBayes, NaiveBayesUpdateble and ND) have a notedly higher CPU time consumption. This could make these four algorithms unusable for real-time traffic classification. Furthermore, the algorithms NaiveBayes and NaiveBayesUpdateble provide nearly the same results, which can be seen in Table II as well as in Fig. 4.

It can be seen from Table II that not all used algorithms are suitable for protocol classification with the selected statistical parameters. Also, it shows that the observed protocols have very different behaviour, so that some could be detected with high accuracy (Bittorrent, RTP, Flash) while other protocols (eDonkey, IMAP) are hard to detect.

Table III: Used training and testing data (number of packets)

Protocol	Training	Testing	Encryption
Bittorrent	7772	10461	yes
eDonkey	19961	48420	yes
Flash	21373	21212	no
HTTP	4743	1236	no
IMAP	2797	1025	yes
Oscar	315	257	no
POP3	673	3044	most
RTP	24404	57428	no
SIP	414	760	no
SMTP	560	774	yes

VI. Parameter Reduction

As described in Section III and listed in Table I, we used 40 parameters, which can be divided into six different parameter classes. Table II shows on which protocols our 18 preselected classifiers have a classification accuracy with greater than or equal to 90%. However, Table II does not show which parameter combinations were suitable for the best classification results.

One important target of our research is the reduction of the used parameters — first, to eliminate unnecessary or disruptive parameters according to the classification accuracy and second, to eliminate parameters and thus reduce the complexity of the classification for optimization according to real-time classification. Therefore, we tested all classifiers with all possible combinations of the parameter classes from Table I. Hence, we tested our 18 classifiers with our training PCAP trace and all 63 combinations of the parameter classes — the 64th combination (all classes removed) was omitted. The results of these tests are evaluated below in this section.

The evaluation of the test results are shown in different histograms. Each parameter combination is expressed as a decimal number in these histograms (see Fig. 4, Fig. 5, Fig. 6 and Fig. 7). These decimal numbers are the sum of the values of each parameter class as shown in Table IV. For example, the combination of the payload size (4) and bulk (16) parameters is the decimal number $20 = 4 + 16$.

 ISBN: 978-1-61208-201-1

Table II: Protocol classification with greater than or equal to 90% accuracy and time factor

WEKA Classifier Name	Bittorrent	eDonkey	Flash	HTTP	IMAP	Oscar	POP3	RTP	SIP	SMTP	Time Factor	
											Train	*Test*
AttributeSelectedClassifier	✓		✓			✓	✓	✓	✓	✓	27.8	2.8
Bagging	✓		✓			✓	✓	✓		✓	127.4	1.4
BayesNet	✓		✓			✓	✓	✓	✓	✓	40.0	42.5
DataNearBalancedND	✓		✓	✓	✓	✓	✓	✓	✓	✓	92.0	5.2
DecisionTable	✓		✓				✓	✓	✓		388.4	1.9
FilteredClassifier	✓	✓	✓			✓		✓	✓		21.9	2.1
J48	✓	✓	✓	✓		✓	✓	✓	✓	✓	37.7	2.1
J48graft	✓	✓	✓	✓		✓	✓	✓	✓	✓	49.6	3.0
NaiveBayes	✓		✓	✓			✓	✓	✓	✓	54.1	118.9
NaiveBayesUpdateable	✓		✓	✓			✓	✓	✓	✓	54.7	119.4
nestedDichotomies.ND	✓		✓	✓		✓	✓	✓	✓	✓	137.2	42.7
OneR	✓	✓	✓					✓	✓		5.0	1.0
PART	✓		✓	✓		✓	✓	✓	✓	✓	60.2	2.2
RandomCommittee	✓	✓	✓	✓		✓	✓	✓	✓	✓	42.3	2.3
RandomForest	✓		✓	✓		✓	✓	✓	✓	✓	35.0	2.3
RandomSubSpace	✓	✓	✓	✓		✓	✓	✓	✓	✓	71.6	1.9
RandomTree	✓		✓	✓	✓	✓	✓	✓	✓	✓	5.3	1.1
REPTree	✓		✓			✓	✓	✓		✓	13.7	1.3

Table IV: Binary coding of parameter combinations

Binary	Decimal	Parameter Class
2^0	1	packet count
2^1	2	interarrival
2^2	4	payload size
2^3	8	duration
2^4	16	bulk
2^5	32	idle

A. Classifier Based Parameter Reduction

The first basic approach to reduce the parameters is to evaluate all classifiers according to their classification accuracy with the different classifier combinations. Therefore we first filtered all parameter combinations for each classifier which have a classification accuracy greater than or equal to 90% on each of the protocols of Table III. Afterwards, we counted for each classifier, which of these parameter combinations were present on at least five protocols. The results of this evaluation are shown in the histograms of Fig. 4. For each of the 18 preselected classifiers one histogram is shown.

As you can see in the histograms of Fig. 4 nearly two-thirds of all possible parameter combinations — 39 out of 63 — have a classification accuracy of greater than or equal to 90% on at least five protocols with the used classifiers. Thus, one third of all parameter combinations is not reaching this limit and can already be dismissed for the further evaluation. Furthermore, on 32 of these 39 parameter combinations the parameter class payload size is included, which means that all possible parameter combinations with the parameter class payload size is present on at least one classifier. The other five parameter classes are only present in combinations with other parameter classes and not on their own like payload size (4).

Comparing the histograms in Fig. 4 with the results of Table II illustrates the low accuracy of the classifiers DecisionTable, OneR and filteredClassifier. In the histograms there are only fewer or no parameter combinations which have a classification accuracy greater than or equal to 90% on at least five protocols. Furthermore the best classifiers with the most classification accuracy greater than or equal to 90% in Table II are DataNearBalancedND, J48, J48graft, RandomCommittee, RandomSubSpace and RandomTree with nine out of ten protocols. But, the histograms in Fig. 4 show no parameter combination on these classifiers which can be used on all of the nine protocols. With the classifiers DataNearBalancedND, J48 and J48graft it is only possible to classify at the most seven protocols with one parameter combination to get a classification accuracy greater than or equal to 90%. For example, for DataNearBalancedND it is the parameter combination 36 (payload size and idle) and for J48 the two parameter combinations 12 (payload size and duration) and 44 (payload size, duration, idle). The classifier J48graft has six possible parameter combinations which can be used to classify seven protocols with a classification accuracy greater than or equal to 90%.

Like the classifiers RandomCommittee, RandomSubSpace and RandomTree, the classifiers PART and RandomForest have parameter combinations which can be used to classify eight different protocols with a classification accuracy greater than or equal to 90%. For all of these classifiers, the eight protocols which can be classified with one parameter combination having a classification accuracy greater than or equal to 90% are: Bittorrent, Flash, HTTP, Oscar, POP3,

 ISBN: 978-1-61208-201-1

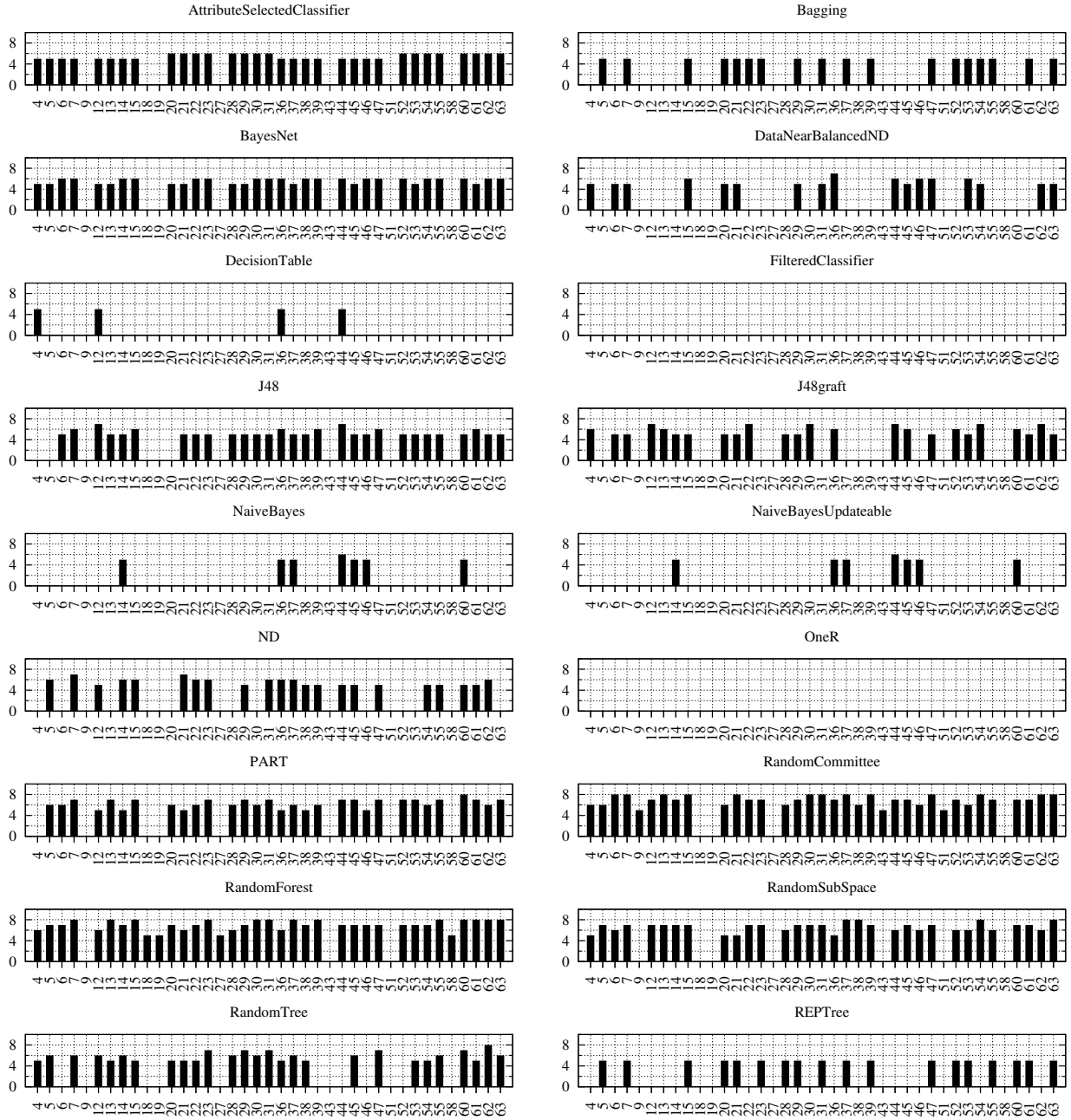


Figure 4: Distribution of parameter combinations for classifiers
(x-axis: parameter combinations, y-axis: number of occurrences)

RTP, SIP and SMTP. In fact, all of the possible parameter combinations contain the parameter class payload size. In addition, the most parameter combinations consist of at least three parameter classes. The only exception is the parameter combination 6 (interarrival and payload size) for the RandomCommittee classifier.

Fig. 5 shows a histogram with the number of classifiers on which each of the 39 parameter combinations has a classification accuracy of greater than or equal to 90% at least on five protocols. There are 16 parameter combinations which facilitate a wide spectrum — at least two-thirds of all classifiers (12 out of 18) — on the protocol classification: 7, 14, 15, 21, 29, 31, 36, 37, 44, 45, 47, 53, 54, 60, 61, 63. Also, these 16 parameter combinations contain the parameter class payload size and the most parameter combinations consist of at least three parameter classes — only exception is the parameter combination 36 (payload size and idle). In addition, the parameter combinations which are only present

 ISBN: 978-1-61208-201-1

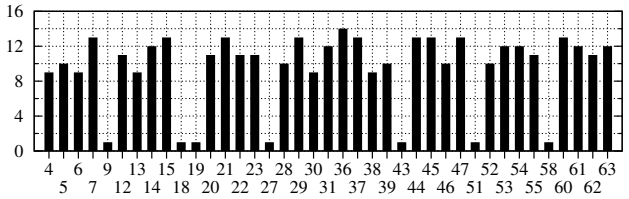


Figure 5: Number of parameter combinations for classifiers (x-axis: parameter combinations, y-axis: number of classifiers)

on one classifier (9, 18, 19, 27, 43, 51, 58) do not contain the parameter class payload size.

Finally, the results of these tests show that combinations of many or all parameter classes do not imply higher accuracy on the classification of the investigated protocols. But, for a good classification accuracy according to the classification of as many protocols as possible with one parameter combination, mostly combinations of three or more parameter classes are required.

B. Protocol Based Parameter Reduction

In contrast to the evaluation of all classifiers according to their classification accuracy with the different parameter combinations, the second approach to reduce the parameters for the classification is to evaluate the classification accuracy of the different parameter combinations depending on the network protocols. As in the first basic approach, we started with filtering all parameter combinations which have a classification accuracy greater than or equal to 90%. In the next step we counted for each protocol, which of these parameter combinations were present on at least nine different classifiers. The results of this evaluation are shown in the histograms of Fig. 6. For each of the ten protocols from Table III one histogram is shown.

The histograms of Fig. 6 include all 63 parameter combinations, but the protocol classes interarrival, duration, bulk and idle are mostly present in combination with other parameter classes and not on their own.

Table II shows some protocols (Bittorrent, Flash, POP3, RTP, SIP) which have a good classification accuracy with the most classifiers. The histograms of these protocols in Fig. 6 indicate this fact with a lot of possible parameter combinations having a classification accuracy of greater than or equal to 90% on at least nine classifiers. Furthermore, there are parameter combinations of the protocols Bittorrent, Flash and RTP with a classification accuracy of greater than or equal to 90% on all 18 classifiers. For Bittorrent these are 16 (bulk) and 48 (bulk and idle), for Flash these are 29 (packet count, payload size, duration, bulk) and 53 (packet count, payload size, bulk, idle) and for RTP these are 1 (packet count), 32 (idle) and 33 (packet count, idle).

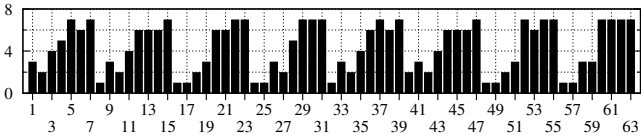


Figure 7: Number of parameter combinations for protocols (x-axis: parameter combinations, y-axis: number of protocols)

Fig. 7 shows a histogram with the number of protocols on which each of the parameter combinations has a classification accuracy of greater than or equal to 90% on at least nine classifiers. There are 18 parameter combinations which facilitate a wide spectrum — at least two-thirds of all protocols (7 out of 10) — on the protocol classification: 5, 7, 15, 22, 23 ,29, 30, 31, 37, 39, 47, 52, 54, 55, 60, 61, 62, 63. Also, these 18 parameter combinations contain the parameter class payload size and the most parameter combinations consist of at least three parameter classes — only exception is the parameter combination 5 (packet count and payload size). In addition, the parameter combinations which are only present on one protocol (8, 16, 17, 24, 25, 32, 48, 49, 56, 57) do not contain the parameter class payload size.

Additionally, the results of these tests show that combinations of many or all parameter classes do not imply higher accuracy on the classification of the investigated protocols. But, for a good classification accuracy according to the classification of a protocol with as many classifiers as possible with one parameter combination, mostly combinations of three or more parameter classes are required.

VII. Conclusion and Further Work

In this paper, we have determined how accurately 18 machine learning algorithms can be used to classify network traffic with 63 combinations of six statistical parameter classes consisting of 40 parameters. In contrast to many other studies, we used a per-packet approach instead of the per-flow approach.

As a result, we can say that the parameter class payload size has a significant influence on the classification accuracy. The four algorithms BayesNet, NaiveBayes, NaiveBayesUpdateble and ND have a time factor for testing, which is between 40 and 120 times greater than the fastest algorithm OneR. Thus these algorithms appear to be unsuitable for real-time traffic classification on high data rate links. The other algorithms have a time factor with nearly the same dimension. The algorithms DesicionTable and OneR reach only a low accuracy with the used parameters. The DataNearBalancedND, J48, J48graft, RandomCommittee, RandomSubSpace, RandomTree algorithms classified the most network protocols with an accuracy of 90% or more. The algorithms PART, RandomCommittee, RandomForest,

 ISBN: 978-1-61208-201-1

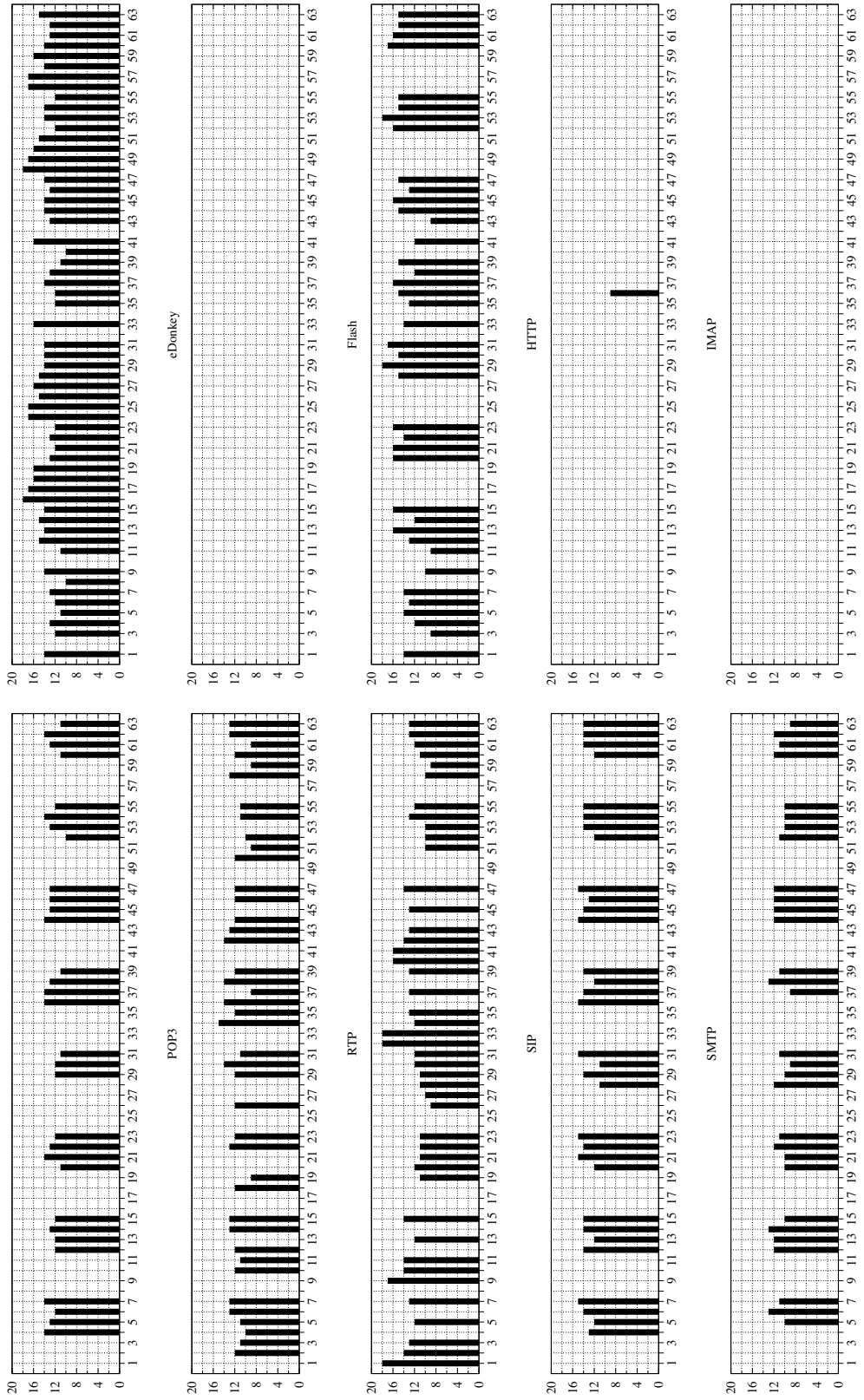


Figure 6: Distribution of parameter combinations for protocols
(x-axis: parameter combinations, y-axis: number of occurrences)

 ISBN: 978-1-61208-201-1

RandomSubSpace and RandomTree could classify eight of ten protocols with one parameter combination with an accuracy of 90% or more. Also the results show also that many parameters do not automatically cause a better classification. Furthermore, parameter reduction can improve the performance of per-packet classification which is necessary for the real-time traffic analysis of "fast" networks. A classification accuracy of 90% and above is a good result in comparison to other papers [3], [4], [5], [6], [15] where the classification accuracy often falls below 90%.

As a main result, we can say that the parameter combination needed for good classification results is depending on the machine learning algorithm used and the protocols to classify. There is no parameter combination, which can be used on all classifiers to get a high classification accuracy on all protocols. This implies that the results of the other studies [3], [4], [5], [6] can only be used with the same algorithm used in the particular paper.

Further Work

Other studies (e. g., [11]) showed that classification accuracy is decreasing by tests on network traffic of different locations. So we have to evaluate our results with other traces. To enhance the accuracy of traffic classification we have to find other parameters. All common statistical parameters were determined in this paper. New parameters should represent more characteristic properties of network protocols and their payload.

Another way to enhance the accuracy could be to split the protocols into those using TCP and those using UDP. Furthermore the parameter classes we are using consist of different parameters that represent information about upload, download and the whole flow. We will determine the benefit of this distinction, because possibly some of them can be dismissed.

Another field of research is the real-time classification of network traffic with machine learning algorithms.

Furthermore, we want to investigate if it is possible with machine learning algorithms to train the algorithms for different protocol classes like e-mail, bulk data transfer, P2P, interactive, gaming or multimedia to be able to classify unknown protocols which belong to a protocol class.

Finally, it would be interesting to study the best suitable algorithms in detail — instead of using them as a black box model — in order to improve them for better classification, to find out why they are more suitable than the others, and to find systematics which explains why there are some algorithms which are more suitable than others.

REFERENCES

[1] "Network Based Application Recognition (NBAR)," Cisco®, Oct. 2011, http://www.cisco.com/en/US/products/ps6616/products_ios_protocol_group_home.html.

[2] "Application Layer Packet Classifier for Linux," Oct. 2011, http://l7-filter.sourceforge.net.

[3] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, "Traffic classification on the fly," vol. 36. New York, NY, USA: ACM, April 2006, pp. 23–26. [Online]. Available: http://doi.acm.org/10.1145/1129582.1129589

[4] H. Jiang, A. W. Moore, Z. Ge, S. Jin, and J. Wang, "Lightweight application classification for network management," in *Proceedings of the 2007 SIGCOMM workshop on Internet network management*, ser. INM '07. New York, NY, USA: ACM, 2007, pp. 299–304. [Online]. Available: http://doi.acm.org/10.1145/1321753.1321771

[5] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in *Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, ser. SIGMETRICS '05. New York, NY, USA: ACM, 2005, pp. 50–60. [Online]. Available: http://doi.acm.org/10.1145/1064212.1064220

[6] S. Zander, T. Nguyen, and G. Armitage, "Automated traffic classification and application identification using machine learning," in *Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on*, Sydney, NSW, nov. 2005, pp. 250 –257.

[7] "TCPDUMP & LibPCAP," http://www.tcpdump.org.

[8] M. Canini, W. Li, and A. W. Moore, "GTVS: boosting the collection of application traffic ground truth," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-748, Apr. 2009. [Online]. Available: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-748.pdf

[9] "OpenDPI.org," http://www.opendpi.org.

[10] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA Data Mining Software: An Update," *SIGKDD Explorations*, vol. 11, no. 1, 2009.

[11] A. W. Moore, M. L. Crogan, and D. Zuev, "Discriminators for use in flow-based classification," Queen Mary University of London, Tech. Rep., 2005.

[12] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE In Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[13] "OpenPacket.org," https://openpacket.org.

[14] "Wiki site for the Wireshark network protocol analyzer," http://wiki.wireshark.org/SampleCaptures.

[15] P. Piskac and J. Novotny, "Using of time characteristics in data flow for traffic classification," in *Proceedings of the 5th international conference on Autonomous infrastructure, management, and security: managing the dynamics of networks and services*, ser. AIMS'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 173–176. [Online]. Available: http://dl.acm.org/citation.cfm?id=2022216.2022243

 ISBN: 978-1-61208-201-1

A Light-weighted Source Address Validation Method in IPv4/IPv6 Translation

Yu Zhu, Jun Bi and Yayuan Sun
Network Research Center, Tsinghua University,
Department of Computer Science, Tsinghua University,
Tsinghua National Laboratory for Information Science and Technology (TNList),
Beijing, China
zhuyu05@mails.tsinghua.edu.cn, junbi@tsinghua.edu.cn, sunyayuan@cernet.edu.cn

***Abstract*— since global IPv4 address has already exhausted in 2011, IPv6 is going to be deployed more widely in the next years. Both IPv4 and IPv6 would coexist in Internet for many years. Some transition technologies can help IPv4 to work with IPv6, but most of them are vulnerable to IP address spoofing attack. This paper proposes a source address validation method which works with IPv4/IPv6 translation. Only one change is required in DNS translation, based on current translation technology. Currently, an IPv4 server's address in DNS reply would be translated to an IPv4-mapped IPv6 address by DNS translator. In this paper, we proposed a method called "gateway identify code" (GIC) that the translator gateway embeds authentication information in IPv4-mapped IPv6 address in translated DNS reply. A host who receives this DNS reply would use this GIC embedded address to start communication. When packets reach translator gateway, validation is performed to check whether the GIC is correct. This technology can work with both stateful translation method and stateless translation method, including NAT-PT, NAT64 and IVI. This method will protect the address pool and filter the IP address spoofing attack.**

Keywords- IPv4/IPv6 translation; Anti-spoofing; Source address validation; Packet filtering; Internet Security; Access control.

I. INTRODUCTION

A. IPv4 address exhaustion and IPv6 development

In Feb 2011, ICANN (The Internet Corporation for Assigned Names and Numbers) declared that it had handed over the last IPv4 blocks to RIRs (Regional Internet Register). This handoff means the global IPv4 Internet has run out of addresses. There are not many addresses in RIRs address pool too. In April 2011, APNIC (Asia-Pacific Network Information Centre) reached the last /8 block of IPv4 address [1]. Asia Pacific is going to be the first region unable to meet the IPv4 Address demand due to the unprecedented fixed and mobile network growth in this area. After the exhaustion of IPv4 addresses, there would be no new address for new device such as cellphones, wireless sensors to visit Internet. IPv4 address exhaustion may cause the growth of Internet to speed down.

Noticing IPv4 cannot provide enough addresses to meet the requirement of the development of Internet, IETF proposed IPv6 in 1990. IPv6 has 128bit address space which could provide far more address than current IPv4 network. In October 2010, 243 (83%) of the 294 top-level domains in the Internet supported IPv6 to access their domain name servers, and 203 (69%) zones contained IPv6 glue records, and approximately 1.4 million domains (1%) had IPv6 address records in their zones [2]. However, IPv6 address family is not compatible with IPv4. An IPv6 host cannot directly connect to an IPv4 host. Moving billions of users and millions of sites from IPv4 to IPv6 will cost many years. A long period is needed for IPv4 users to transit to IPv6. Two categories of transition methods have been proposed to provide a way to connect IPv4 and IPv6 network, namely, translation and tunnel. Both of them are important scenarios in IPv6 deployment.

B. IP spoofing attack in IPv6/IPv4 translation

IP address spoofing attack uses forged source address in packets. Since the address is forged, the reply of the attack packet will not reach the attacker. This kind of attack is widely used in a DoS (Denial of Service) attack [3]. Although many ISPs have enforced spoofing prevention functions on their network infrastructures over these years, no mitigation improvement has been achieved over four years [4]. The MIT ANA Spoofer project [5] shows that 17.2% of the IP addresses, 15.2% of the net blocks and 24.4% of the ASes are still spoofable across Internet.

Many kinds of anti-spoofing methods have been proposed by researchers, including SAVI [6], uRPF [7], Packet Passports [8], DPF [9], SPM [10], etc. SAVI switches sniffer packets of IP address allocation protocol, uses information in the packet to create binding entry on the switch. However, it needs to replace all access switches change access switch to enable SAVI function. uRPF reversely utilizes the forwarding table on routers to check packets' source addresses, but may drop valid packets in case of asymmetric routing. Packet Passports inserts "AS passports" which are checked by the ASes along the packets' paths into packets. It fails when packets are delivered through different AS paths due to routing dynamics. DPF is a distributed route-based packet filtering framework. With partial deployment, DPF can decrease spoofing packets significantly. The major omission from the DPF research is the method for routers to learn the incoming direction information, which is very critical and hard in practice. SPM is an AS-level source address validation system. A unique temporal key is associated with each ordered pair of source destination ASes in SPM. Edge routers will add the key to outbound packets and verify the key in the inbound packets.

 ISBN: 978-1-61208-201-1

All of these anti-spoofing methods are not designed for the IPv6/IPv4 transition scenario. So a new method is needed to apply anti-spoofing in IPv6/IPv4 transition.

Address spoofing attack is a tremendous threat to stateful IPv6/IPv4 translation method. In stateful translation method, translator allocates temporary IPv4 address from address pool for IPv6 hosts. The address pool is an important resource of translator gateway. Attacker can exhaust the address pool by forged address attack. Each packet with a different source address can obtain an address from the address pool. One reason that NAT-PT [11] is moved to history status is "Creation of a DoS (Denial of Service) threat relating to exhaustion of memory and address/port pool resources on the translator."[12].

This paper proposes a novel anti-spoofing method "Gateway embed and verify" (GEAV) method to prevent IP spoofing attack in IPv6/IPv4 translation. DNS translator will translate an IPv4 address to an IPv6 address in DNS records. In GEAV, the translator gateway sends "gateway identify code" information to hosts in the IPv4-mapped address in DNS reply packet. This special IPv4-mapped address will be carried by every packet a host sends out which needs translation. The translator gateway checks GIC in packets to apply anti-spoofing when packets reach. GEAV does not need any host change. The detail of this anti-spoofing method will be discussed in next chapters.

II. GATEWAY EMBED AND VERIFY ANTI-SPOOFING METHOD IN IPV6/IPV4 TRANSLATION

A. *IPv4/IPv6 translation technologies*

Transition technologies could be mainly classified in two classes: tunnel and translation. Tunnel enables IPv6 (IPv4) communications to pass through IPv4 (IPv6) network, while translation enables communication between IPv6 and IPv4 hosts, as Figure 1 shows. Both technologies are used widely in IPv4/IPv6 transition.

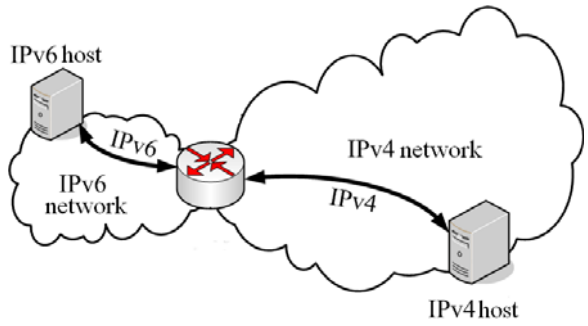


Figure 1. IPv6 / IPv4 translation.

Classified by working layers, translation technologies are sorted into three categories: application layer, transport layer and network layer translation. Network layer translation methods translate IP packets between different address families and typical solutions are NAT-PT, NAT64 [13] and IVI [14].

NAT-PT assigns a temporary IPv4 address for the IPv6 client and translates each packet on the translator gateway according to the mapping table of IPv6 client address and IPv4 temporary address. NAT64 conquers NAT-PT's flaw, separates the translate DNS with the gateway. IVI uses special IP address allocation strategy for hosts. As a stateless method, IVI do not save the mapping information of IPv4 and IPv6 addresses. IVI calculates IPv4 address/port for a host using its IPv6 address, replacing the mapping table with a global mapping rule.

Let us take NAT-PT as an example to see how IPv4/IPv6 translation works. Step1, an IPv6 host (client) wants to use the service provided by an IPv4 server, so it sends a DNS request to look for the server's address. Step2, An A record, which contains an IPv4 address, would be carried in the DNS reply to response the request. The translator gateway would hijack the DNS reply and transform the A record in the DNS reply into an AAAA record, which contains an IPv4-mapped IPv6 address. Figure 2 shows an A record of 1.2.3.4 is translated to a AAAA record with 2001::[1.2.3.4] by a translate device.

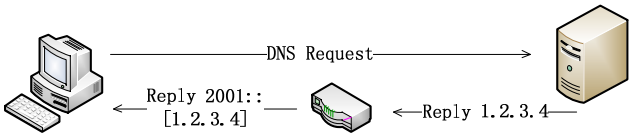


Figure 2. DNS interaction in NAT-PT translation.

Step3, after the DNS interaction, the IPv6 client will start to send packets to the address in AAAA record to acquire service. When the packet reaches the gateway, the translator gateway recognizes the packet's destination address represents an IPv4 host. The gateway allocates an IPv4 address from address pool for client, and translates the packet to IPv4 address family. When the server responses the request, the translator gateway translates server's reply and forward it to the IPv6 client. In Figure 3, the gateway allocates a temporary IPv4 address 4.3.2.1 for the IPv6 host 2001:da8:10::4. As Figure 3 shows, in the view point of the client, it communicate with host 2001:[1.2.3.4]. In the view point of the server, it has a IPv4 session with host 4.3.2.1 .

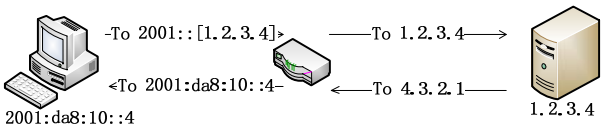


Figure 3. IPv6 client visit IPv4 server using translation.

In all these steps, the packets pass through translator gateway. The communication between the clients and translator gateway looks like a three-way handshake. In step2, if the translator gateway embeds some information in the DNS reply, this information would only be acknowledged by the owner of the source address of DNS request and the gateway. When there is an attack using spoofed address, if the owner of the source address of the

 ISBN: 978-1-61208-201-1

DNS request did not send the request, the reply would be discarded by the receiver. Unless the attacker uses Man-in-the-middle attack, he cannot receive the information embedded by the translator. The translator gateway embeds GIC (Gateway Identify Code) information to the DNS reply, and verify it when receive packets from hosts as shown in Figure 4. This method is called “Gateway embed and verify” method.

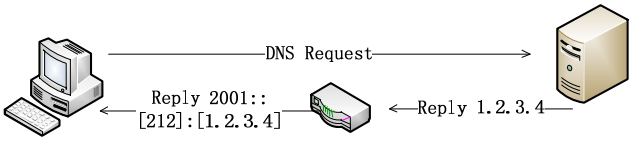


Figure 4. GIC embedded DNS reply.

To deal with man-in-the-middle attack, a proper method is encrypting all the packets which cost extra resources. This is not a light-weighted method that we are looking for.

B. Gateway embed and verify method

Without any DNS extension, the only information stored in DNS record andused by the sender is the address. In IPv4/ IPv6 translation, the IPv4 address after translation would be an IPv6 address with the original IPv4 address embedded. In NAT-PT address format, as shown in Figure 5, 64bit will be used for the prefix; 32bits will be used for the original IPv4 address; the rest 32bits are set to 0.

0 63	64	96	127
Local ISP prefix	Zero	IPv4 addr	

Figure 5. Normal IPv4-mapped address format.

These unused bits can be used to store information. An identify code named “gateway identify code” can take some bits. For example, as shown in Figure 6, 8 bits GIC is placed in 88-95bits.

0 63	64	88	96	127
Local ISP prefix	Zero	GIC	IPv4 addr	

Figure 6. Improved IPv4-mapped address format.

When a packet arrives at the translator gateway, the embedded GIC would be checked. Gateway forwards a packet if the GIC is correct, or discards it if it is wrong, as shown in Figure 7. If the GIC in the packet is 212, it would be translated. If not, it will be blocked.

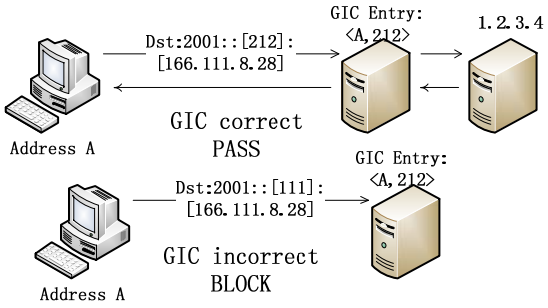


Figure 7. Gateway validation with GIC.

The translator gateway keeps a table for hosts who sent out DNS requests. Gateway maintains the table that stores the IPv6 source addresses of the DNS requests and GICs for verifying. The GIC entry has a very short lifetime. Typically, it is 5 minutes, assuming a host start to send a request to the server address after it receives the address in 5 minutes. When a client start to use the address to connect to an IPv4 server and an allocation of temporary address happens in the address pool, this GIC entry changes into a static entry until the resource is reclaimed.

The improved process of a session using IPv4/IPv6 translation would look like this.

- Step1, IPv6 client sends request to an DNS server.
- Step2, translator gateway receives the A record of IPv4 server.
- Step3, translator gateway tries to find the source address of the DNS request in the GIC table. If this entry could be found, then use the GIC in this entry. If it is not found, then a new entry would be created, add assign a GIC for this host.
- Step4, translator gateway sends the GIC-embedded IPv4-mapped address in AAAA record to the IPv6 client.

The GIC entry will be inactive and be removed when the IPv4 translation address is reclaimed by the address pool.

C. Infomatin sharing issue between gateway and DNS

In some translation method, DNS-ALG [15] will provide translation of DNS replies. In the previous part of this chapter, we assume that it is the gateway device that sends the IPv4-mapped DNS reply to the host. If a DNS reply a host receives was sent out by an independent DNS server, the assumption that GIC information is shared only by the gateway and the host is violated. Two solutions can be used in this scenario.

The first is to launch a communication between DNS and gateway to share information with each other. By sharing information, DNS and gateway could be treated as one device. In the next chapters, we would treat the translate DNS and the translator gateway as one gateway.

The second is to deploy the independent DNS-ALG server out of the translator gateway. That means every message from the DNS to the host would pass the gateway. The DNS sends out IPv4-mapped IPv6 address in DNS reply,

 ISBN: 978-1-61208-201-1

the gateway hijacks the DNS reply and embeds GIC information in it.

III. CORRECTNESS ANALYSIS

A. *Correctness in stateful method*

Now check some scenario in this system when an address spoofing attack happens.

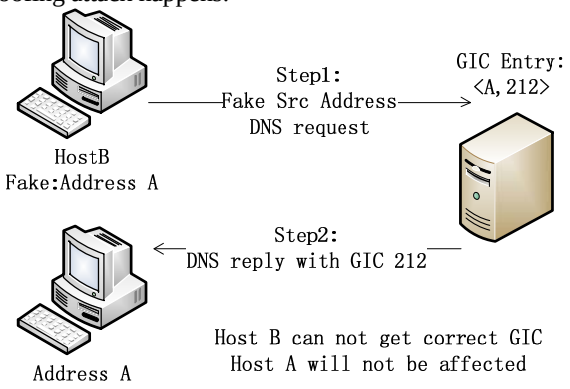


Figure 8. Scenario1

Scenario1, as shown in Figure 8, IPv6 host A keeps a session with an IPv4 server, and has already gotten an IPv4 address from the address pool. Host B tries to fake host A to start an attack. If host B sends a DNS request using host A's address, then the gateway would send to GIC to A but not B. If B skipped the DNS step and try to send a packet to an IPv4-mapped address directly without a correct GIC, this attack packet cannot pass the gateway. If an attacker tries to enumerate GIC, only a small part of attack stream would pass.

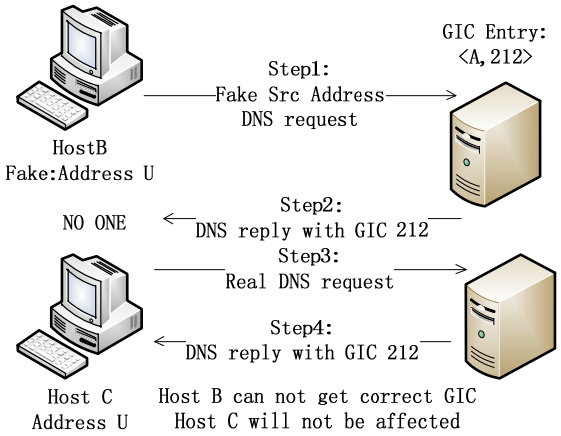


Figure 9. Scenarios 2&3

Scenario2, Host B wants to fake an unused IPv6 address U. B uses U as source address and starts to send packets to an IPv4-mapped address. The translator gateway finds no entry of U in GIC table when receives a packet using U as source address, and discard the packet. When host B sends a DNS request using U, the gateway would add an entry in GIC table, and send the GIC to the address U which cannot be picked by host B. Host B do not know the correct GIC because it cannot receive any packets sent to U unless Man-in-the-middle situation.

Scenario3 is a transition of Scenario2. Host B fakes a host with an unused IPv6 address U. Then the legal owner of U, called host C, start to use U. Host B may have sent some DNS request, so the gateway may have already got an entry of U in GIC table. When C sends DNS request, gateway uses the entry already exists in GIC table and sends the GIC to C. Any attack performed by B would not affect host C to use U to communicate to IPv4 servers. Figure 9 shows the GEAV method could work properly in scenarios 2 and 3.

From the analysis of the above scenarios, it can be concluded that a spoofing address attack will only form an entry in the GIC table. Most attacks would be filtered at the gateway. Only a small number of attacks can get out of the gateway when the attacker happens to guess correctly.

B. *Security issues of GIC table*

In GEAV, before a request gets resource from the address pool, a GIC entry will be set up for verification. If the verification fails, the allocation will not happen. But the GIC table might be a new target of DoS attack. An attacker might forge different addresses to form a lot of GIC entries. When gateway detect the size of GIC table is growing very fast, it can reduce the lifetime of a GIC entry to a very short time, for example 5 seconds. The speed rate of attack stream multiple the lifetime of GIC entry is the amount of forged entries in the GIC table. By reducing the lifetime of GIC entry, the pressure from DoS attacks to the GIC table could be reduced.

C. *Correctness in stateless method*

There is no address spool to protect in stateless method. Though this verifying method is stateful, it can also be used in stateless IPv6/IPv4 translate method to reduce the address spoofing attack. But additional verification cost is required.

The size of the GIC table would grow larger than that in stateful method. In stateful method, the GIC table is as large as the scale of the address pool. In stateless translation, there are no allocation in the address pool and no reclaim of address, so the size of GIC table would reach the size of the address space of the subnet since GIC for each address should be assigned in advance. This will cause terrible scalability problem. We will solve it in the next Chapter.

IV. SCALABILITY ANALYSIS

The protection is done on the translator gateway. The scale of the GIC table could be a problem as gateway needs to store the whole GIC table. Though the GIC entry lifetime could be extremely short in worst cases, the GIC table will grow to approximate the size of address space. Unlike the address pool, GIC table will be visited more frequently. Each packet using IPv6/IPv4 translation will visit GIC table. If the scalability problem is not handled properly, this mechanism is fragile.

 ISBN: 978-1-61208-201-1

A. *Scalability in stateful method*

There are many ways to solve this scalability problem. *First, multiple gateway devices could be used for load balance.* Each gateway would be in charge of a part of the prefixes of the IPv6 subnet and store the GIC table of its own part. Using multiple gateway devices and using load balance technology could enhance the support of a large number of users. By separating the hosts in prefixes, the scale of GIC of every device could be reduced.

Second, a GIC entry could be shared by a group of hosts. Hosts could be sorted into small groups first. We can change the GIC allocate strategy from allocate a GIC for each host to allocate a GIC for each group. When the first client in one group sends a DNS request, a new entry is created for this group. GIC of one group should change from time to time to avoid guess attacks. The gateway keeps the former GIC for a period of time after the GIC changes because many hosts are still using the former GIC. Some former GICs and the current GIC should be considered as correct when verifying. Use multiple GICs for one prefix may reduce the performance, but dynamic GIC could provide more protection than fixed GIC. Assuming a method of separating address space by prefix is used, a final version would look like this. When gateway allocate an IPv4 address from the address pool to an IPv6 host, the gateway stores the IPV6 address A6, the IPv4 address A4 and the GIC G. Gateway keeps a group-shared GIC usage table, which logs the count of GICs hosts in one group. If one IPv4 address is leased from translator gateway, a GIC is assigned for the host of this address. The count of according entry will plus 1. When A4 is reclaimed, the count will minus 1. If a former GIC entry has a count of zero, this entry will be deleted because no host is using this GIC.

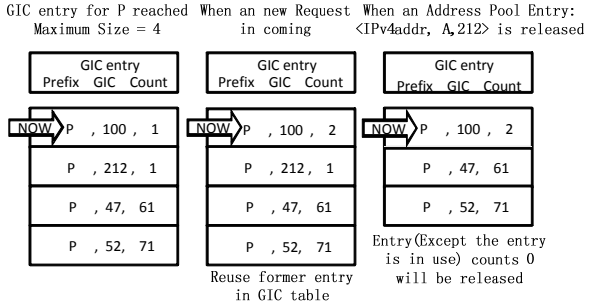


Figure 10. Prefix-shared GIC using count with reuse strategy

Below are the choices to cope with the case when the GIC set for this prefix has grown to the maximum size:

1. *Stop changing GIC temporarily.* This may cause the changing of GIC stops. But it may increase the risk of guess attacks.

2. *Delete the oldest item.* This method keeps the GIC changing, but may block connections which established in the past. For example, if the lifetime of GIC is 2days and a set contains 8 GICs, connections established 16 days ago maybe interrupted.

3. *Reuse the GIC in the oldest item for new requests.* The number of GIC in every set and the lifetime of every GIC could be adjusted to keep GIC changes for better filtering performance and sufficient security. Figure 10 shows this process.

B. *Scalability in stateless method*

In stateless translation, there are no allocation and reclamation of address in the address pool. One possible solution could be like this:

Set up a GIC table for the every prefix after separating. For each specific prefix, a list of GICs is maintained. The GICs in the list change periodically. Once a GIC is changed, the new one would be added to the end of the list. The new GIC is applied as soon as it is created. A variable indicates the visit number of the GIC in last small period would be added into the GIC entry. If one entry's visit number is zero, we may assume no one is using this GIC anymore. Then this entry would be removed from the table, as shown in Figure 11. The GIC entry of 100 is removed as the visit number is decreased to zero, but the entry of 52 is kept as it is a new entry just in use.

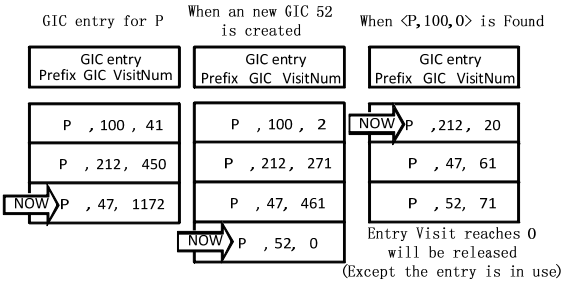


Figure 11. Prefix-shared GIC using visitnum

Deleting an entry with a non-zero visit number will pose risk to the system. If this list reaches the maximum length, we could use the reuse old-item method to maintain as shown in Figure 12. The old GIC entry of 100 is reused as the list reaches maximum length of 4. In this case, most users' interest will be protected. Special policy could be applied to delete old entries.

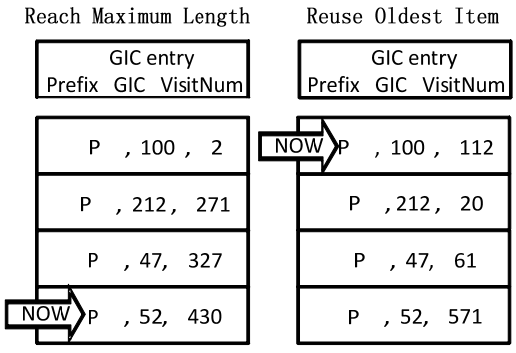


Figure 12. Prefix-shared GIC using visitnum with reuse strategy

 ISBN: 978-1-61208-201-1

C. Using real-time calculating in stateless method

When a packet reaches the gateway, the gateway should recognize the correct GIC. The gateway should have the mapping information of source address and GIC. In the previous part of this paper, we use a table to save GIC for verifying. In this part, we will introduce another approach.

A real-time calculating method could be used to replace GIC table. For security reasons, GIC should be private. For different addresses, they should have different GICs. So source address should be an input of the algorithm. For each incoming packet, the gateway uses its source address to calculate a GIC and compare with the one carried in the packet. To prevent guess attacks, the GIC should be dynamic. The algorithm needs a dynamic input besides address. This dynamic input for generating algorithm is called "seed", which should be unique and used for all hosts in the gateway. A set of seeds would be kept at the same time as multiple GIC is used.

Since the seed is unique, the safety of seed should be considered. The algorithm is usually public, if a seed can be inferred by one's address and GIC, the safety of this mechanism is challenged. So the generating algorithm should be irreversible.

Now this process would look like this: GIC is calculated with an input of seed and the address of source. GIC sent to the client in a DNS reply. When a packet comes from the client reaches the gateway, it calculates the GIC for this packet and check to see if they are equal. Each packet would trigger one calculating. The algorithm is the critical part of this method. It should be irreversible and fast. Comparing to store the whole GIC table, using real-time calculating is still an alternative choice when the scalability becomes a critical problem.

V. PERFORMANCE ANALYSIS

There are many parameters used in this mechanism, which includes the number of bits used in IPv4-mapped address N, the number of gateway devices used for load balance G.

Block Number (BN) indicates the number of small blocks which the whole address space has been separated to. Lifetime (LT) of GIC entry in stateful method would be from the temporary IPv4 is allocated from the address pool to the resource is reclaimed by the gateway. The number of entries in the block-GIC table is L.

Now, we will define key measurement of this method:

Protection: the percentage of address spoofing packets can affect the victim. Smaller value represents a better effect.

Lookup cost: the cost depends on the lookup algorithm and the number of GIC entry. Smaller value represents a better effect.

Minimum long connection time: the time a connection would possibly be reset by the gateway since it established. Bigger value represents a better effect.

Assuming the gateway has an IPv6 address space size: S_6 addresses. And an IPv4 address pool size: S_4 addresses.

Firstly, we will analysis how the bits that GIC take in IPv4-mapped address would influence the performance. There are 32 unused bits in IPv4-mapped address in total. When an attacker uses brutal force, it enumerates all 2^N combinations in GIC field. There are only L valid GICs at one time. The protection could be enhanced by more bits used for GICs. But the bits may be limited by longer prefix.

The GIC table size in stateful method is the size of IPv4 address space. In stateless method, it equals to the size of IPv6 address space. Both of them could be reduced by separating the address space to small blocks.

The GIC changes periodically. One GIC is valid until it is removed from the list. As the length of list is L, the minimum long connection time would be LT*T. If the reuse strategy in last section is applied, no connection would be interrupted by the GIC method. Table I and Table II show the performance of GEAV method in different scenarios.

TABLE I. PERFORMACE IN STATEFUL METHOD

	Stateful GIC	Stateful with separating blocks	NO GEAV
Protection	$1-1/2^N$	$1-L/2^N$	0
GIC entry number	S_4	S_4 /BN	N/A
Min long conn time	∞	LT*L (∞ if entry reuses)	∞

TABLE II. PERFORMACE IN STATELESS METHOD

	Stateless GIC	Stateless with separating blocks	NO GEAV
Protection	$1-1/2^N$	$1-L/2^N$	0
GIC entry number	S_6	S_6 /BN	N/A
Min long conn time	LT*L	LT*L (∞ if entry reuses)	∞

A typical campus network has an IPv4 address space of about one /16 and an IPv6 address space of some /48. This means the address pool size in IPv6/IPv4 translation would between tens of thousands to hundreds of thousands. The GIC table entry number would be around 100,000. The GIC table would take 100,000*(128+32+N*L). Setting N to 16 and L to 8, the GIC table would take 36MByte storage. If 100,000 entries lookup is a too heavy load for gateway devices, the IPv4 addresses pool could be spilt into small pools. Using 5 gateways, only about 20,000 entries would be loaded on one device. A /48 IPv6 prefix could be break into /60 blocks, only around 4,000 entries are needed. The deployment of GIC system would filter 65528/65536 attack packets, which is a significant benefit for the network security and performance.

We have a prototypical system in development and tested it with Tsinghua University IVI experimental deployment. This environment has an IPv4 prefix of 58.200.228.0/24, and

 ISBN: 978-1-61208-201-1

an IPv6 prefix of 2001:da8:ff3a:c8e4::/64 . There are one IVI router and 250 IVI-enabled hosts in this experimental subnet. Our GEAV device can support 64K GIC entries and bi-directional 10Gbps traffic in tests. After the development is finished, we will try to test it in real network.

VI. CONCLUSION AND FUTURE WORK

Applying with IPv4/IPv6 translation, "gateway embed and verify" (GEAV) use translated DNS reply to carry information to hosts. To map an IPv4 address to an IPv6 address, there are many unused bits in IPv4-mapped address. GEAV method uses these blank bits in the IPv4-mapped address of the DNS record to pass GIC information to client. The client uses this GIC-embedded IPv6 address for communication. The GIC will be carried in the destination address of all packets. Verification is performed on the translator gateway when a packet needs translation reaches. By implementing function on translator gateway and DNS translator, no change in host is required. GEAV is not transparent to hosts, but it will not affect communications between hosts. The risk of address pool being exhausted would be reduced by using this technology. Applying this technology with stateless translation method can also help reduce a lot of address spoofing packets. In the paper, we analyses the correctness of GEAV method, and solve the scalability problem when applied to both stateless and stateful translation method.

But, GEAV method still needs a lot of improvement. Unfortunately, this method can only work with DNS-based IPv4/IPv6 translation for now. GEAV method cannot provide protection when facing middle-man attack. Until now the tests are done in experimental network, which only has a small number of hosts. We are still improving the prototype system and try to deploy it in the campus network which has more users than the experimental subnet. Also, the algorithm, which can be used in the real time calculation, needs future discussion. An irreversible hash algorithm is needed, and there is a tradeoff between performance and security.

VII. ACKNOWLEDGEMENT

This paper is supported by National Science Foundation of China under Grant 61073172, Program for New Century Excellent Talents in University, and Specialized Research Fund for the Doctoral Program of Higher Education of China under Grant 20090002110026.

REFERENCES

[1] http://www.apnic.net/__data/assets/pdf_file/0018/33246/Key-Turning-Point-in-Asia-Pacific-IPv4-Exhaustion_English.pdf [retrieved: May, 2012]

[2] M. Leber, "Global IPv6 Deployment Progress Report". Hurricane Electric. 2010. http://bgp.he.net/ipv6-progress-report.cgi [retrieved: May, 2012]

[3] C. Labovitz, "Botnets, DDoS and Ground-Truth -- A Look at 5,000 Confirmed Attacks", NANOG 50, October, 2010.

[4] R. Beverly, A. Berger, Y. Hyun, and K. Claffy, "Understanding the efficacy of deployed internet source address validation filtering", Proc. 9th ACM SIGCOMM conference on Internet measurement conference, pp. 356–369, 2009.

[5] The MIT ANA Spoofer Project, http://spoofer.csail.mit.edu/ [retrieved: May, 2012]

[6] J. Bi, G. Yao, J. Wu and F. Baker, "SAVI Solution for DHCPv4/v6", draft-bi-savi-cps-02.txt, October, 2009

[7] Unicast Reverse Path Forwarding, Cisco IOS, http://www.csm.ornl.gov/~dunigan/oci/uni_rpf.pdf. [retrieved: May, 2012]

[8] X. Liu, X. Yang, D. Wetherall, and T. Anderson, "Efficient and secure source authentication with packet passports", Proc. of SRUTI '06, pp. 2-2, July, 2006.

[9] K. Park, and H. Lee, "On the effectiveness of Route-Based packet filtering for distributed DoS attack prevention in Power-Law Internets", Proc. 2001 SIGCOMM, volume 31, issue 4, pp. 15-26, Oct 2001.

[10] A. B. Barr, and H. Levy, "Spoofing prevention method". Proc. IEEE INFOCOM, 2005.

[11] G. Tsirtsis, P. Srisuresh, "Network Address Translation - Protocol Translation (NAT-PT)", RFC 2766, February, 2000

[12] C. Aoun, E. Davies. "Reasons to Move the Network Address Translator - Protocol Translator (NAT-PT) to Historic Status", RFC4966, Jul.2007.

[13] M. Bagnulo ,P. Matthews and I. van Beijnum," Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", April 2011.

[14] X. Li. C. Bao,F. Baker and K. Yin," IVI Update to SIIT and NAT-PT draft-baker-behave-ivi-01", September 16, 2008.

[15] P. Srisuresh, G. Tsirtsis and P. Akkiraju,A. Heffernan, "DNS extensions to Network Address Translators (DNS_ALG)", September 1999

 ISBN: 978-1-61208-201-1

Formal Treatment of Distributed Trust in Electronic Voting

Stephan Neumann and Melanie Volkamer
CASED / TU Darmstadt
Hochschulstraße 10
64289 Darmstadt, Germany
Name.Surname@cased.de

***Abstract*—Electronic voting systems are among the most security critical distributed systems. Different trust concepts are implemented to mitigate the risk of conspiracies endangering security properties. These concepts render systems often very complex and end users no longer recognize whom they need to trust. Correspondingly, specific trust considerations are necessary to support users. Recently, resilience terms have been proposed in order to express, which entities can violate the addressed security properties in particular by illegal collaborations. However, previous works derived these resilience terms manually. Thus, successful attacks can be missed. Based on this approach, we propose a framework to formally and automatically derive these terms. Our framework comprises a knowledge calculus, which allows us to model knowledge and reason about knowledge of collaborating election entities. The introduced framework is applied to deduce previously manually derived resilience terms of three remote electronic voting systems, namely Polyas, Helios and the Estonian voting system. Thereby, we were able to discover mistakes in previous derivations.**

***Keywords*-trust, distributed systems, formal methods, resilience, electronic voting, knowledge calculus, conspiracies**

I. Introduction

Recently, the interest in electronic voting systems increases as more and more states implement electronic voting, both with voting machines as well as remote internet voting schemes. In this paper, we only consider internet voting schemes and use the term electronic voting or eVoting interchangeably. Electronic voting schemes are complex distributed systems with particularly strict security requirements due to the nature of elections. It is therefore of great importance to evaluate these schemes prior to their use in legally binding elections.

Numerous analysis and verification techniques for electronic voting have been proposed over the past decades. The Common Criteria, in particular the Protection Profile for electronic voting [1], is an international standard for security evaluation, which has been successfully applied to electronic voting schemes, see [2] for an example. Additionally, many researchers evaluated proposed voting schemes, both with formal methods as in [3], [4], [5] and by cryptographic means as in [6], [7]. However, these techniques mainly investigate external attacks and do not address illegal collaborations between different entities.

However, specific trust considerations are necessary because the implemented trust concepts result in very complex systems and voters are faced with the problem whom to trust not to illegally collaborate with other entities. In [8] Volkamer et al. propose resilience terms to derive which entities need to be in particular trusted not to collaborate maliciously in order to ensure security properties. Thus, resilience terms express how robust a system is against conspiracies of entities that do not behave properly. Resilience terms have shown their benefit in the evaluation of different systems, while their derivation remains informal and potential conspiracies can be missed or misinterpreted, which leads to wrong resilience terms.

In this paper, we review resilience terms and introduce a logic, which allows us to formally and automatically derive resilience terms in electronic voting schemes. While there are many security properties that need to be satisfied by voting schemes, we focus on secrecy as property of crucial importance to most voting schemes. Our idea is based on formal methods and knowledge management. We establish distributed knowledge bases of entities in logical terms and propose a inference system, which incorporates the deduction rules to extend the obtained knowledge by the adversary. We apply our framework for three electronic voting schemes, namely Polyas, Helios and the Estonian one; compare the results with those from [8]; and thereby detect mistakes in the previously established resilience terms.

The remainder of this paper is organized as follows: In Section II, we review related work in the formal analysis of electronic voting schemes. In Section III we review the existing framework on resilience terms as proposed in [8]. In Section IV, we introduce our knowledge calculus, while Section V is dedicated to the formalization of the secrecy property. Thereafter, in Section VI we exploit the proposed knowledge calculus in order to derive resilience terms. In Section VII, we deduce the resilience term for the three electronic voting schemes based on our proposal. Section VIII concludes this paper and shows future directions.

II. Related Work

The formal security evaluation of electronic voting schemes has been approached by different means. In this section, we review related literature and check whether these

 ISBN: 978-1-61208-201-1

works can be adapted to our own needs. As this paper addresses only internet voting schemes, we do not take into account the verification of voting machines as proposed for instance in [9].

Several works build upon the applied pi calculus [10] and its formalization in ProVerif: Backes et al. [3] prove coercion-resistance of the JCJ protocol. Kremer et al. [4] prove fairness and eligibility of the FOO 92 [5] protocol and Cortier et al. [11] adapt the Helios system and prove secrecy of the system. These techniques assume (or encode this in the properties specification) a fixed scenario, where a given set of entities is assumed to be trusted not to collaborate maliciously. The goal of these works is to identify scenarios in which a security property can be verified rather than determining the smallest set of entities that might assure that property. Therefore, a derivation of resilience terms with respect to any illegal conspiracies can not be handled by these techniques without further adaptation.

Jonker et al. [12] propose a formalization of receipt-freeness in a state-orientated manner, which allows them to verify receipt-freeness for different electronic voting schemes. Their formalization is strongly generic, i.e., term structures underlying the schemes need to be generated independently for protocols. In [13] and [14] Bräunlich et al. follow the idea of Jonker and use the state-orientated model to identify state transitions not violating state invariants. Their main purpose is to support engineers in the design of protocols in a way that state invariants hold. Due to the abstract nature of their approach, it is currently not applicable for voting scheme evaluations.

Finally, we will review whether approaches proposed in the context of formal trust management can be adapted to our needs. Trust concepts and management systems have been approached from different directions; in [15] and [16], the authors rely on game-theoretic approaches to evaluate the cost-benefit relation for different entities to collaborate. We see the value of these approaches mainly in the evaluation of resilience terms. Once resilience terms have been determined, the risk or chance of entities ensuring or violating security properties can be estimated. We refer to [17] and [18] for a comprehensive overview on trust concepts.

III. Existing Framework

Electronic voting systems process sensitive data, so different trust concepts were proposed in order to mitigate the risk of conspiracies endangering security properties such as integrity or secrecy. There are mainly three trust concepts applied to electronic voting, namely separation of duty, the four eyes principle and the multiple execution of a duty. These trust concepts are usually combined and applied multiple times. This leads to complex trust distributions where the question whom to trust not to collaborate maliciously regarding certain security properties can become hard to answer.

The evaluation of distributed systems with resilience terms has been introduced in [19] and adapted to electronic voting in [8]. Resilience terms allow one to identify which entities a voter has to trust - in particular not to collaborate maliciously - in order not to violate an investigated security property. Terms however need to be determined independently for different properties. Entities can be voting servers, administration staff, developers and key holders. The framework assumes voters to behave properly and, thus, voters are not considered as entities in this framework[1]. Resilience terms can be derived on different levels, where level 1 corresponds to servers or key holders, level 2 to the local position of components and administration staff of servers correspondingly, and level 3 to the manufacturer of the voting software run on the servers. Terms of higher levels can be (formally) derived based on lower level terms in a straight-forward manner.

The framework of resilience terms has been applied to derive resilience terms for different voting systems, namely the Estonian voting system, the Polyas [20] and the Helios 2.0 [21] system. The framework has been extended in [22] towards post-processing of these terms by means of transformation into logical terms and the evaluation of terms with respect to trust metrics. It allows one to determine the probability with which an electronic voting system fulfills a security property. Figure 1(a) shows how resilience terms can be evaluated with respect to trust metrics. In order to determine the resilience term, in [8] the authors propose to manually and informally determine the knowledge of entities once the voting phase has terminated. These local knowledge sets of entities are thereby determined from a *worst-case* point of view, i.e., entities are assumed not to behave properly in the sense that they store all terms they obtain and furthermore they store all visible relations between terms. Based on the obtained local knowledge sets, attack scenarios are identified and the resilience terms are derived correspondingly. Resilience terms have the form

$$t = (k_1 + \cdots + k_m) \text{ out of } (N_1, \ldots, N_m)$$

where $N_1, \ldots, N_m$ is the list of identified entities that are able to violate a security property if k_1 entities out of N_1 and $\ldots$ and k_m entities out of N_m collaborate maliciously. If different terms allow the violation of a security property, then these terms $t_1, \ldots, t_m$ are separated by the ";" symbol, that is

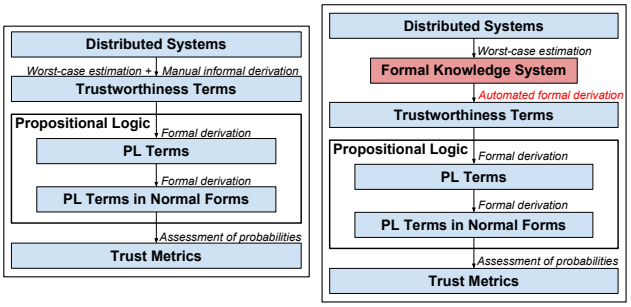
$$t_1; \ldots; t_m.$$

$t_1; t_2$ can be reduced to t_1 if $t_1 = i$ out of N and $t_2 = j$ out of M with $i < j$ and $N \subset M$ holds.

The informal derivation of these terms can miss or misinterpret attack scenarios, which leads to wrong resilience terms. Therefore, in this work, we adapt the framework of

[1]Correspondingly, the critics in [11] is not justified.

 ISBN: 978-1-61208-201-1



(a) Original Research Framework. (b) Extended Research Framework.

Figure 1. Research Framework.

[8] and propose the formal derivation of resilience terms based on formal knowledge representations. The highlighted parts of Figure 1(b) integrate our extension into the existing framework. Due to space constraints, in this paper, we focus on level 1 while we recall that higher levels can be (formally) derived in a straight-forward manner.

IV. Knowledge Calculus

In Section II, we reviewed literature in the context of formal analysis of electronic voting and argued that none of these approaches can be extended for our purpose. In this section, we therefore commit on basics to underlie our ideas and motivate the used methodology. We propose to apply the concept of knowledge representation and reasoning about knowledge as it is a well established concept of artificial intelligence and has been influenced and improved by formal methods. As Dolev-Yao (DY) [23] adversary models have been successfully used to analyze cryptographic protocols also in the context of electronic voting, we integrate a DY adversary model in our approach.

In this section, we first introduce the knowledge algebra to represent terms, which can be known by entities. Thereafter, the knowledge system is introduced in terms of a state transition system, which allows the adversary to extend his knowledge in terms of corrupting election entities. Finally, reasoning over knowledge is realized by the adversarial deduction rules used to extend adversarial knowledge. Due to space constraints, we restrict our attention to entities and inference rules, which will be used in the following examples rather than a more comprehensive specification.

Correspondingly, we do not settle our work in protocol analysis but rather see the contribution in trust and knowledge management in the field of electronic voting schemes.

A. Knowledge Algebra

In this section, we introduce the algebra composed by terms and equations making the semantics of terms.

1) Term Signature: We define the term signature to be

$$Sig = \left(\bigcup_{i \in \mathbb{N}} F^i \right) \cup R$$

where F^i represents the function symbols of arity i and R implements the relation between terms. The signature is later on used to represent known messages and to formalize security properties.

We define a subtype Ent, which embodies entities carrying out an election, namely voters [2] and election services as well as election authorities, such as key holders. We present the entities in extracts while a more detailed consideration depends on the voting scheme under investigation.

$$\begin{aligned} Ent \quad = \quad & \{voter(i) \mid i \in \mathbb{N}\} \cup \quad \text{(*Voters*)} \\ & \{KH(i) \mid i \in \mathbb{N}\} \cup \quad \text{(*Key holders*)} \\ & \{BBS\} \cup \quad \text{(*Ballot box server*)} \\ & \ldots \end{aligned}$$

We refer to $role_{type}$ as a set of entities of a certain type, e.g., $role_{KH} = \bigcup_{i \in \mathbb{N}} KH(i)$. The set of voters $role_{voter}$ is abbreviated by V. The complete function symbols are specified by the following signature:

$$\begin{aligned} F^0 &= Ent \cup \{vote, sk, pk, k, tan, token\} \\ F^1 &= \{hash, ss\} \\ F^2 &= \{sig, a{-}enc, enc\} \\ F^4 &= \{share\} \end{aligned}$$

Apart from entities, the signature provides symbols for votes, secret keys, public keys, symmetric keys, transaction authentication numbers (TAN), tokens, hash values, symmetric and asymmetric encryption. We will provide function $a{-}enc$ with explicit randomness whenever this is of importance to the protocol specification. The function ss denotes the secret sharing of a term into different shares. Each share contains information about the shared term, the index of the share, as well as information about how many shares need to be collected in order to reconstruct the shared term and how many shares exist. Below, we often use asymmetric key pairs where entities sometimes hold different key pairs for different use, such as encryption, signature or database key pairs. By sk_{ent}^{type} we denote the private key of entity ent to be used for $type$.

2) Equations: The semantics of function symbols are given by the following equations, which we abbreviate by E.

$$R(t_1, t_2) = R(t_2, t_1) \qquad (1)$$

$$ss(\overbrace{share(t, i, k, n), \ldots, share(t, j, k, n)}^{k \text{ times}}) = t \qquad (2)$$

Equation 1 indicates the commutativity of the knowledge relation. Following the four eyes trust principle, electronic

[2] Note that voters need to be considered in order to specify the secrecy property while they still remain incorruptible

 ISBN: 978-1-61208-201-1

voting schemes often distribute secrets among independent entities in order to mitigate the risk of small conspiracies violating security properties. Equation 2 prescribes how a distributed secret can be reconstructed using the secret shares. It holds $\forall share(t,a,k,n), share(t,b,k,n),$ with $i \leq a,b \leq j : share(t,a,k,n) \neq share(t,b,k,n)$. By $t_i^{k,n}$ we denote $share(t,i,k,n)$.

Signature Sig and the equation set E lead our electronic voting theory, which underlies the remainder of this paper.

B. Knowledge System

We model the knowledge system as state transition system where transitions between states model corruption of entities. We define $Knowledge$ to be a set of ground terms $T(Sig)$, which embodies the local knowledge of an entity. Global knowledge is defined as composition of the entities' local knowledge bases.

$$GlobalKnowledge \quad ::= \quad Knowledge^*$$

Accordingly, the intruder knowledge refers to type $Knowledge$:

$$IntruderKnowledge \quad ::= \quad Knowledge$$

1) State and Traces: A state is given by an execution trace, the global knowledge of entities as well as the intruder knowledge.

$$State \quad ::= \quad Trace \times GlobalKnowledge \times IntruderKnowledge$$

Collaboration is collectively embodied in the adversary, i.e., the corrupted participants' knowledge sets pass into adversarial ownership. The execution of our corruption model is carried out based on the initial distribution of knowledge. At this point, we only consider one adversarial event, the corruption of participants, which releases their knowledge to the adversary.

$$Event \quad ::= \quad corrupt(id)$$

Traces are composed inductively by sequences of events:

$$Trace \quad ::= \quad Event.Trace$$

Initial State: The initial state of a knowledge system with respect to electronic voting schemes is defined as:

$$s_0 \quad = \quad \epsilon \times K_{init} \times IK_{init}$$

Initially, no identity is corrupt, hence the execution trace is empty. The local knowledge is given by the scheme analysis and is generally abbreviated by K_{init}. After the successful completion of the voting phase, the adversary's knowledge, generally referred to as IK_{init}, is defined by the network model and infrastructural details. The initial intruder knowledge might consist of terms, which are publicly known or which are given by the curious behavior of the adversary, hence the interception of public channels and public bulletin boards.

2) Execution Model: Given a state defined by an event trace tr, the local knowledge states of entities collectively encoded in K and the adversary knowledge given by a set of terms IK, the adversary may issue a corrupt event targeted at entity ID. The execution of this event by the system results in a state s_{i+1} where tr is extended by the recent corrupt event, the local knowledge of entities remains unchanged and the adversary's knowledge set is extended by the local knowledge of the corrupted entity.

$$\frac{s_i = \langle tr, K, IK \rangle \qquad ev = corrupt(ID)}{s_{i+1} = \langle ev.tr, K, IK \cup K(ID) \rangle}$$

C. Adversary Deduction System

Based on the adversarial knowledge resulting from corruption of entities, we introduce a deduction system that allows the adversary to extend gained knowledge in a logical sense, hence based on acquired terms, the adversary is allowed to extend its knowledge according to an inference system, which is given by the rules below. As we only consider secrecy properties, the attacker can only decompose terms rather than synthesize them.

1) Basic Rules: Knowledge is given by means of sets over terms, hence elements of knowledge sets are derivable according to the inference system.

$$\frac{m \in IK}{IK \vdash_E m}$$

2) Asymmetric Encryption Rules: The rule enables the adversary to decrypt publicly encrypted messages if he holds the corresponding private key.

$$\frac{IK \vdash_E sk_i^{enc} \qquad IK \vdash_E a{-}enc(pk_i^{enc}, m)}{IK \vdash_E m}$$

3) Symmetric Encryption Rules: The adversary can derive a message if he holds the encryption of that message and the corresponding symmetric encryption key.

$$\frac{IK \vdash_E k \qquad IK \vdash_E enc(k,m)}{IK \vdash_E m}$$

4) Hash Rules: The adversary is allowed to derive hash values of messages he holds.

$$\frac{IK \vdash_E m}{IK \vdash_E hash(m)}$$

5) Signature Rules: Signatures reveal the relation between signer and the signed message.

$$\frac{IK \vdash_E sig(sk_i, m)}{IK \vdash R(i,m)}$$

 ISBN: 978-1-61208-201-1

6) Secret Sharing Rules: We allow the adversary to use the ss operator in order to reconstruct distributed terms.

$$\frac{IK \vdash_E t_1 \quad \ldots \quad IK \vdash_E t_n}{IK \vdash_E ss(t_1, \ldots, t_n)}$$

7) Relational Rules: Local knowledge sets of entities are considered to be faithful, i.e., the union of these sets never allows for inconsistencies. The rules given below specify the projection on relations and the transitivity of the knowledge relation.

$$\frac{IK \vdash_E R(a,b)}{IK \vdash_E a} \qquad \frac{IK \vdash_E R(a,b)}{IK \vdash_E b}$$

$$\frac{IK \vdash_E R(t,x) \quad IK \vdash_E R(t,y)}{IK \vdash_E R(x,y)}$$

V. Secrecy

The term secrecy often refers to different details while the underlying idea remains mainly the same with respect to electronic voting. In contrast to classical cryptographic protocols, secrecy properties of electronic voting schemes do not require the secrecy of terms, as the public availability of votes is central to the public nature of elections. In fact, secrecy in electronic voting resembles the idea of anonymity in cryptographic protocols, i.e., an adversary should not be able to link voters and their votes. Therefore, constructing relations between terms may enable the adversary to violate secrecy properties and is therefore of central importance in our approach.

The *intruder deduction problem* for secrecy denotes the problem to deduce a term t from a set of terms IK based on a given inference system. In general, this fact is abbreviated by

$$IK \vdash_{IS} t$$

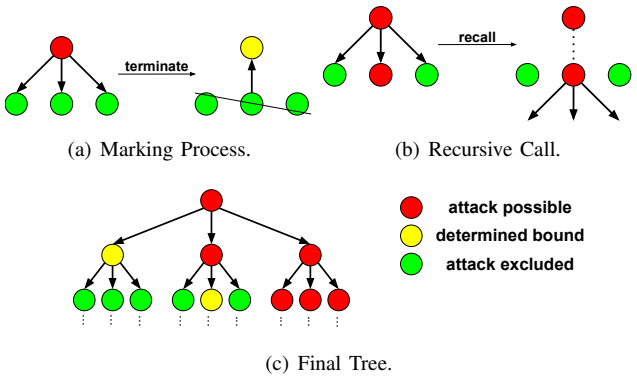
where IS denotes the corresponding inference system. Let a state $s = \langle tr, K_{init}, IK \rangle$ be given. The intruder deduction problem for secrecy in electronic voting refers to

$$IK_s \vdash R(voter(i), vote(j))$$

for some $i, j \in \mathbb{N}$. Therefore, assumptions about investigated states have to be made, which subsequentally allows for resilience term derivation. By logical means this can be expressed as follows: For a state $s = \langle tr, K_{init}, IK \rangle$, one needs to find $min|bound(role_r)|$ for all $role_r$ where $\left(\bigwedge_{i \in role_r}^{bound(role_r)} corrupt(role_r(i)) \notin tr\right)$ such that $IK_s \nvdash R(voter(i), vote(j))$.

VI. Determination of Resilience Terms

The ultimate goal of our approach is to automatically derive the minimal sets of entities that need to be trusted in order to ensure the security properties of interest. Before diving into details of the algorithm, we emphasize that the algorithm assumes a worst-case estimation about the obtained knowledge of local entities. Hence, we manually consider the entire protocol and formalize relational knowledge and knowledge of terms that entities might gather if they do not behave properly. Once, this estimation is available, the following recursive algorithm is executed:



(a) Marking Process. (b) Recursive Call.

(c) Final Tree.

Figure 2. The proposed algorithm.

Assume a scenario with n entities. In such a situation, the prover is initially called with a collaboration of all n entities in order to determine if secrecy is violated. If so, n instances of the prover are called each analyzing a different collaboration scenario, i.e., each call excludes one entity from the collaboration. Once, a collaboration can not violate secrecy, we cut the tree at this point and give the result back. In case no child of a node can violate secrecy, then this node is marked and will be used for the final computation as depicted in Figure 2(a). In case some children of a node can violate the property, while other children can not, the algorithm is recursively called for the violating children as shown in Figure 2(b). The final output of this algorithm corresponds to a tree of the form given in Figure 2(c).

Of crucial importance to our proposal is the handling of entities in identical roles, e.g., key holders. In order to reduce the computational complexity, the algorithm therefore is designed in the following way: Think of a 3 out of 6 threshold scheme for distributed decryption among $\{KH_1, \ldots, KH_6\}$. We therefore invent a new super entity $KH_{3,6}$, which correspond to the reconstructed decryption key. This entity is used throughout the algorithmic proceeding. In the final k-resilience value, this entity is than resubstituted by 3 out of $\{KH_1, \ldots, KH_6\}$. Finally, marked notes represent the determined resilience term.

In worst-case the number of prover calls is

$$\#calls(prover) = \sum_{i=1}^{n} \binom{n}{i}.$$

Note that we follow a top-down approach, although it can easily be adapted to a bottom-up proceeding.

 ISBN: 978-1-61208-201-1

VII. Evaluation of eVoting Schemes

We deploy the calculus in order to evaluate different electronic voting schemes by means of resilience terms. We briefly present the voting schemes that have been investigated also in [8], namely Polyas, Helios and the Estonian voting system. Thereafter, we deduce the obtained local knowledge sets from a worst-case, which allows us to automatically derive the corresponding resilience terms and contrast these terms with the previously informally derived terms.

A. Polyas

Polyas is a remote voting system developed by Micromata in 1996, with which many elections have been carried out. Polyas comprises the following components:

Printing Service (PS): The PS prints the election material and sends this material to the voters via postal mail.

Election Registration Server (ERS): The ERS implements the electoral roll, which is accessed to verify the eligibility of the voter.

Validation Server (VS): Similar to the ERS, the VS re-verifies the eligibility of the authenticating voter such that both the ERS and the VS control each other.

Ballot Box Server (BBS): The BBS stores the encrypted votes cast by eligible voters.

Tallying Component (TC): The TC is an offline component, which tallies the stored votes in the BBS after the election has terminated.

Key holders: There are two independent election officials (KH_1, KH_2), which hold the private key shares corresponding to the public key used to encrypt votes.

Figure 3 depicts the protocol interaction in form of a sequence diagram. For further information about the protocol specification, we refer to [24], [8], [20]. Note that the communication between different entities is secured by https connections. Once, the voting phase has terminated, the content of the BBS is carried over to the TC, which is offline, where the votes are collectively decrypted by the collaboration of both key holders.

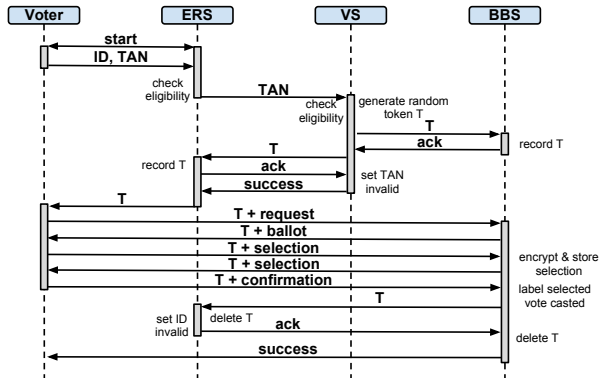


Figure 3. Polyas Voting Scheme.

The entities' knowledge can be formalized in the following way:

Election Registration Server: The hash values of TANs prepared for eligible voters are available to the ERS.

- $\forall i \leq |V|$ s.t. $R(voter(i), tan(j)) : hash(tan(j)) \in K(ERS)$

All voters' IDs are available to the ERS.

- $\forall i \leq |V| : voter(i) \in K(ERS)$

The ERS is aware of the voter-TAN relation.

- $\forall i \leq |V| : R(voter(i), tan(j)) \in K(ERS)$

The ERS knows the relation between TAN and tokens generated by the VS.

- $\forall i \leq |V|$ s.t. $R(voter(i), tan(j)) : R(tan(j), token(k)) \in K(ERS)$

Printing Service: The PS receives the voting material and distributes eligible TANs among the voters via postal mail, hence the service knows:

- $\forall i \leq |V| : R(voter(i), tan(j)) \in K(PS)$

Validation Server: TANs prepared for eligible voters are available to the VS.

- $\forall i \leq |V|,$ s.t. $R(voter(i), tan(j)) : tan(j) \in K(VS)$

Furthermore is the relation between prepared TANs and prepared tokens known to the VS, as the VS generates these tokens.

- $\forall i \leq |V|$ s.t. $R(voter(i), tan(j)) : R(tan(j), token(k)) \in K(VS)$

Ballot Box Server: The tokens prepared for eligible voters are available to the BBS.

- $\forall i \leq |V|$ s.t. $R(voter(i), tan(j)), R(tan(j), token(k)) : token(k) \in K(BBS)$

We recall that the local knowledge sets are determined by a worst-case estimation about the complete protocol run. Hence, the BBS might store information about votes together with the respective token used to cast this vote.

- $\forall i \leq |V|,$ s.t. $R(voter(i), token(k)) : R(token(k), vote(l)) \in K(BBS)$

The BBS might store the encrypted version of the cast votes, together with the corresponding tokens used to submit these votes.

- $\forall i \leq |V|,$ s.t. $R(voter(i), token(k)) : R(token(k), a{-}enc(pk_{DB}, vote(l))) \in K(BBS)$

Tallying Component: The TC obtains the knowledge from the BBS. The TC furthermore stores the encrypted version of each cast vote together with the vote.

- $\forall i \leq |V|,$ s.t. $R(voter(i), token(k)), R(token(k), a{-}enc(pk_{DB}, vote(l))) : R(enc(pk_{DB}, vote(l)), vote(l)) \in K(TC)$

Key holders: The private database key sk_{DB} is shared among two independent key holders.

- $\forall i \in \{1, 2\} : sk^{2,2}_{DB_i} \in K(KH_i),$ s.t. $sk^{2,2}_{DB_1} \neq sk^{2,2}_{DB_2}$

 ISBN: 978-1-61208-201-1

Resilience Term Derivation: The first boundary our algorithm detects leads to a state $s = \langle tr, K_{init}, K(BBS) \cup K(ERS) \cup \ldots \rangle$, hence the corruption of the ERS and the BBS. In s, the relational axioms allow the adversary to reason in the following way:

$$\frac{\begin{array}{c} IK_s^{ERS} \vdash R(voter(i), tan(j)) \\ IK_s^{ERS} \vdash R(tan(j), token(k)) \end{array}}{\begin{array}{c} IK_s^{ERS} \vdash R(voter(i), token(k)) \\ IK_s^{BBS} \vdash R(token(k), vote(l)) \end{array}}{IK_s \vdash R(voter(i), vote(l))}$$

Note, that our algorithm does not investigate any further conspiracies where ERS and BBS are involved as such conspiracies automatically also violate the secrecy property. Trace tr with $corrupt(PS)$, $corrupt(VS)$, $corrupt(BBS) \in tr$ results in s, allowing the adversary to reason in the following way:

$$\frac{\begin{array}{c} IK_s^{PS} \vdash R(voter(i), tan(j)) \\ IK_s^{VS} \vdash R(tan(j), token(k)) \end{array}}{\begin{array}{c} IK_s \vdash R(voter(i), token(k)) \\ IK_s^{BBS} \vdash R(token(k), vote(l)) \end{array}}{IK_s \vdash R(voter(i), vote(l))}$$

Hence, conspiracies between the PS, the VS and the BBS allow the adversary to violate the secrecy property.

Finally, the algorithm terminates and returns the following secrecy resilience term for Polyas:

$$\begin{array}{rcl} t & = & 2 \text{ out of } \{ERS, BBS\}; \\ & & 3 \text{ out of } \{PS, VS, BBS\} \end{array}$$

The informal resilience term derivation in [8] resulted in the fact that BBS can guarantee the secrecy of the vote. This coincides with our result. Furthermore, their result states that also ERS and VS together can ensure secrecy. As opposed to our consideration, they did not take into account the printing service, therefore our second attack is out of scope for their scenario. Hence, the entities ERS and VS should not be part of their resilience term for the sake of consistency.

B. Helios

The Helios voting system has been introduced in [21] by Ben Adida, while currently Helios version 3.1 is available. In contrast to prior and later versions, Helios 2.0 is based on homomorphic tallying [25]. This work is based on Helios 2.0 as this version has already been investigated manually and a resilience term has been determined [8]. The protocol is based on the idea of separating ballot preparation/encryption and authentication. Helios comprises the following components:

Election Builder (EB): The EB initially determines candidates and eligible voters and provides eligible voters with their login data and the URL.
Voting JavaScript: The script allows the voter to process his vote and to interact with the backend of the system. The JavaScript is launched by the Helios website. The randomness used to encrypt the voter's choice is stored within this script. We assume the script to behave properly and therefore do not distinguish between voter and his script.
Ballot Verifiers (BV): There are three BV, which can be involved by voters to audit the encryption process.
Authentication Server (AS): The AS allows the voter to authenticate himself in order to submit his encrypted vote.
Bulletin Board (BB): The BB is a public channel on which voters may verify if their cast votes are stored.
Tallying Component (TC): According to the multiple execution of a duty principle, there are two independent offline tallying components TC_1 and TC_2, which tally the stored votes on the BB once the election has been finished.
Key holders (KH): There are six independent election officials that hold private key shares in order to decrypt stored ballots.

An overview over the Helios system is given in Figure 4. At the beginning of the election, the Helios election builder

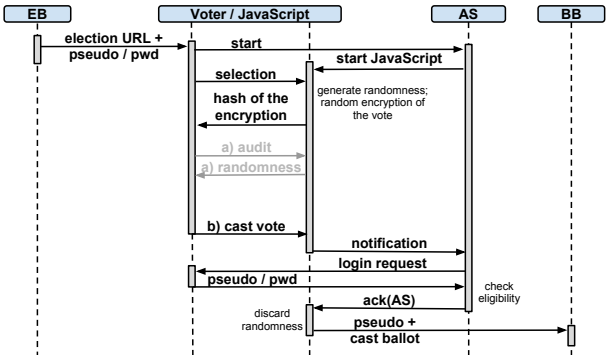


Figure 4. Helios Voting Scheme.

sends the voter an invitation e-mail containing a link to the election website together with his ephemeral login data. At the end of the election process, the BB contains all voters' pseudonyms together with the encrypted vote and the hash value of the encrypted votes such that voters may verify the process. Once, the election has terminated, the encryptions are homomorphically summed up and decrypted by at least three out of six key holders.

The local knowledge sets of the entities are given by:
Election Builder: The EB stores the association between voter and pseudonym for each voter.

- $\forall i \leq |V| : R(voter(i), pseudo(j)) \in K(EB)$

Voting JavaScript: The voting java script of voter i stores the association between the voter's pseudonym and his vote.

 ISBN: 978-1-61208-201-1

- $R(voter(i), pseudo(j))$:
  $R(pseudo(j), vote(l)) \in K(VJS_i)$

Furthermore, the encryption of this vote together with the used randomness is stored.

- $R(voter(i), pseudo(j))$:
  $R(pseudo(j), a{-}enc(vote(l), pk_{TC}, r)) \in K(VJS_i)$
- $R(voter(i), pseudo(j))$,
  $R(pseudo(j)$,
  $a{-}enc(vote(l), pk_{TC}, r)) \in K(VJS_i)$:
  $r \in K(VJS_i)$

Ballot Verifiers: We omit the consideration of ballot verifiers in the reasoning as the auditing of these servers causes a new voting and encryption step. These verifiers therefore do not influence the resilience term.
Authentication Server: The AS obtains the pseudonyms of eligible voters.

- $R(voter(i), pseudo(j)) : pseudo(j) \in K(AS)$

Bulletin Board: The BB stores and publishes the encrypted version of the cast votes, together with the corresponding pseudonym used to submit these votes.

- $\forall i \leq |V|$ s.t. $R(voter(i), pseudo(j))$:
  $R(pseudo(j), a{-}enc(pk_{TC}, vote(l), r)) \in K(BB)$

The BB stores and publishes the hash value of the encrypted votes, together with the corresponding pseudonym used to submit these votes.

- $\forall i \leq |V|$ s.t. $R(voter(i), pseudo(j))$:
  $R(pseudo(j)$,
  $hash(a{-}enc(pk_{TC}, vote(l), r)) \in K(BB)$

Tallying Component: The TC obtains the knowledge from the bulletin board.
Key holders: The private database key sk_{TC} is shared among six independent key holders in the following way:

- $\forall i \in \{1,2\} : sk^{3,3}_{TC\_1} \in K(KH_i)$
- $\forall i \in \{3,4\} : sk^{3,3}_{TC\_2} \in K(KH_i)$
- $\forall i \in \{5,6\} : sk^{3,3}_{TC\_3} \in K(KH_i)$

We denote the groups of key holders that hold identical key shares by KH^1, KH^2, KH^3. We emphasize that the content of the bulletin board is public, which means that the attacker after the tallying is aware of relation $R(pseudo(j), a{-}enc(pk_{TC}, vote(l), r))$.

Resilience Term Derivation: The algorithm detects that in case one key holder of each key holder group is corrupt, the key sk_{TC} can be reconstructed. If additionally the EB is compromised this leads to state s, which allows the following reasoning:

$$\frac{\dfrac{IK_s^{KH^1} \vdash sk^{3,3}_{TC\_1} \quad \dfrac{IK_s^{KH^2} \vdash sk^{3,3}_{TC\_2} \qquad IK_s^{KH^3} \vdash sk^{3,3}_{TC\_3}}{IK_s \vdash sk_{TC}} \quad IK_s \vdash R(pseudo(l), a{-}enc(pk_{TC}, vote(m), r))}{IK_s \vdash R(pseudo(l), vote(m))} \quad IK_s^{EB} \vdash R(voter(o), pseudo(l))}{IK_s \vdash R(voter(o), vote(m))}$$

Finally, the algorithm terminates and returns the following secrecy resilience term for the Helios scheme:

$$\begin{aligned} t \;=\; & (1+1+1+1) \text{ out of} \\ & (\{EB\}, \{KH_1, KH_2\}, \\ & \{KH_3, KH_4\}, \{KH_5, KH_6\}) \end{aligned}$$

Level 1 resilience term in [8] expresses that the authentication server and one key holder of each group needs to be trusted, while the authentication server in their consideration plays the role of our EB. This observation coincides with our result.

C. Estonian Voting System

In 2005, Estonia was the first country in which electronic elections were legally binding for the municipal elections. Their system relies on the Estonian ID card, which is both the regular ID card and a smart card capable of pursuing legally binding digital signatures. The Estonian electronic voting system comprises the following components:
Voter Application (VA): Each voter runs a VA on which he selects his preferred candidates. After this, the application encrypts the vote by the election key and signs the ballot with the voter's private key stored on his national ID card.
Vote Forwarding Server (VFS): The VFS is directly accessible over the internet and once the voter prepared his signed ballot, this ballot is sent to the VFS, which then forwards the ballot to the vote storage server.
Vote Storage Server (VSS): The VSS receives ballots from the VFS. After the election, the VSS eliminates double votes and votes from ineligible voters. It then removes all signatures and stores the unsigned ballots on a CD.
Vote Counting Application (VCA): The VCA reads the encrypted ballots from the provided CD upon which the key holders collectively start the tallying process.
Key holders (KH): There are seven key holders among which four need to collaborate in order to decrypt cast votes.

A simplified overview of the Estonian internet voting system is given in Figure 5. Apart from the description presented here, the Estonian system allows vote updating, which is omitted in our consideration due to space constraints.

The electronic voting system deployed in Estonia implements the Estonian postal voting by electronic means. Once, the election has terminated, the stored encrypted votes in VSS are burned on a CD and carried over to the VCA.

 ISBN: 978-1-61208-201-1

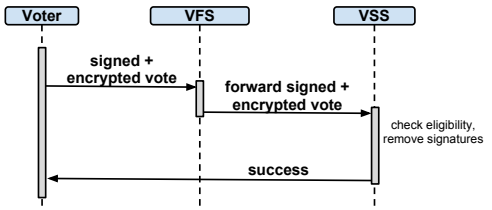


Figure 5. Estonian Voting Scheme.

There, the encrypted votes are mixed and decrypted by a collaboration between four out of seven key holders.

Vote Forwarding Server: The VFS receives signed encrypted votes from the voters.

- $\forall i \leq |V| : voter(i) \in K(VFS)$
- $\forall i \leq |V|$ s.t. $R(voter(i), vote(j)) :$ $sig(sk_i^{sig}, a{-}enc(pk_{VCA}, vote(j))) \in K(VFS)$

Vote Storage Server: The VSS stores the signed encrypted votes from the VFS.

- $\forall i \leq |V| : voter(i) \in K(VSS)$
- $\forall i \leq |V|$ s.t. $R(voter(i), vote(j)) :$ $sig(sk_i^{sig}, a{-}enc(pk_{VCA}, vote(j))) \in K(VSS)$

Vote Counting Application: The VCA only receives encrypted votes.

- $\forall i \leq |V|$ s.t. $R(voter(i), vote(j)) :$ $a{-}enc(pk_{VCA}, vote(j)) \in K(VCA)$

The VCA stores the relation between encrypted votes and votes.

- $\forall i \leq |V|$ s.t. $R(voter(i), vote(j)) :$ $R(a{-}enc(pk_{VCA}, vote(j)), vote(j)) \in K(VCA)$

Key holders: The Estonian Voting System implements a distributed threshold scheme in order to reconstruct the secret key of the VCA.

- $\forall i \in \{1, \ldots, 7\} : sk_{VCA_i}^{4,7} \in K(VCA_i)$ s.t. $sk_{VCA_k}^{4,7} \neq sk_{VCA_l}^{4,7}$

We emphasize that the tallying of ballots in the VCA is public, once the key holders provided their keys, which means that the attacker after the tallying is aware of relation $R(a{-}enc(pk_{VCA}, vote(j)), vote(j))$.

Resilience Term Derivation: The first boundary our proposed algorithm returns is the corruption of the VSS and the VCA as this allows the following reasoning:

$$\frac{\dfrac{IK^{VSS} \vdash sig(sk_i^{sig}, a{-}enc(pk_{VCA}, vote(j)))}{IK^{VSS} \vdash R(voter(i), a{-}enc(pk_{VCA}, vote(j)))} \quad IK^{VCA} \vdash R(enc(pk_{VCA}, vote(j)), vote(j))}{IK \vdash R(voter(i), vote(j))}$$

Finally, the algorithm terminates and returns the following secrecy resilience term for the Estonian internet voting system:

$$\begin{aligned} t \;=\; & (1+1) \text{ out of } \{VFS, VSS\}, \{VCA\}; \\ & (1+4) \text{ out of } (\{VFS, VSS\}, \{KH_1, \ldots, KH_7\}) \end{aligned}$$

The informal derivation of the resilience term in [8] led to the fact that VSS can guarantee the secrecy of the vote. Our result however shows that also VCA together with 4 out of T key holders can ensure secrecy. This possibility has not been discovered in [8].

VIII. Conclusion and Future Work

This paper takes up the resilience terms proposed by Volkamer et al. [8] used to evaluate distributed systems. Based on this approach we developed a knowledge calculus upon a theory adapted to most general electronic voting schemes. This calculus allows for formal reasoning over our proposed theory. On the basis of this calculus, we formalized the secrecy property and described how to determine resilience terms in this framework. Based on a worst-case knowledge estimation, we iteratively investigate collaboration scenarios and thereby deduce the resilience term. We finally applied our proposal to three electronic voting schemes and came up with mistakes in previously informally derived terms of two of these three schemes.

In future work, we plan to incorporate our theory into SPASS [26], an automated theorem prover, in order to fully automatize the deduction, which allows us to run performance tests on our proposal. The defined theory therefore has to be very precise, such that attacks or the absence of attacks may be decided and proven in an automated way. In protocol analysis, the secrecy property is undecidable in the general case. In this work, we consider completely passive adversaries, for which it has recently been shown that deciding knowledge in security protocols can be done in polynomial time under some e-voting theories [27]. We therefore plan to compare these theories with our own theories in order to obtain decidability results about our own theories. In addition, in the future we will consider the adversary's capability of linking voters and votes by means other than the proposed theory, e.g., an adversary might link voters and votes by IP addresses or even timestamps of messages. Furthermore, in order to integrate other security properties, e.g., integrity, a more adequate adversary model has to be considered. In future work, we therefore plan to allow the adversary to become active at an earlier stage, hence manipulating, dropping or injecting messages throughout the protocol run, thereby incorporating advances from the protocol analysis. Due to the nature of elections, the investigation of electronic voting schemes always comes along with legal considerations. Therefore, resilience terms compliant with legal frameworks need to be discussed and determined in close collaboration with legal scientists.

Acknowledgment

This paper has been developed within the project "ModIWa2" - Juristisch-informatische Modellierung von Internetwahlen - which is funded by the Deutsche Forschungsgemeinschaft (DFG, German Science Foundation).

 ISBN: 978-1-61208-201-1

REFERENCES

[1] Melanie Volkamer and Roland Vogt. Basic set of security requirements for Online Voting Products. (BSI-PP-0037). Common Criteria Protection Profile, 2008.

[2] Hugo Jonker and Melanie Volkamer. Compliance of RIES to the proposed e-voting protection profile. In *First International Conference on E-voting and Identity*, pages 50–61. Springer-Verlag, 2007.

[3] Michael Backes, Catalin Hritcu, and Matteo Maffei. Automated verification of remote electronic voting protocols in the applied pi-calculus. In *21st IEEE Computer Security Foundations Symposium*, pages 195–209. IEEE Computer Society, 2008.

[4] Steve Kremer and Mark Ryan. Analysis of an electronic voting protocol in the applied pi calculus. In *14th European Symposium On Programming*, volume 3444 of *LNCS*, pages 186–200. Springer, 2005.

[5] Atsushi Fujioka, Tatsuaki Okamoto, and Kazuo Ohta. A practical secret voting scheme for large scale elections. In *Proceedings of the Workshop on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, pages 244–251. Springer-Verlag, 1993.

[6] Jonathan Katz, Steven Myers, and Rafail Ostrovsky. Cryptographic counters and applications to electronic voting. In *Eurocrypt 2001*, pages 78–92, 2001.

[7] R. Küsters, T. Truderung, and A. Vogt. Proving coercion-resistance of Scantegrity II. In *12th International Conference on Information and Communications Security*, volume 6476, pages 281–295. Springer, 2010.

[8] Melanie Volkamer and Rüdiger Grimm. Determine the resilience of evaluated internet voting systems. In *First International Workshop on Requirements Engineering for e-Voting Systems*, pages 47 – 54. IEEE Digital Library, 2009.

[9] Komminist Weldemariam, Richard A. Kemmerer, and Adolfo Villafiorita. Formal specification and analysis of an e-voting system. In *Fifth International Conference on Availability, Reliability and Security*, pages 164–171, 2010.

[10] Martín Abadi and Cédric Fournet. Mobile values, new names, and secure communication. In *28th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 104–115. ACM, 2001.

[11] V. Cortier and B. Smyth. Attacking and fixing Helios: An analysis of ballot secrecy. In *24th IEEE Computer Security Foundations Symposium*, pages 297 –311, 2011.

[12] H. L. Jonker and E. P. De Vink. Formalising receipt-freeness. In *Information Security*, volume 4176 of *LNCS*, pages 476–488. Springer, 2006.

[13] Rdiger Grimm, Katharina Hupf, and Melanie Volkamer. A formal IT-security model for the correction and abort requirement of electronic voting. In *4th International Conference on Electronic Voting, EVOTE*, volume 167 of *LNI*, pages 89–107. GI, 2010.

[14] Katharina Bräunlich and Rudiger Grimm. Formalization of receipt-freeness in the context of electronic voting. In *Sixth International Conference on Availability, Reliability and Security*, pages 119 –126, 2011.

[15] Walid Saad, Tansu Alpcan, Tamer Basar, and Are Hjørungnes. Coalitional game theory for security risk management. In *Fifth International Conference on Internet Monitoring and Protection*, pages 35–40. IEEE Computer Society, 2010.

[16] Thomas Moscibroda, Stefan Schmid, and Roger Wattenhofer. The price of malice: A game-theoretic framework for malicious behavior in distributed systems. *Internet Mathematics*, 6:125–155, 2009.

[17] Tyrone Grandison and Morris Sloman. A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 3(4):2–16, 2000.

[18] Sebastian Ries. *Trust in Ubiquitous Computing*. PhD thesis, TU Darmstadt, 2009.

[19] Council of Europe. Legal, Operational and Technical Standards for E-Voting. Recommendation Rec (2004)11 adopted by the Committee of Ministers of the Council of Europe and explanatory memorandum. 2004.

[20] Kai Reinhard and Wolfgang Jung. Compliance of POLYAS with the BSI protection profile - basic requirements for remote electronic voting systems. In *First Conference on E-Voting and Identity*, pages 62–75, 2007.

[21] Ben Adida. Helios: Web-based open-audit voting. In *Proceedings of the 17th conference on security symposium*, pages 335–348. USENIX Association, 2008.

[22] Guido Schryen, Melanie Volkamer, Sebastian Ries, and Sheikh Mahbub Habib. A formal approach towards measuring trust in distributed systems. In *ACM Symposium on Applied Computing*, pages 1739–1745. ACM, 2011.

[23] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29:198–208, 1983.

[24] Maina M. Olembo, Patrick Schmidt, and Melanie Volkamer. Introducing verifiability in the polyas remote electronic voting system. In *Sixth International Conference on Availability, Reliability and Security*, pages 127–134. IEEE, 2011.

[25] Ben Adida, Olivier Pereira, Olivier De Marneffe, and Jean Jacques Quisquater. Electing a university president using open-audit voting: Analysis of real-world use of Helios. In *Electronic Voting Technology/Workshop on Trustworthy Elections*, 2009.

[26] Christoph Weidenbach, Uwe Brahm, Thomas Hillenbrand, Enno Keen, Christian Theobalt, and Dalibor Topić. SPASS version 2.0. In *18th International Conference on Automated Deduction*, volume 2392 of *LNAI*, pages 275–279. Springer, 2002.

[27] Mouhebeddine Berrima, Narjes Ben, Rajeb Veronique Cortier, Theme Sym, Mouhebeddine Berrima, Narjes Ben Rajeb, Veronique Cortier, and Equipe projet Cassis. Deciding knowledge in security protocols under some e-voting theories. In *20th International Conference on Rewriting Techniques and Applications*, pages 148–163. Springer, 2009.

 ISBN: 978-1-61208-201-1

sGUTS: Simplified Grid User Trust Service for Site Selection

Ioanna Dionysiou
Department of Computer Science
University of Nicosia
Nicosia, Cyprus
dionysiou.i@unic.ac.cy

Harald Gjermundrod
Department of Computer Science
University of Nicosia
Nicosia, Cyprus
gjermundrod.h@unic.ac.cy

***Abstract*—Even though trust plays a significant role during decision-making in open collaborative environments, still Grid user trust mechanisms have not been widely deployed in Grid computing settings. In this paper, a conceptual framework that is an extension of a novel Grid user trust service (GUTS) is presented, which aims at leveraging Grid functionality with trust mechanisms with a special focus on achieving end-user trust in an intuitive and practical manner. Trust in GUTS is utilized during the grid site selection process, where sites are ranked based on expected service requirements for a user grid project. In the proposed conceptual framework, the center of the trust management process is the user who decides and specifies the needs of his/her project, which in turn are mapped to trust requirements.**

***Keywords*-grid computing, trust mechanisms, user perceptive**

I. INTRODUCTION

Trust is an abstraction of individual beliefs that an entity has for specific situations and interactions. It encompasses even more than message confidentiality and source authentication, which have been the traditional trust scopes. Trust's broader scope covers not only security issues but behavioral and Quality of Service (QoS) issues as well. Consider a data dissemination service, that operates on the following policy: valid and non-malicious information (behavioral requirement) is publicly available but must not be tampered with (security requirement) and must be received in a timely manner (QoS requirement). In order to enforce this policy the appropriate security, behavioral, and QoS mechanisms must be in place to implement the policy. Digital signing algorithms can guarantee message integrity but they offer no assurance about the quality of the message contents; this is the task of behavioral mechanisms that deduce behavioral patterns and trends for the information producer. Finally, QoS mechanisms are needed to provide guarantees that the information producer and the network will meet the QoS properties as contracted. We call behavior, security and QoS the three *general trust facets*, which are further refined into more specific facets called *requirements*. Requirements include authentication, competence, and delivery rate. Those, could be further refined into attributes. Any trust requirement for a distributed application can be categorized as security, behavioral, or QoS requirement.

While trust is an integral part of decision making in collaborative models, there is no unique way to determine the right level of trust, or which facets to use. Researchers have defined trust concepts for many perspectives, with the result that trust definitions overlap or even contradict each other. The reason is that decisions about how to evaluate each facet lie with the evaluator and can differ substantially from situation to situation. End-to-end trust is essential for topologies where interactions are dynamic and they always involve the collaboration of multiple entities to disseminate data from its source to its destinations. Needless to say, trust is useful only if it is managed in an appropriate and systematic manner. An entity's beliefs are not static but they change as time progresses and new information is processed into knowledge. Trust must evolve in a consistent manner so that it still abstracts the entitys beliefs accurately. In this way, an entity continuously makes informed decisions based on its current beliefs.

Collaborative settings, such as grid environments, where risk and uncertainty are inherent due to their open nature could greatly benefit from using trust as an integral part of decision-making. For example, a grid user could select the most *trustworthy* site from a pool of available sites to submit a job. A grid user could specify *trust requirements* in a parametrized job description. Sites, offering computational resources, could be rated based on their *reputation* among grid users. Trust in the Grid environment has been analyzed and various systems have been proposed (a summary of such systems can be found in [5]). The difference between these systems and the one proposed in this paper is that the former have focused on how to define/model trust in a Grid environment from the system point of view, while sGUTS tries to abstract away the notion of trust from the end user and present him/her with a set of questions that specify the trust needs for a project. Based on these questions, the trust requirements will be automatically derived. To the best of our knowledge, trust is not utilized in this manner by existing trust frameworks for grid infrastructures.

This paper extends a framework that utilizes trust for ranking grid sites and in consequence, allowing grid users to select a site that is the most appropriate for their specific needs. The proposed framework is at the current stage a

 ISBN: 978-1-61208-201-1

conceptual framework and implementation is currently under progress. The primary contribution is the simplification of the service requirements specification by the end-user, which is done in an intuitive manner. It is not always apparent to the Grid user what is the most appropriate configuration for a particular job, something that is vital for selecting the site that best matches the job requirements. In order to address this limitation, the configuration is automatically generated upon the user responses to a predefined set of questions.

The remaining of the paper is organized as follows: Section II discusses existing trust approaches in grid infrastructures, followed by Section III that presents an overview of Grid User Trust Service (GUTS), a trust-based ranking framework applicable to grid interactions. Section IV extends this framework by simplifying the trust specification process. Finally, Section V concludes.

II. Trust Management in Grid Environments

A computational Grid [11],[12],[10] is a collection of distributed, possibly heterogeneous resources that can be used as an ensemble to execute computational-intense applications, such as earth observation, climate modeling, and biology applications. The two pillars of the Grid paradigm are access to shared services and support of multi-user collaboration, while the resource owner is always in control. Sites are organized in one or more virtual organizations, thus creating federations of central services, such as cross-domain authentication, authorization, job-site matching, and job dispatching. Authorized users access computational and storage resources of a site by contacting either the central services or the site itself.

The Grid must be managed to allow coordination of the dynamic cross-organizational resource sharing among virtual organizations not only in an efficient manner but securely as well. This is nontrivial to achieve, mainly due to the self-managed and unpredictable nature of the virtual organizations. Nevertheless, there are deployed mechanisms that provide a number of security services. For instance, a single sign-on authentication mechanism is already available via proxy certificates. Authorization is implemented via access control lists. X.509 certificates could be used not only to authenticate a user but to encrypt traffic flows.

Humphrey et al. [15] analyzed a comprehensive set of Grid usage scenarios with regard to security requirements. However, cryptographic algorithms and access control schemes cannot be used to reason about the more general concept of trust, – the *belief* that an entity will behave as expected under certain conditions – as there are no provisions for a number of security, behavioral, and QoS issues such as data privacy, site administrators qualifications, and service reliability provided by the various sites. An authenticated and authorized user has no guarantees that the Grid infrastructure will successfully carry out the execution of a submitted job. The Grid user remains defenseless against job failures, which according to a recent study [18] account for a large percentage of all submitted jobs, and attempts to compensate for any potential failures by submitting the same job to multiple sites.

The failures could be attributed to security, behavioral, or QoS factors, thus making the Grid environment the ideal setting for deploying trust as an integral part of the decision-making. In the recent years, there has been an increasing interest in addressing specific trust challenges in Grid environments.

In [21], the Trust domains establishment is mentioned as being one of the three key functions in a Grid Security Model, where virtual organizations establish trust among users and resources that is expressed in policies and proxy certificates. The authors in [6] leverage the authentication and authorization capabilities of the Grid security framework using trust negotiation with PeerTrust policy language whereas the importance of trust negotiation is reiterated in [17]. Similarly, [1] uses trust federation and dynamic authorization supported by GRIA middleware to demonstrate the dynamic federation of resources capability. The research work in [7] focuses on a decentralized resource access control scheme using trust chains and an extended SPKI/SDSI that allow intermediate levels of trust to be expressed per chain, rather as a binary model of valid or invalid. In [20], [4], and [16] trust management systems for Grids are presented, which assist in evaluating the trust value of the various Grid sites and specify how to set the metrics trust evaluation.

A more general approach to trust is presented in the survey by Arenas et al. in [3], which discusses the trust classifications in Internet services [14] from the Grid perspective. Furthermore, [2] investigates the possibility of exploiting reputation systems for managing virtual organizations. The SCOUT [22] middleware assists the user in belief calculation and evidence source trust calculation in order to use service in a Service Oriented Architecture.

Still, there is no implementation of a suite of trust mechanisms that the average Grid end-user could utilize to specify its trust requirements and incorporate them in decision-making. Although, GridAdmin [19] is a trust management system to be used by administrators of Grid sites, and not end users. The proposed work in [8] investigates the emerging technological challenges associated with the support of such a comprehensive user-oriented adaptive trust framework deployed in Grid infrastructures.

III. Grid User Trust Service (GUTS) Overview

The Grid User Trust Service (GUTS) framework is a trust management service tailored to the needs of a typical Grid user [9]. It comprises of three main components, as shown in Figure 1. The first component, **Grid Middleware-Agnostic Trust Specification**, allows a user to specify the trust requirements for a Grid service. The second component,

 ISBN: 978-1-61208-201-1

Grid Middleware-Dependent Trust Specification, maps those general requirements to the specific Grid infrastructure, yielding the trust profile of a project. Finally, the **Trust Management and Visualization** component gathers and evaluates evidence provided by the specific Grid infrastructure, updates the trust profiles accordingly, and produces a ranking list of the various Grid sites.

Starting with the Grid Middleware-Agnostic Trust Specification, the objective is to formulate an XML schema that captures the trust requirements for a grid service. Those are being abstracted to the user as a set of attributes along with their types and associated value ranges. The XML schema is used to instantiate valid XML trust requirements documents for a Grid project. Attributes that could be specified by the user include administrator certification, host site information, security level, proximity to local site, uptime, job failures, and hardware profile, and all of them are irrespective of the underlying Grid middleware. Two different methods are provided to help the user supply the trust requirements. For the Grid novice e-scientist, a wizard is available and for the Grid-aware e-scientist, a multi-paged editor is available. In GUTS, the wizard concept is used to guide the user in creating a trust requirements document and it consists of both required and optional dialogs. Similarly, the GUTS multi-page editor consists of tabs performing the same task - the exact number of tabs depends on the XML schema and on the way the set of attributes can conceptually be grouped together.

Proceeding with the Grid Middleware-Dependent Trust Specification component, those trust requirements specified in the previous component are translated and mapped into specific requirements that could actually be evaluated, based on the information supplied or deduced by the specific Grid middleware. The GUTS framework supports a specific Grid infrastructure/middleware only in the case where plug-ins for that specific Grid middleware are available. GUTS plug-ins and abstract interfaces will be accessible to the developer for extension as to support new Grid middleware.

Trust is not useful unless it becomes part of the decision process. In the case of the Grid, an end-user could utilize trust knowledge to choose the most suitable site for the specific job. An important aspect though is the presentation of trust results to the user. The final component is the trust management component, which is responsible for not only managing the project trust profiles that are stored in a database but also for presenting the user with a ranking list that could serve multiple purposes such as becoming a decisive factor when choosing a site to submit a job and provide the current "trustworthiness" of the various Grid sites that are available to the e-scientist. In addition, the user could also initiate to view past ranking lists as well as generate list where the rank over time for a specific site is provided.

It has been demonstrated in [9] that conceptually GUTS could be integrated with g-Eclipse client [13] that supports Grid/Cloud middleware like gLite, GRIA, and Amazon Web Services.

IV. sGUTS: Simplified GUTS

GUTS allows a user to specify the project needs by assuming that the user is knowledgable when it comes to technical specifications. However, this shouldn't be expected and instead of having the user profiling the grid service, it is wiser to have GUTS profile the project and map the profile into a set of trust requirements that the service needs to fulfill. This section presents sGUTS, an extension to GUTS, that abstracts the service requirements, and in consequent offloads the user from technical jargon and assists in a more appropriate site selection.

A. Diversity of Service Requirements for Grid Projects

The popularity of grid computing mainly lies on the emerging needs of scientists to process and store vast amount of heterogeneous data and at the same time increase collaborations among laboratories and research institutions, not necessarily in the same geographical location. Nevertheless, the changing scale and scope of science should not have an impact on a very simple premise: A scientist wants to do science (or e-science) and not computer science. The user interface that serves as the gateway to the grid infrastructure resources must be simplistic and intuitive for the average e-scientist.

It has been observed that grid failures or security incidents occur due to misconfigurations, with the source of these often being the lack of technical knowledge by the user. The service provider overwhelms an e-scientist with technical jargon, resulting in either specifying too strict requirements or too few. Depending on the nature of the experiment, the service requirements can greatly vary. Below, are examples illustrating the aforementioned claim.

1) Scenario 1: Molecular Modeling for Drug Discovery Experiment: Drug design using molecular modeling techniques, or molecular docking, helps scientists to predict how small molecules bind to an enzyme or a protein receptor. It is both a computationally and data intensive process to dock each molecule in the target chemical database. The grid infrastructure could facilitate the parallel and distributed processing of molecular docking. The average e-scientist would expect high computational accuracy from the site resources and data communication integrity during the data traversal of the communication network. If primary data were to be stored in the grid, then the storage has to be reliable with a very low possibility of loss of data. The classical requirements on databases, such as durability, consistency, reliability, scalability are needed for critical experiments. On the other hand, performance is not a primary concern as the focus is on correct and reliable results.

 ISBN: 978-1-61208-201-1

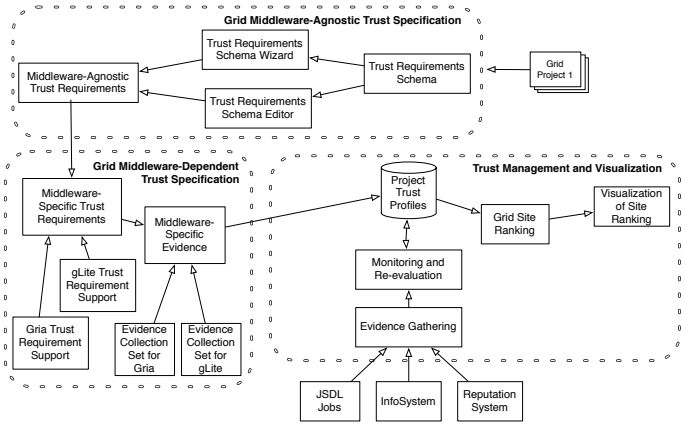


Figure 1. GUTS Framework

2) Scenario 2: Environmental Phenomena Prediction Task: Real time data about environmental phenomena could be processed, modeled, and correlated to predict natural disasters, leading into an early warning system. Grid computing could facilitate such a system, having appropriate sensors integrated with the underlying infrastructure at various locations, leading into the collection and distribution of the measurements to applications that use them to make predictions. Such an application will impose soft real time delivery on data: it is essential to use fresh data. However, data loss could be acceptable as prediction algorithms usually operate on incomplete data sets. Furthermore, the nature of the data does not justify any confidentiality or access control restrictions as the data is public information.

3) Scenario 3: Training Event: Grid federations or funded grid projects often offer initial training events for end-users. An induction course on grid technologies usually aims at demonstrating the capabilities of the underlying grid infrastructure. A successful event heavily relies on the availability of the resources. A site under maintenance or a site that is down could disrupt the normal flow of the training. Thus, high availability is expected, without too much concern on other security and QoS parameters. Jobs that are submitted during a training or educational experience could tolerate slower execution than normal or unencrypted traffic or even non-authenticated provider. Data loss is acceptable as well.

B. Abstraction of Service Requirements

The service requirements for a project depend on its nature, as demonstrated above. Table I illustrates the space (note: this space is not exhaustive, as it is a work in progress to derive a comprehensive list) of the requirements that must be imposed on the grid service for a successful project outcome. Each requirement could be further partitioned into a set of attributes. For example, physical properties could consist of storage room specifications and room temperature whereas confidentiality could be comprised of encryption algorithms and key lengths.

Table I
SERVICE REQS FOR GRID PROJECT

Requirements	Possible Attributes
Security	
authentication	(username+password), (X509 cert.), (biometrics)
integrity	(digital signing algorithm), (key length)
confidentiality	(encryption algorithm) (key length) ((a)symmetric)
availability	(uptime), (downtime frequency)
access control	(ACL), (RBAC), (authentication token)
privacy	(sensitive), (public), (ACL for data)
Behavioral	
Competence	(job failures), (administrator certification)
Motivation	(sysadmin: student, staff, faculty), (host site type)
Physical Info	(hardware/server-room profile), (location)
QoS	
Latency	(proximity), (site infrastructure)
Bandwidth	(site infrastructure), (country infrastructure)
Comp. Accuracy	(hardware/server-room profile)
Database Storage	(RAID), (hardware/server-room profile)

C. Deducing Service Requirements from Coarse-Grained Specifications

The idea is to abstract the process of explicitly specifying all these properties, a task that is carried out by the user. For example, the e-scientist of Scenario 2 may select to encrypt data without knowing that encryption is a costly operation unnecessary for the project needs, that could affect the latency of the received data. Therefore, the vision is to automatically populate the entries of Table I based on the responses that the user will supply to GUTS regarding high-level coarse-grained specifications of the desired

 ISBN: 978-1-61208-201-1

project. This trust project profile, in turn, will be mapped against existing grid services specifications that are ranked according to the level of matching. Similar to GUTS, the user will be given the opportunity to edit the generated trust profile using a multi-paged editor.

The questions below are forming the coarse-grained specifications of a project, and these are grouped into three categories, namely Project Needs, Data Needs, and Computational Needs. Based on the answers that the user provides, a project trust profile is created, and thus a set of service requirements.

1) Project Needs: The first category of questions is directly related to the overall needs of the project. Depending on the user responses, some of the questions in the other two categories will either not be asked or the answer options may be reduced. The questions of this category are the following:

Q1.1: *Is your project computational intensive or data intensive or an equal share of both?*
This will help the system decide what tradeoffs to apply when the choice is between data and computational needs.

Q1.2: *Is the computation more important than the storage of the data or an equal share of both?*
Even though it is a computational intensive project the e-scientist may be more concerned abut how the result is stored than how the computation was performed.

Q1.3: *What is the expected life-time of the resulting data of the project?*
The lifetime of the project may influence where the resulting data should be stored.

Q1.4: *By whom the results of the project will be used for?*
The usage of the results will guide the system to decide which security mechanisms will be needed. The user will choose one of the following predefined choices:

- Single user (me)
- Small group size (<10)
- Medium group size (<100)
- Large group size (>99)
- Public access, anyone can access the result

2) Data Needs: The second category prompts the user to provide input regarding the needs of handling the data related to the project. This includes both the input data and any derived results from computations. The questions are used to profile the requirements on the reliability, integrity, and privacy for the project data.

Q2.1: *Is the provenance of the stored data of the project necessary?*
The provenance is important for certain applications in proving that the data has not been tampered with/alternated.

Q2.2: *Where would you prefer the data to be stored?*
Due to the nature of the data, local government laws may prohibit export of the data in another country. Furthermore, the e-scientist may wish to store the data close to the local site. The predefined choices are:

- Close to my site
- Preferable in my country/region
- Location is not important

Q2.3: *What would the consequences of data loss be?*
Depending on whether or not the input data is primary data or derived data, the loss of the resulting data can greatly vary. Similarly, if the computation cost of deriving the data is extremely high, the cost of recomputing it may not be feasible. The predefined choices for this question are.

- Danger for loss of life (the user should be warned that the Grid environment may not be the most appropriate service provider in this case)
- Scientific findings may be lost (forever)
- Loss of investments made by doing the study
- Inconvenience of having to rerun the computation
- No loss due to nature of the data

Q2.4: *What would be the consequences of any modification (accidental or deliberate) to the stored data?*
Similar to Q2.3, except that the data is not lost but modifications have occurred. The choices that the user is presented with are the same as Q2.3.

Q2.5: *Does the input/resulting data contain any sensitive information?*
This question will be split into two questions, one for the input data and one for the resulting data. This question will define what are the privacy needs with regard to the data of the project:

- Highly sensitive (lose of life may result if leaking occurs)
- Sensitive (personal privacy laws apply)
- Moderate (lose of business secrets)
- Low (prefer to keep the data secret)
- N/A (publicly available data)

3) Computational Needs: The third category captures the computational needs for the project, and to be more specific its performance, reliability, and integrity requirements.

Q3.1: *What are the consequences of missed deadline of the completion of the computation?*
Deadlines can be missed in case of server failure, power failure, sever room overheating, server miss configuration, or accidental shutdown of the site. The user will choose one of the following predefined choices:

- Danger for loss of life

 ISBN: 978-1-61208-201-1

- Scientific findings may be lost (forever)
- Loss of investments made by doing the study
- Inconvenience of having to rerun the computation
- No consequence due to nature of the project

Q3.2: *What the delay of receiving computational data result in?*
This question is similar to Q3.1, except that here we are referring to the case where the initial computation service fails and the complete computation will have to be redone on other resources. The choices that the user can pick from are the same as those for Q3.1.

Q3.3: *What would be the consequences of any modification (accidental or deliberate) to the computation?*
Most Grid computational services have *server class* infrastructure (including hardware and server room), but not all. It could be the case that bit-flipping could not be detected, hence what would be the consequences of such problems. The choices that the user can pick from are the same as those for Q3.1.

4) Demonstration of Service Requirements Mapping from Specifications: It is beyond the scope of the paper to present the actual workings of the mapping, as the objective is to present a conceptual proof-of-concept of the usefulness of such a mechanism. Table II illustrates how user responses get translated to specific service requirements for Scenario 1. In this scenario, the user is a university researcher analyzing data to discover a cure for a disease, hence there is no need for secrecy of results but accuracy is vital for the success of the project.

Table II
SERVICE REQUIREMENTS FOR SCENARIO 1

	Attributes	**User Response**
Security Req.		
authentication	(X509 certificate)	(default)
integrity	(MD5)	Q3.3 - 4th option
confidentiality	(N/A)	Q1.4 - 5th option
availability	(best effort)	Q3.1, Q3.2 - 4th option
access control	(public)	Q1.4 - 5th option
privacy	(public)	Q1.4 - 5th option
Behavioral Req.		
Competence	(N/A)	Q2.3, Q3.1- 4th option
Motivation	(N/A)	Q2.3, Q3.1- 4th option
Physical Properties	(high quality)	Q3.3 - 4th option
QoS Req.		
Latency	(N/A)	Q3.1, Q3.2 - 4th option
Bandwidth	(N/A)	Q3.1, Q3.2 - 4th option
Comp. Accuracy	(high quality)	Q1.2 - computational
Database Storage	(average quality)	Q2.3, Q2.4 - 4th option

V. CONCLUSION

This paper presented sGUTS, an extension of GUTS, that supports automatic generation of service requirements for a grid project based on user responses on a set of predefined coarse-grained questions that capture the project nature and its data and computational needs. In this way, the average e-scientist does not need to be knowledgeable on technical grid details as the system maps his/her answers to service requirements that are further used to select the most appropriate grid service to satisfy the project at hand.

Even though research efforts exist for managing trust in the grid environment, still the focus is on how trust is perceived by the system (or site administrator) rather than on attempting to simplify the interpretation of trust for the end user. In the proposed conceptual framework, the center of the trust management process is the user who decides and specifies the needs of his/her project, which in turn are mapped to trust requirements. The proposed techniques leverage the functionality of the trust management system to include user input in an intuitive manner. The research effort presented in this paper is still under development.

As far as future directions are concerned, the applicability of sGUTS in cloud services will be investigated. In these settings, there is an additional constraint, which is the budget allocated for the project, that must be utilized in an efficient way and at the same time fulfill the aforementioned service requirements.

REFERENCES

[1] Mehran Ahsant, Mike Surridge, Thomas Leonard, Ananth Krishna, and Olle Mulmo. Dynamic trust federation in grids. In Ketil Stølen, William H. Winsborough, Fabio Martinelli, and Fabio Massacci, editors, *iTrust*, volume 3986 of *Lecture Notes in Computer Science*, pages 3–18. Springer, 2006.

[2] Alvaro Arenas, Benjamin Aziz, and Gheorghe Cosmin Silaghi. Reputation management in grid-based virtual organisations. In *International Conference on Security and Cryptography (SECRYPT08)*, pages 538–545, 2008.

[3] Alvaro Arenas, Michael Wilson, and Brian Matthews. On trust management in grids. In *Autonomics '07: Proceedings of the 1st international conference on Autonomic computing and communication systems*, pages 1–7, ICST, Brussels, Belgium, Belgium, 2007. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[4] B. Ashrafijoo, A. Habibizad Navin, M.-K. Mir Nia, S. Abedini, and N. Azari. Trust management in grid computing systems based on probability theory. In *Education Technology and Computer (ICETC), 2010 2nd International Conference on*, volume 4, pages V4–316 –V4–320, June 2010.

[5] Benjamin Aziz, Alvaro Arenas, Fabio Martinelli, Paolo Mori, and Marinella Petrocchi. *Trust Modeling and Management in Digital Environments: from Social Concept to System Development*, chapter Trust Management for Grid Systems, pages 149 – 178. IGI Global, 2010.

[6] J. Basney, W. Nejdl, D. Olmedilla, V. Welch, and M. Winslett. Negotiating trust on the grid. In *In 2nd WWW Workshop on Semantics in P2P and Grid Computing*, pages 1–20, 2004.

 ISBN: 978-1-61208-201-1

[7] José de R. P. Braga, Jr, Alexandre C. T. Vidal, Fabio Kon, and Marcelo Finger. Trust in large-scale computational grids: an spki/sdsi extension for representing opinion. In *MCG '06: Proceedings of the 4th international workshop on Middleware for grid computing*, pages 7–12, New York, NY, USA, 2006. ACM.

[8] Ioanna Dionysiou, Harald Gjermundrød, and David E. Bakken. An initial approach for adaptive trust in grid environments. In *Proceedings of 1st Workshop on Computational Trust for Self-Adaptive Systems (SELFTRUST'09)*, pages 719–722, Athens, Greece, November 2009.

[9] Ioanna Dionysiou, Harald Gjermundrød, and David E. Bakken. Guts: A framework for adaptive and configureable grid user trust service. In *6th International Workshop on Security and Trust Management (STM 2010)*, pages 84–99, Athens, Greece, September 2010.

[10] Ian Foster. What is the grid? - a three point checklist. *GRIDtoday*, 1(6), July 2002.

[11] Ian Foster and Carl Kesselman. The globus toolkit. In Ian Foster and Carl Kesselman, editors, *The Grid: Blueprint for a New Computing Infrastructure*, pages 259–278. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[12] Ian T. Foster. The anatomy of the grid: Enabling scalable virtual organizations. In *Euro-Par '01: Proceedings of the 7th International Euro-Par Conference Manchester on Parallel Processing*, pages 1–4, London, UK, 2001. Springer-Verlag.

[13] H. Gjermundrod, M. D.Dikaiakos, M. Stuempert, P. Wolniewicz, and H. Kornmayer. An integrated framework to access and maintain grid resources. In *Proceedings of the 9th IEEE/ACM International Conference on Grid Computing (Grid 2008)*, pages 57–64, 2008.

[14] Tyrene Grandison and Morris Sloman. A survey of trust in internet applications. *IEEE Communications Surveys and Tutorials*, 3(4):2–16, 2000.

[15] Marty Humphrey and Mary R. Thompson. Security implications of typical grid computing usage scenarios. *Cluster Computing*, 5(3):257–264, 2002.

[16] P.D. Manuel, S. Thamarai Selvi, and M.I.A.-E. Barr. Trust management system for grid and cloud resources. In *Advanced Computing, 2009. ICAC 2009. First International Conference on*, pages 176 –181, Dec. 2009.

[17] Dilal Miah. A matter of trust: enabling grid security through bilateral negotiation. International Science Grid this week (ISGTW), 2008. http://www.isgtw.org/?pid=1001540, last accessed April 6, 2012.

[18] K. Neokleous, M. D. Dikaiakos, P. Fragopoulou, and E. Markatos. Failure management in grids: The case of the egee infrastructure. *Parallel Processing Letters*, 17(4):391–410, 2007.

[19] T.B. Quillinan, B.C. Clayton, and S.N. Foley. Gridadmin: decentralising grid administration using trust management. In *Parallel and Distributed Computing, 2004. Third International Symposium on/Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks, 2004. Third International Workshop on*, pages 184 – 192, July 2004.

[20] Kai Wei and Shaohua Tang. A cloud-based recommendatory trust processing method for trust management system of grid. In *Communications and Mobile Computing (CMC), 2010 International Conference on*, volume 1, pages 289 –293, April 2010.

[21] Von Welch, Frank Siebenlist, Ian Foster, John Bresnahan, Karl Czajkowski, Jarek Gawor, Carl Kesselman, Sam Meder, Laura Pearlman, and Steven Tuecke. Security for grid services. In *HPDC '03: Proceedings of the 12th IEEE International Symposium on High Performance Distributed Computing*, pages 48–57, Washington, DC, USA, 2003. IEEE Computer Society.

[22] Chern Har Yew and H. Lutfiyya. A middleware-based approach to supporting trust-based service selection. In *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, pages 407 –414, May 2011.

 ISBN: 978-1-61208-201-1

A Trust-based DRM Scheme for Content Sharing in an Open Environment

Qin Qiu, Zhi Tang, Yinyan Yu
Institute of Computer Science and Technology
Peking University
Beijing, China
e-mail: {qiuqin, tangzhi, yuyinyan}@pku.edu.cn

***Abstract*—Interpersonal or inter-organizational content sharing is a popular activity for casual or cooperation purposes. On one hand, content sharing is turning more and more open for better outcome or stronger influence; on the other hand, it is important to protect shared sensitive content from being misused or disclosed to malicious users. To secure content sharing in open environment, this paper proposes a DRM scheme built upon a trust model. With the proposed scheme, secure content sharing is open to all trusted content users, and user authentication and authorization can be performed autonomously by content owners. Experiments and comparisons indicate that the proposed scheme achieves satisfactory security and usability.**

Keywords-DRM; trust model; content sharing

I. INTRODUCTION

With the popularization of electronic devices and the development of Internet, lots of digital content are created and shared among individuals or organizations for casual or cooperation purposes. Examples of such open sharing include:

- Alice creates an original work and shares it with her friend Bob, expecting Bob or Bob's friends to offer some advices for improvement;
- Organization A cooperates with partner organization B on an innovation, and allows other unknown but eligible organizations to join for better outcome.

On one hand, the content owner may want the content to be shared with more users for better cooperation outcome or enlarge the influence; on the other hand, to preserve rights or interests, the content owner needs to have control on who and how to use the content.

Digital Rights Management (DRM), which achieves persistent content protection in the whole life-cycle of digital content and controls how digital content may be used [1], is a desirable solution to protect the shared content. However, existing DRM schemes serve for closed systems and depend on Trusted Authority (TA), who has priori-knowledge of all content users, to authenticate users and issue licenses [1-4]. The dependence on TA hinders existing DRM schemes from being applied into secure content sharing in open environment: firstly, it is impossible for TA to supervise all potential content users in an open environment; secondly, content owners may be reluctant to have their authorization information in the charge of a third party for privacy concerns. Therefore, it is important to enable autonomous rights management by content owners and provide a mechanism for content owners to evaluate the eligibility of potential content users in an open environment.

Social trust is a belief in the honesty, integrity and reliability of others; it is the basic environmental factor of content sharing [5]. Because the danger of being misquoted or discovering that the shared content has been used for underhanded or unsavory purposes is always there, before one shares important content, there is an assumed understanding of trust that the content will be used only for the good [6]. For example, Alice shares her original work with Bob on condition that Bob is trusted not to plagiarize innovations in the work and publish a similar work in advance. Content sharing in an open environment, which assumes that anyone may be a potential participant, is inherently a social activity. Establishing of trust in this context inevitably requires some form of social computing supported by a trust model [7].

To achieve secure content sharing in an open environment, we model social trust between content owners and content users, and propose a decentralized DRM scheme in this paper. The trust model enables content sharing with unknown content users, and eliminates the necessity of TA; authentication and authorization are performed autonomously by content owners. To reduce interaction and authorization overheads on content owners, contents are organized into groups, and a batch authorization method based on key derivation mechanism is integrated into the DRM scheme.

II. RELATED WORK

A. DRM Schemes

Most existing DRM systems, such as Microsoft Windows Media Rights Manager and InterTrust Rights|System are set up to preserve the commercial profits of content providers. In those systems, License Server that is trusted by content providers is indispensable; it records transaction information and issues content users licenses for requested content [1-3]. Sometimes external Certificate Authority (CA) is also needed for identity authentication and certificate issuance [4]. Content users are only consumers of the protected content.

A few DRM schemes have been presented to secure content sharing; however, those systems are for closed systems where all content users are pre-known to content

 ISBN: 978-1-61208-201-1

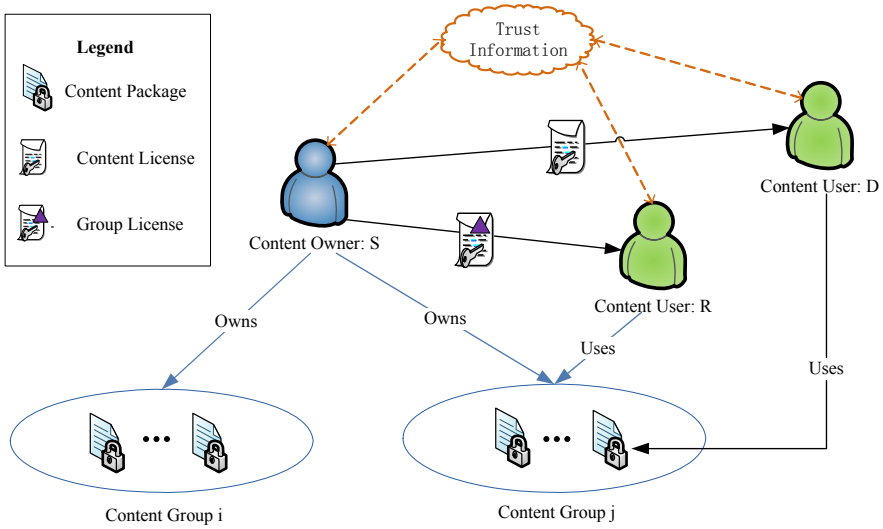


Figure 1. The model of our DRM system

owners and the system. Microsoft IRM [8] and Voltage SecureFile [9] need a trusted server for user enrollment, authentication and license issue according to content encryption keys and permission lists from content owners. The problem is that the server is able, though not bound, to decrypt content owners' secret contents and grasp the relations among content owners, content users and shared contents. Bhatt et al. [3] proposed a personal DRM manager for content sharing between smart phones, which works on the assumption that each participating smart phone holds a certificate issued by CA. Feng [10] proposed a decentralized copy protection solution, but it requests that there is pre-established trust relationship between content owners and content users.

In short, there is hardly any DRM solution for secure content sharing in an open environment.

B. *Trust Model*

Trust modeling was first proposed by Marsh [5] to assist decision making in distributed artificial intelligence systems. Till today, many trust models have been presented in areas of public key authentication [11], ubiquitous computing [12], and distributed network [13, 17]. In trust models, trust is generally regarded to be non-symmetrical (the fact that A trusts B does not indicate that B trusts A), and conditionally transitive (the fact that A trusts B and B trusts C does not indicate that A trusts C unless certain conditions are satisfied) [5, 12].

There are three basic types of trust in a trust model [11-17]:

- *Direct trust* reflects the trustor's judgment on the trustworthiness of an acquainted entity, without intervention of third parties.
- *Confidence of recommendation* represents the trustor's confidence in an entity to provide accurate recommendations.
- *Indirect trust* in an unknown entity is built through recommendations from those that have trust in the recommended one. By performing some evaluation on the recommendations, the trustor can make judgment on the trustworthiness of the recommended entity.

Trust context is considered in some trust models. Abdul-Rahman [14] uses trust category to express particular semantic of trust, so that the model can be used in different applications. Ray [16] uses a set of keywords with equivalent semantics to represent context, so that trust relationships in same or similar contexts can be compared.

To our best knowledge, no trust model has been presented or applied in the area of DRM. Some researchers proposed all-purpose trust models [14, 17], but they are not so suitable to be directly used in our DRM scheme. In Rahman's trust model [14], users can only claim what the trust is about in the one-dimension context, not able to clarify more complex information like trust conditions or constraints; Liu's trust model [17] allows users to self-define trust contexts through XML schema, which is cumbersome and difficult to adopt. To be applied in DRM, a trust model has to be aware of context information related with user authentication and content authorization. We present a tailored trust model with contexts about trust types, constraints, and objects in Section IV.

III. SYSTEM MODEL

In our scheme, secure content sharing progresses between Content Owner and Content User in client-to-client communication model and no TA is involved in the system. DRM agent of Content Owner and Content User manages trust information, content information and authorization information.

Fig. 1 shows the model of our scheme. As the existence of social trust is the premise of authorization, the general process of content sharing is as follows:

 ISBN: 978-1-61208-201-1

TABLE I. NOTATIONS

Notation	Description
//	or, connecting alternative items
UID_i	user identifier
TD(i,j,c)	user i's trust degree in user j under context c
CoR(i,j,c)	user i's confidence degree in user j's recommendations under context c
$VT_i(c)$	Validity Threshold set by user i under context c
PU_i; PR_i	public key; private key
Kd(•,•)	key derivation function
Sig_i	signature on message digest
PEnc(pu,•)	asymmetric encryption with key pu
Enc(k,•)	symmetric encryption with key k
H(•)	Hash function

(1). Content Owner establishes Sharing Trust with Content User. If the owner has knowledge about the user, the establishment can be directly completed by the owner; if the user is unknown, the owner can establish indirect trust with the user based on recommendations from other Content Users.

(2). After establishing Sharing Trust, Content Owner issues a license to Content User. In batch authorization mode, the license is Group License, with which Content User is able to use any content that is or will be categorized into the content group.

The notations in Table I are used throughout this paper.

IV. THE UNDERLYING TRUST MODEL

To be applied in open sharing environment, a distributed trust model is built in DRM agent. In this section, we describe how we model social trust between system users for DRM application.

A. Trust Representation and Decision

Our trust model is context-sensitive. For a trustor, her trust relationship towards a trustee is defined as below:

$$Trust(UID_{trustee}, context)=\{TD, CoR\}; \quad (1)$$

context=<trustCategory, trustConstraint>.

- $UID_{trustee}$ is the user identifier of the trustee.
- context is a feature vector providing background information including trust category and trust constraint. While trustCategory is used to discriminate different kinds of trust involved in DRM application, trustConstraint limits the range that the trust is valid in.
- TD is trustor's trust (either direct trust or indirect trust) degree in the trustee under the specified context. CoR is the trustor's confidence degree in the trustee's recommendations under the specified context. Being fuzzy logics, both TD and CoR are continuous variables in the interval of [0,1]. 0 indicates lowest degree of trust or confidence, while 1 indicates highest.

According to the contexts involved in the DRM system, we have two categories of trust: Key Trust and Sharing Trust.

Procedure: TrustPro(*source*, *dest*, *type***)**

rslt←0, *j*←0.
if there is a direct trust path from *source* to *dest* **then**
  if *type*=1 **then**
    rslt←TD(*source*, *dest*)
  else if *type*=2 **then**
    rslt←CoR(source, dest)
  end if
else *n*←the number of recommendation paths from *source* to *dest*
  if *n*>=1 **then**
    for every recommendation path *i*<=*n* **do**
      source finds recommender R_i that has direct trust path to *dest*
      CoR(*source*, R_i)← **TrustPro(***source*, R_i, 2**)**
      if CoR(*source*, R_i)>VT **then**
        j←*j*+1;
        if *type*=1 **then**
          $rslt_j$←CoR(*source*, R_i)*TD(R_i, *dest*) ----(2)
        else if *type*=2 **then**
          $rslt_j$←CoR(*source*, R_i)*CoR(R_i, *dest*) ----(3)
        end if
      end if
    end for
    N←*j*
    rslt←average of $rslt_k$, where *k*=1,2,⋯,*N* ----(4)
  end if
end if
return rslt

Figure 2. Procedure for trust propagation

- Key Trust (KT) is the trust in authenticity of the binding between the trustee and the claimed public key. It provides foundation for user authentication. A trustor's Key Trust towards a trustee can be described as $Trust(UID_{trustee},<KT>)$. Here, trustConstraint is set void.
- Sharing Trust (ST) is the trust in eligibility of the trustee to share the content. It provides foundation for user authorization. A trustor's Sharing Trust towards a trustee can be described as $Trust(UID_{trustee},<ST,CID//GID>)$. Here, trustConstraint is a content identifier CID or a content group identifier GID; it confines the range of contents that the trustee is trusted to share.

We use Validity Threshold (VT) to map TD and CoR to valid or invalid states. VT is a continuous variable in the open interval of (0,1). It is adjustable by system users according to specific contexts and security policies. For example, if Content Owner S expects only very trustworthy entities to share sensitive content in group GID, she can set a high value for $VT_S(<ST,GID>)$.

B. Trust Propagation

A trustor's trust relationship with other entities can be regarded as a directed graph. With recommendations from different recommenders, multiple recommendation paths connecting the trustor to the target entity are built. The trustor propagates trust along all paths to evaluate the trust degree in the target entity [5, 12-17].

The procedure of our trust propagation is in Fig. 2. There are three input parameters, and the procedure outputs the trust propagation result. The input parameter *source* is the

 ISBN: 978-1-61208-201-1

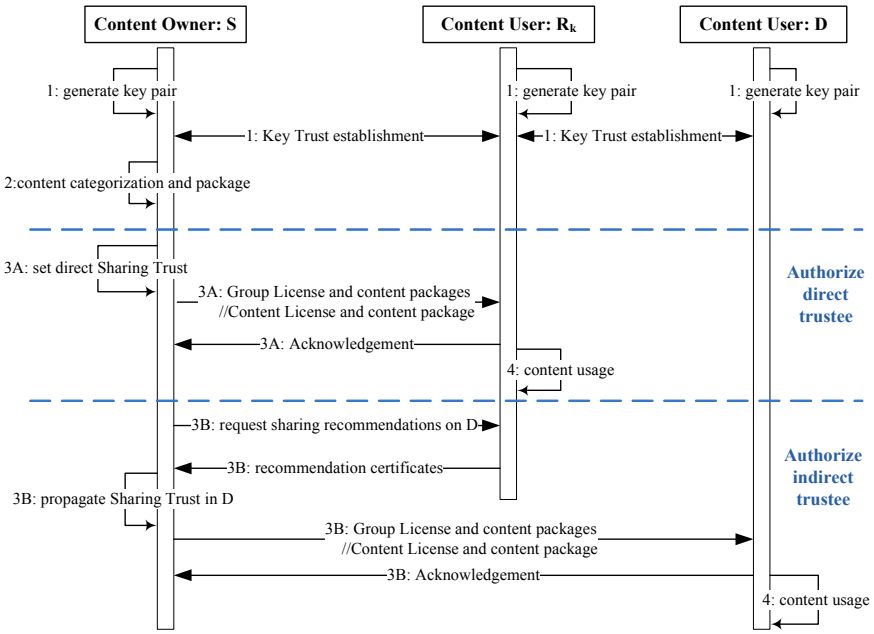


Figure 3. Overview of operations and protocols

trustor's user identity, and *dest* is the target trustee's user identity. When input parameter *type* is set "1", the procedure outputs TD(*source*, *dest)*; when *type* is set "2", the procedure outputs CoR(*source*, *dest*).

In Fig. 2, we use expression (2), (3) and (4) for trust propagation because they conform to both Weighted Average Operator in D-S theory and Consensus Operator in Subjective Logic [15, 18]. To avoid the problem of opinion dependence [15], the procedure only considers the direct trust path and ignores all recommendation paths when an entity has direct trust in the target trustee. With the maximal length of recommendation paths limited with a reasonable constant, the complexity of the procedure is O(n), where n is the scale of valid recommenders.

V. OPERATIONS AND PROTOCOLS

This section describes how our trust based scheme works in enabling secure content sharing in open environment. As illustrated in Fig. 3, the whole process consists of four phases: initialization, content categorization and package, content authorization, and content usage.

A. Phase 1-Initialization

To begin with, system user S sets a unique user identifier UID_S through her DRM agent. The agent generates a public-private key pair $\{PU_S, PR_S\}$ for S.

Next, S establishes Key Trust and exchanges public keys with others. Firstly, with secure communication or auxiliary verification methods, S gets legal public keys of some friends R_i, i=1,2,…, and establishes direct Key Trust with them. When S needs the public key of some unknown system user D, S requests friends for recommendations. If a friend R_i, $\forall i$, has direct Key Trust in D, R_i returns S a recommendation containing UID_D, PU_D and TD(R_i,D,<KT>); otherwise, R_i forwards the request to the next hop. Finally, S's DRM agent performs trust propagation on all the received recommendations. If the result is a valid trust value, S successfully builds Key Trust with D and saves PU_D.

B. Phase 2-Content Categorization and Package

Content Owner S sets up some content groups, each of which is assigned a unique group identifier GID and a random secret key GK . All contents in a content group have some identical properties, and target same Content Users.

When S needs to protect some content, S firstly categorizes it into a content group GID, assigns it a content identifier CID, and then derives content encryption key CEK from GK and CID with a key derivation function that satisfies one-way and randomness [19,20]. Next, S encrypts content plaintext M, and packages the cipher text with GID, CID and signature. CP can be distributed to Content Users in any way at any time.

S: CEK=kd(CID, GK)
C=Enc (CEK, M)
CP= {UID_S, GID, CID, C, Sig_S}

C. Phase 3-Content Authorization

In an open environment, content sharing may happen not only between friends, but also between strangers. In this phase, Content Owner first establishes direct or indirect Sharing Trust with Content Users, and then performs authorization for them. According to the authorization mode, there are two kinds of licenses:

- Group Licenses are issued for Content Users to use all contents that are or will be categorized to the corresponding content group.
- Content Licenses are issued for Content Users to use only the prescribed content.

 ISBN: 978-1-61208-201-1

We first describe authorization for directly trusted Content Users, and then authorization for indirectly trusted Content Users.

1) Direct Trust Establishment and Authorization.

Suppose there is direct Sharing Trust from Content Owner S to Content User R, and they have stable content sharing relation. S sets GID as the range of contents that is ready for R to use, TD (S,R,<ST,GID>) as the trust degree in R to share the contents in group GID, and CoR (S,R,<ST,GID>) as the confidence degree in R to recommend other Content Users to share one or more contents in the group GID.

If TD(S,R,<ST,GID>) > VT_S(<ST,GID>), S deems R as an eligible content sharer of content group GID, and generates Group License $L_G(R)$ for R. To preserve R's privacy, authorization information is encrypted with system default key SysKey as ℓ.

S: ℓ=Enc(SysKey,{GID,RightsInfo})

S→R: $L_G(R)$={UID_S,UID_R, ℓ,PEnc(PU_R,GK),Sig_S}

After receiving $L_G(R)$, R collects ℓ and PEnc(PU_R,GK) from it, and then returns S an acknowledgement message AM.

R→S: AM={UID_R,UID_S,H(ℓ,PEnc(PU_R,GK)),Sig_R}

2) Indirect Trust Establishment and Authorization.

Suppose Content User D, who is unknown to S, wants to share content CID. As CID belongs to content group GID and S has no direct Sharing Trust in D, S sends Sharing Recommendation Request (SRR) to R_k (k=1,2,…) who are authorized Content Users of GID or CID. SRR contains the recommendation deadline τ, and a random number γ to prevent message replay. It should be noted that Content Owner only asks authorized Content Users for sharing recommendations, because only authorized Content Users can make proper judgment about whether the content can be shared by a candidate user.

S: α =Enc(SysKey,{CID,UID_D})

S→Rk: SRR={UID_S, α, τ, γ, Sig_S}

If having direct Sharing Trust in D, R_k returns S a Recommendation Certificate RecCert(R_k) with trust information encrypted to protect privacy.

R_k: β_k =Enc(SysKey,{CID,UID_D,TD(R_k,D,<ST,CID>)})

R_k→S: RecCert(R_k)={ UID_{R_k},UID_S, β_k,γ,Sig_{R_k} }

After the deadline τ, S's DRM agent verifies γ and recommenders' signatures in received recommendation certificates, and then propagates TD(S,D,<ST,CID>). If the result is larger than VT_S(<ST,CID>), S deems D to be an eligible Content User of CID, and generates Content License for D.

S: ℓ'=Enc(SysKey,{CID,RightsInfo'})

S→D: $L_C(D)$ = {UID_S, UID_D, ℓ', PEnc(PU_D,CEK), Sig_S}

After receiving $L_C(D)$, D collects ℓ' and PEnc(PU_D,CEK) from it, and then returns S an acknowledgement message AM'.

D→S: AM'={ UID_D,UID_S,H(ℓ',PEnc(PU_D,CEK)),Sig_D }

In another case, if D wants to share all contents belonging to GID, S propagates Shaing Trust in D with trust constraint GID; if the trust establishment is successful, S issues D a group license.

D. Phase 4-Content Usage

1) Usage with Group License. R's DRM agent first associates CP with $L_G(R)$ by checking whether GID in $L_G(R)$ and that in CP are identical, and then ensures that the issuer identifier in $L_G(R)$ and the owner identifier in CP are the same. After successful verification, R's DRM agent derives CEK with GK collected from $L_G(R)$ and CID collected from CP, and then decrypts the content cipher in CP.

2) Usage with Content License. D's DRM agent associates CP with $L_C(D)$ according to CID and the owner identifier; next, D's DRM agent directly obtains CEK by decrypting its cipher in $L_C(D)$, and then decrypts the content cipher in CP; finally, the agent manages content usage according to rights information in $L_C(D)$.

E. Revocation of Group License

Suppose R is a frequent Content User of contents in group GID owned by S, and has been issued a group license $L_G(R)$. When S wants to revoke $L_G(R)$, so that R cannot use contents categorized into group GID after the revocation, there are two methods.

1) Group Alteration: S builds a new content group GID' that is associated with GID, and issues Group Licenses corresponding to GID' to valid Content Users. New contents are categorized to GID' instead of GID.

2) Key Update: S updates the group key of GID to be GK', and issues an updated Group License containing the cipher of both GK and GK' to valid Content Users.

VI. SECURITY ANALYSIS

A. Robustness of the Trust Model

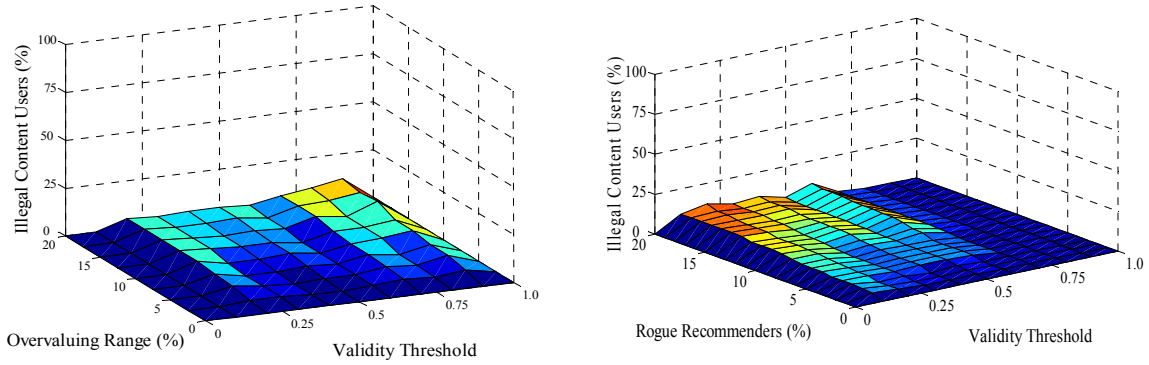
Illegal Content Users may be introduced in two ways: (1) a trustor overvalues trust degrees in trustees out of subjective faults, causing that trust degrees in some untrustworthy entities turn larger than VT mistakenly; (2) some rogue recommenders provide unfair positive recommendations for untrustworthy entities individually or collusively.

To test the robustness of our trust model, we simulated the above two ways in random trust networks; VT was set from 0 to 1 to observe its effects to the result. Shown in Fig. 4, the simulation results indicate that: (1) the proportions of illegal Content Users are in very low levels, and our trust model achieves satisfactory robustness; (2) setting a proper value for VT helps impede the appearance of illegal Content Users.

B. Proctection of User Privacy

There are mainly two kinds of privacy information involved in our scheme: authorization information in licenses and trust information in recommendation certificates. Both of them are encrypted with system keys and can only be decrypted by the DRM agent of the target receiver. Nobody

 ISBN: 978-1-61208-201-1



(a) (b)

Figure 4. The proportion of illegal Content Users caused by: (a) trust overvaluation simulations where trustors overvalue trust degrees in all their trustees with random scales within a range from 0% to 20%; (b) unfair positive recommendations simulation where random rogue recommenders (occupying from 0% to 20% of all recommenders) assigned the highest trust degree (i.e. 1) to all they recommend

else, except the message sender, knows the plaintext of the privacy information.

By requesting only one recommender for recommendations, a malicious trustor may infer the recommender's trust information from the result of trust propagation. However, such method is low-efficient and troublesome. It can hardly cause privacy concerns on a large scale of system users.

VII. PROTOTYPE IMPLEMENTATION

We have developed a prototype system of the proposed scheme to protect Microsoft Office Word 2007 documents and enable secure document sharing in an open lab. The system is composed of a desktop manager and a Microsoft Office Word 2007 plug-in. Through the desktop manager, users can manage personal information and trust relationships; through the plug-in, users can perform content protection and usage operations. The main User Interface (UI) of the prototype system is shown in Fig. 5.

In the prototype system, content packages, licenses and recommendation certificates are all described in XML files. The file size of a license is about 393 bytes. For a plaintext document with the size of 1124 kilobytes (KB), the encryption time is 35.692 milliseconds by Content Owner, and the decryption time of the corresponding content package is 3.392 milliseconds by Content User with a content license (tests were carried out on a PC with Pentium D CPU, 3.00GHz and 1.00GB RAM).

VIII. CONCLUSION

In this paper, we propose a DRM scheme to secure content sharing among those with direct or indirect social ties. A comparison of our scheme with related solutions is shown in Table II. Based on the social trust model, our scheme has the following advantages:

(1). It is independent of TA; authentication and authorization are performed by Content Owner autonomously.

(2). Unknown content users in open environment may participate in content sharing according to the result of trust evaluation.

(3). According to the constraint information of Sharing Trust, Content Owner can perform either content authorization or group authorization, which achieves a good balance between security and authorization efficiency.

In the underlying trust model, we consider the contexts of authentication and authorization, and allow system users to set different thresholds for trust decision in different scenarios. The trust propagation procedure combines some existing achievements in opinion combination [15, 18] and eliminates the problem of opinion dependence.

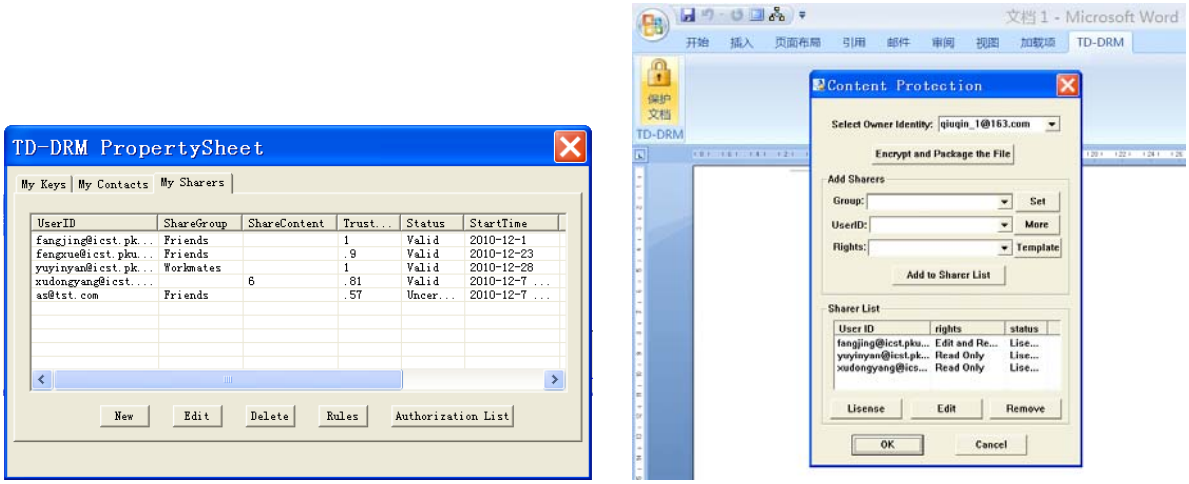
ACKNOWLEDGMENT

This work was supported by Major Scientific and Technological Project of GAPP in China (No. GXTC-CZ-1015004).

TABLE II. COMPARISON OF RELATED SCHEMES

	[21, 22, 23]	[8, 9]	[3, 10]	Our Scheme
Usage Scenario	Content retail	Content sharing	Content sharing	Content sharing
Autonomous protection	No	No	Yes	Yes
Supports open sharing	No	No	No	Yes
Authorization mode	Content based	Content based	Content based	Content based & Group based

 ISBN: 978-1-61208-201-1



(a) (b)

Figure 5. UI of the prototype system: (a) the desktop manager, and (b) the plug-in

REFERENCES

[1] J. S. Erickson, "Fair Use, DRM, and Trusted Computing," Communications of the ACM. U.S., vol. 46, no. 4, pp. 34-39, 2003.

[2] W. Ku and C. H. Chi, "Survey on the Technological Aspects of Digital Rights Management, " in ISC 2004. LNCS, vol. 3225, K. Zhang, and Y. Zheng, Eds. Heidelberg: Springer, 2004, pp. 391-403.

[3] S. Bhatt, R. Sion, and B. Carbunar, "A Personal Mobile DRM Manager for Smartphones," Computers & Security. U.S. issue 28, pp. 327-340, 2009.

[4] Z. Zhang, Q. Pei, and J. Ma, et al., "Security and Trust in Digital Rights Management: A Survey," International Journal of Network Security. Taiwan, vol. 9, no.3, pp. 247-263, 2009.

[5] S. P. Marsh, "Formalizing Trust as a Computational Concept," PhD Thesis, Department of Mathematics and Computer Science, University of Stirling, 1994.

[6] C. R. McInerney and S. Mohr, "Trust and knowledge sharing in organizations: Theory and Practice," RETHINKING KNOWLEDGE MANAGEMENT, Information Science and Knowledge Management. U.S., vol. 12, pp. 65-86, 2007.

[7] Jaehong Park, Yuan Cheng, and R. Sandhu, "towards a framework for cyber social status based trusted open collaboration," Proc. 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom), 2010, pp. 1-8.

[8] Microsoft IRM: http://office.microsoft.com/en-in/excel-help/information-rights-management-in-the-2007-microsoft-office-system-HA010102918.aspx [retrieved: January, 2011].

[9] Voltage SecureFile: http://www.voltage.com/products/sfclient.htm [retrieved: January, 2011].

[10] X. Feng, Q. Qiu, and Z. Tang, "Copy Protection for Email," Proc. 12th International Conference on Electronic Commerce, 2010, pp. 177-183.

[11] A.-A. Rahman, "The PGP Trust Model," The journal of Electronic Commerce, 1997.

[12] F. Almenarez, A. Marin, C. Campo, and G. R. Carlos, "PTM: A Pervasive Trust Management Model for Dynamic Open Environments," Proc. IFIP International Conference on Pervasive Computing and Communications, Pisa, Italy, 2006, pp. 267-271.

[13] Y. Sun, W. Yu, Z. Han, and K. J. Liu, "Information Theoretic Framework of Trust Modeling and Evaluation for Ad Hoc Networks," IEEE Journal on Selected Areas in Communications, special issue on security in wireless ad hoc networks, vol 24, no.2, pp. 305-317, 2006.

[14] A.-A. Rahman and S. Halles, "A Distributed Trust Model," Proc. 1997 workshop on new security paradigms, 1998, pp. 48-60.

[15] A. Josang, "An Algebra for Assessing Trust in Certification Chains," Proc. Network and Distributed Systems Security Symposium (NDSS), 1999, pp. 1-10.

[16] I. Ray, I. Ray, and S. Chakraborty, "An Interoperable Context-Sensitive Model of Trust," Journal of Intelligent Information Systems, vol 32(1), pp. 1-12, 2009.

[17] Z. Liu, S. S. Yau, D. Peng, and Y. Yin, "A Flexible Trust Model for Distributed Service Infrastructures," Proc. 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing, 2008, pp. 108-115.

[18] A. Josang and M. Daniel, "Strategies for Combining Conflicting Dogmatic Beliefs," Proc. 6th International Conference on Information Fusion, 2003, pp. 1133-1140.

[19] U. Blumenthal, N. C. Hien, and B. Wijnen, "Key Derivation for Network Management Applications," IEEE Network, pp. 26-29, May/June, 1997.

[20] M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies," ACM Transactions on Information and System Security (TISSEC), vol. 12, issue 3, Article 18, pp. 1-43, 2009.

[21] Architecture of Windows Media Rights Manager: http://www.microsoft.com/windows/windowsmedia/howto/articles/drmarchitecture.aspx [retrieved: May, 2011].

[22] Adobe Content Server: http://www.adobe.com/products/content-server.html [retrieved: April, 2011].

[23] Microsoft DAS: http://www.microsoft.com/reader/developers/info/das.aspx [retrieved: March, 2012].

 ISBN: 978-1-61208-201-1

A Gateway-based Access Control Scheme for Collaborative Clouds

Yongdong Wu, Vivy Suhendra, Huaqun Guo
Institute for Infocomm Research, 1 Fusionopolis Way, Connexis, Singapore
{wydong,vsuhendra,guohq}@i2r.a-star.edu.sg

***Abstract*—A collaborative cloud, formed by private enterprise clouds, is usually task-oriented and tightly correlated. It brings new ways to build and manage computing systems in terms of software development, resource sharing, and maintenance. However, there is little research on the security of collaborative clouds. This paper presents a virtual private cloud for collaborative clouds based on security-enhanced gateways. It enables users in each private cloud to access other private clouds in the collaboration transparently, dynamically and anonymously.**

***Keywords*-Virtual cloud computing; Identity management; Access control.**

I. INTRODUCTION

In the computing field, the requirements of cost, security and ease of use are conflicting. PC users have full control of their computers, but in return have to take full responsibility for software installation, patching-up, viruses, spyware, crashes, software and hardware upgrades. This makes the total maintenance cost very high. On the other hand, the low-cost NetPC (or Network PC), known as a thin client, is intended to be centrally managed and to function without diskette drive nor CD-ROM drive. All NetPC software and data are stored on a server and accessed over a private network from the NetPC box. Offering a trade-off between these two situations is the cloud computing paradigm[1], a system that provides services to customers at low cost [1].

In the cloud computing paradigm, a service provider builds the cloud infrastructure, and leases it to users with a "pay-as-you-go" business model. From the viewpoint of users, cloud computing has many merits such as "infinite" scalability, "always-on" availability, light-weight system maintenance[2], fast access to best-of-breed applications, and the potential to significantly reduce operating costs [2]. Thus, cloud computing is becoming one of the most important topics in the IT world, and the use of cloud computing services is an attractive opportunity for companies to improve their IT services. For instance, EMEA (Europe, Middle East and Africa) will spend $18.8 billion on cloud services provided by third-party suppliers in 2014 [3], while China will invest about $154 billion to develop cloud computing hubs. The South Korean government has also decided to invest $500m in cloud initiatives, and intended to raise $2 billion investment by 2014 [4].

Despite the value proposition that cloud computing has, its adoption has been slow due to issues of reliability, consistency, privacy, and federation, especially security issues [5]–[8]. For example, the security breach of Twitter and Vaserv.com (via a zero-day vulnerability) in 2010 and the data breach at Sony Corporation and Go-Grid in 2011 [9], compromising data of 100 million customers [10], have made it quite clear that stringent security measures need to be taken in order to ensure security and proper data control in the cloud. As IDC researchers indicated, "*Security was a long-term inhibitor to cloud adoption*" [11]. Although Cloud Security Alliance promoted the use of best practices for providing security assurance within cloud computing [12], it did not propose a concrete security solution for collaborative clouds, where security risk is amplified and accelerated by the potential spread of a security flaw from a compromised cloud to a collaborative peer.

A collaborative cloud is a cloud community which consists of private enterprise clouds. It comprises virtual computing resources dedicated to a particular collaborative activity (e.g., correlated intrusion analysis [13] or detection [14]), and is subject to information sharing policies that restrict the scope of information sharing within the cloud. Users in each private cloud is able to access the resources of other private clouds in the collaboration (henceforth termed *peer clouds*) in a controllable way. Additionally, as it is impossible to require that all cloud providers offer the same services, users in different clouds may exchange information via third-party platforms (e.g., Facebook). In all, a collaborative cloud is a task-oriented, high-access relationship.

This paper proposes a Virtual Private Cloud (VPC), similar to a virtual domain [15] in Grid computing, based on secure inter-connective cloud gateways. The VPC enables each user to perform authentication in its own cloud so as to obtain access to peer clouds anonymously. It also provides a secure channel for users in the virtual cloud to communicate with each other via a third-party platform.

The remainder of this paper is organized as follows. Section II elaborates the security structure in a VPC environment, particularly the authentication diagram. Section III discusses the security, property and implementation of the proposed VPC. Section IV presents the related work, and a conclusion is drawn in Section V.

[1]According to the definition from National Institute of Standards and Technology (NIST): Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, storage, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

[2]The end user may have to upgrade/patch some basic components such as the OS, browser, media decoders etc.

 ISBN: 978-1-61208-201-1

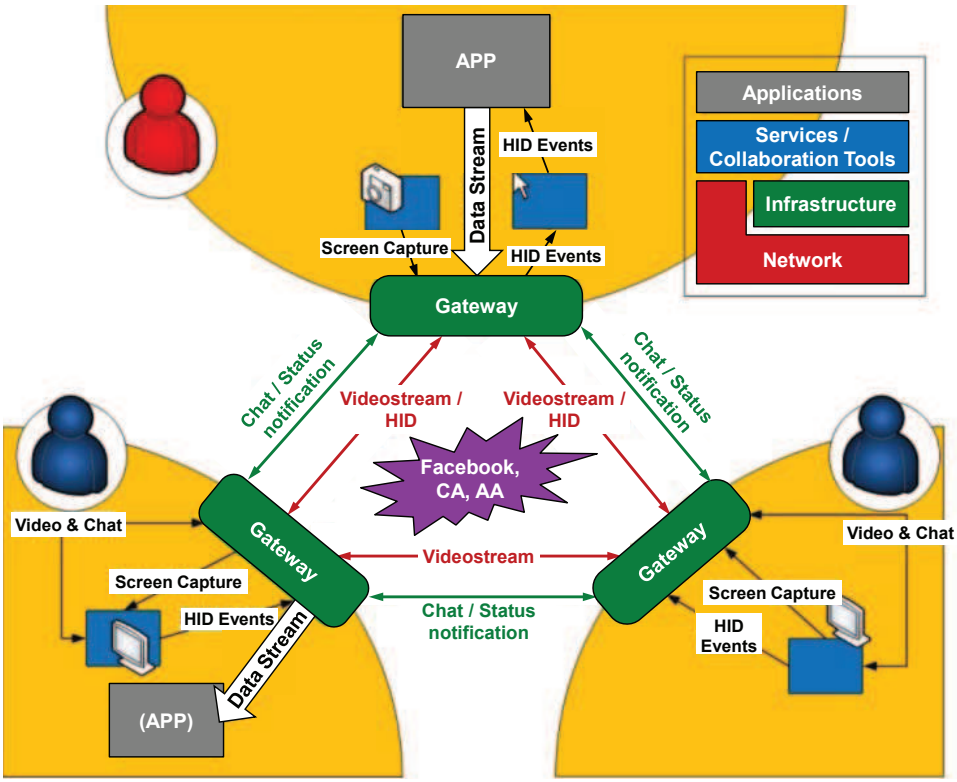


Fig. 1. Gateway-based architecture of VPC, adapted from [16].

II. Virtual Private Cloud

In this paper, we assume that each enterprise has its own private cloud. In order to complete a task, several enterprises will form a collaborative cloud so that they can share resources. As a collaborative cloud is task-oriented, users involved in the task form a virtual team. The team members are dynamic and anonymous to the peer clouds. Further, team members may need to use a third-party platform to communicate with each other because the peer clouds may not have the same communication platform.

As an illustrative example, assume two team members Alice and Bob localized in two different cloud environments, Alice prepared a project presentation for their collaborative project. Bob likes to download the proposal from the database of Alice's home cloud. After reading the proposal, Bob has some questions and wants to solve them with Alice. Because their clouds does not share the same interactive platform, they agree to use Facebook to communicate with each other, but they do not like to disclose their discussion to Facebook.

A. Virtual Private Cloud Diagram

In a VPC, a user in one cloud is able to access the resources in a peer cloud. As each peer cloud has its own authentication mechanism, an identity management mechanism is required to enable users of one private cloud to securely access resources of a peer cloud seamlessly, without requiring redundant user administration. Additionally, in a dynamic collaborative environment, some resources (e.g., enterprises, users, applications or services) may join or leave the environment at any point of time. Hence, we design a gateway-based structure (adapted from [16]) as shown in Figure 1.

In the present diagram, the gateway plays a critical role. It enables secure connection between two private clouds transparently. In addition, as it is highly possible that the clouds do not have the same communication platform, the team members may have to use the third-party platform (e.g., social network) to exchange message or interactive communication such as instant-messenging. The present diagram enables secure communication between two team members when a third-party platform is used.

B. Secure Gateway Structure

We adapt the Grid security architecture [17] [18] for the VPC gateways by adding the data security unit. The architecture includes:

- Traffic Collection Unit: collects traffic from network devices such as routers and servers, or peer secure gateways.
- Traffic Processing Unit: classifies the traffic data from the Traffic Collection Unit, then records information such as IP source and destination addresses along with timestamps in a database.
- Network Security Unit: comprises firewall, IDS and virus scanner, etc., which handles the network security function as local legacy gateway. When a threat is identified, it notifies the Response Unit.
- *Data Security Unit*: the security core of the present VPC

 ISBN: 978-1-61208-201-1

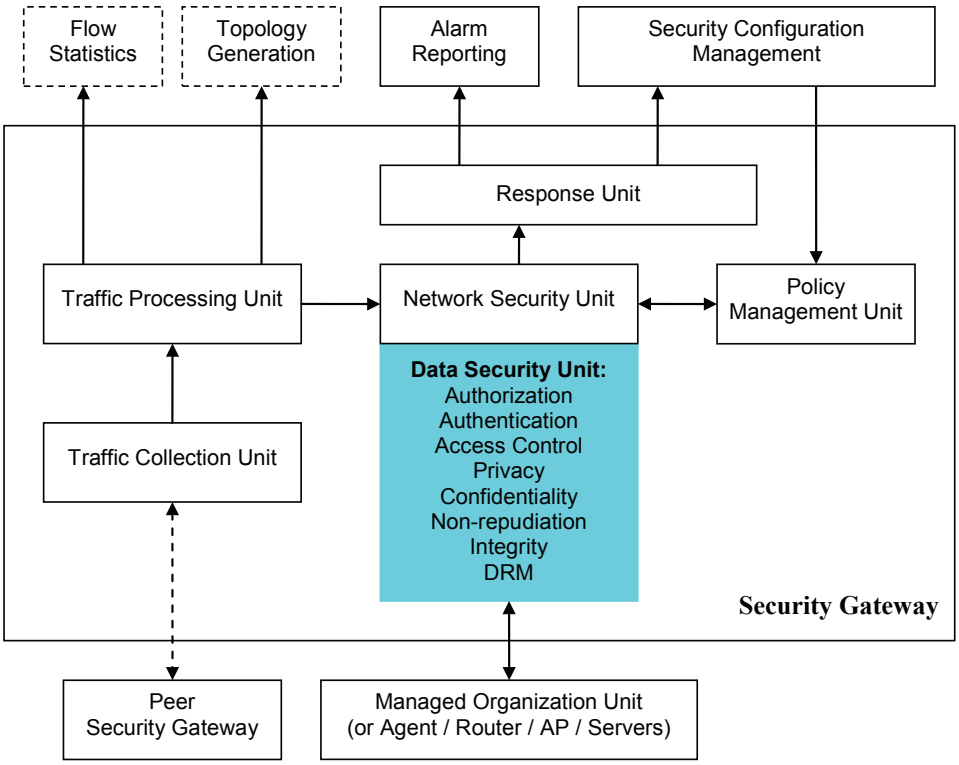


Fig. 2. A VPC gateway structure showing the functional modules.

gateway. It uses the database in Traffic Processing Unit together with the rules from Policy Management Unit to analyze network traffic. When a threat is identified, it notifies the Response Unit.

- Policy Management Unit: provides predefined rules for the Behavior Analysis Unit to identify network anomalies. Depending on management requirement, policies may be updated by Security Configuration Management.
- Response Unit: in the event of detected threats, notifies the Alarm Reporting and Security Configuration Management, who will then react correspondingly.

With reference to Figure 2, a Managed Organization Unit (MOU) may be installed in each computer of the enterprise in order to reduce the burden of the security gateway and to reduce the risk of information leakage. The MOU acts as a coordination point for security functionalities. As different users in an enterprise may have different access priorities and different applications may have different connectivity priorities, the MOU locally enables the users to enjoy cooperation and share resources and services. This capability opens up exciting opportunities for different applications in various fields, such as entertainment, business, healthcare, emergency and education.

C. Secure Connection between VPC Gateways

Figure 3 shows the diagram of the VPC gateway, which comprises two layers. The first layer is used to define and enforce inter-enterprise security, while the second layer is used to define and enforce intra-enterprise security. At the inter-enterprise layer, the gateway includes Network Security Unit (NSU), which is beyond the scope of this paper, and Data Security Unit (DSU). As shown in Figure 2, DSU should implement security functions such as authorization, authentication, access control, confidentiality, and privacy for any transaction between the two private (or enterprise) clouds. In addition, a VPC shall be compliant with the existing private clouds (or peer clouds) and require little change to the intra-enterprise layer. To this end, when a user from a collaborative cloud would like to make use of the resource of a peer cloud, he/she should be treated as a user of the target cloud. Thus, the gateways shall ensure that the security functions can take effect in the process.

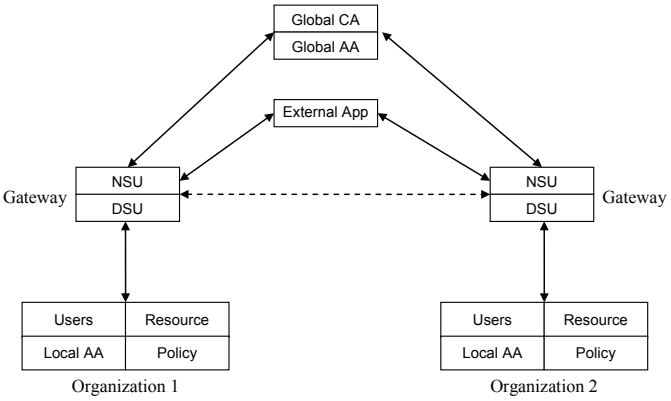


Fig. 3. VPC Gateway connections

For simplicity, we assume the channel between two gateways is mutually authenticated with Public Key Infrastructure (PKI). This assumption can be satisfied easily if each gateway has a digital certificate issued by a trusted Certificate Authority (CA). This secure channel ensures the security of communication traffic such as identification management among private clouds.

D. Access Control in the VPC

In the collaborative cloud, the resource are accessible to the cloud users in two modes which are transparent to the users.

1) Access to Intra-cloud Resources: In an enterprise, any user can be verified using the legacy authentication mechanism (e.g., LDAP). If we regard a gateway as a special user of the enterprise, the user and the gateway can be authenticated via the enterprise's internal mechanism so that they can achieve mutual authentication. As the security structure in Figure 3 does not require any change to the internal access control mechanism of an enterprise, access to the internal resources is transparent to the enterprise members.

2) Access to Inter-cloud Resources: In the VPC diagram shown in Figure 4, the gateway can represent any user within its enterprise to send and receive data across clouds. When a user requests a service or resource from the collaborative enterprise, the gateway is authorized to complete the request on the user's behalf once the user-gateway authentication and the user's privilege checking are successful.

As it is impractical to demand that all the collaborative enterprises adopt the same access control strategy, a VPC gateway should translate the access request from its local user to a standard format (e.g. SAML (Security Assertion Markup Language)) so that the gateway in the target enterprise can enforce the access control. For instance, if the user requests access to the resources of a peer cloud, the gateway in the user's enterprise will translate the request into another format that is compliant with the target enterprise, so that the request can be handled as a local request by the target enterprise. In the collaborative inter-cloud access, the requestor pays the target cloud in name of his/her home cloud so as to maintain the anonymity.

Figure 4 illustrates the authentication process for inter-cloud access. When a user wants to access the resource (or service) of one collaborative enterprise, he sends a request to the local authenticator A1 along with his authentication information (e.g., credential, identity/role/attribute). He also notifies the local gateway (e.g., by network traffic sniffing) to send its credential to the local authenticator A1. After the local authenticator A1 verifies their authenticity, it sends the request to the gateway G1.

The gateway G1 translates the request into a "standard" Collaborative Clouds request format (e.g., SAML format), replaces the requestor with an authorized identity, and signs on the translated request. Then it sends the request to the target gateway G2.

The target gateway G2 verifies the request based on the signature of the sending gateway G1 and translates the "standard" request format into its own request format. Then it sends the request to its own authenticator A2. Once A2 authenticates the request, the user can access the resource or service.

3) Access to External Resources: When two users want to communicate with each other via a third-party platform (e.g., Facebook), the virtual cloud should build its own protection, as the third-party platform may provide no protection at all. To guarantee the security level defined by the enterprises, the VPC gateways should ensure end-to-end security. Loosely speaking, both gateways should create a secure channel for any information exchange between them. Specifically, after each user authenticates himself/herself to the third-party platform as usual, the gateway will encrypt all outgoing messages and decrypt all incoming messages.

III. Discussions

A. Security

In the gateway-based access control scheme, we should consider three security issues. The first issue is the intra-cloud security. As the present scheme does not modify the intra-cloud access or identification method, the security level of the private cloud remains the same. The second issue is the inter-cloud security. As the channel between two gateways is authenticated and confidential, the scheme maintains the security of the inter-cloud. Further, as requestors are authenticated in their own private cloud, the inter-cloud has the same level of security as the intra-cloud. The third security issue is third-party attacks. As the present scheme adopts end-to-end security, it has the same security level as the widely-deployed security systems such as HTTPS-based e-business.

B. Property

Transparency: In the present scheme, a user can access intra-cloud resources and inter-cloud resources in the same way (differing only in the target URI), hence the access mechanism is fully transparent to the users.

Anonymity: When a user sends a request to a peer cloud, a pseudo user name will be used to inform the peer cloud, thus enforcing anonymity.

Dynamics: Due to the anonymity property, when a user joins or leaves the virtual cloud (or task group), the home cloud can handle the dynamics without informing the peer clouds. This property simplifies the collaboration management greatly.

C. Implementation

As proof of concept, we built a simple VPC consisting of three private clouds. Each cloud is constructed with computers supporting BIOS virtualization technology so as to simulate a group of computers. And the network is configured with *OpenStack Flat Network* mode.

Within each private cloud, local authentication and identity management is performed with *Kerberos* 10.04, using the GSS-API mechanism. All local users are registered in the Kerberos system. Upon login, the user is issued with a Kerberos ticket that can be forwarded to other Kerberos users, including the VPC gateway, as proof of his identity. Any two Kerberos

 ISBN: 978-1-61208-201-1

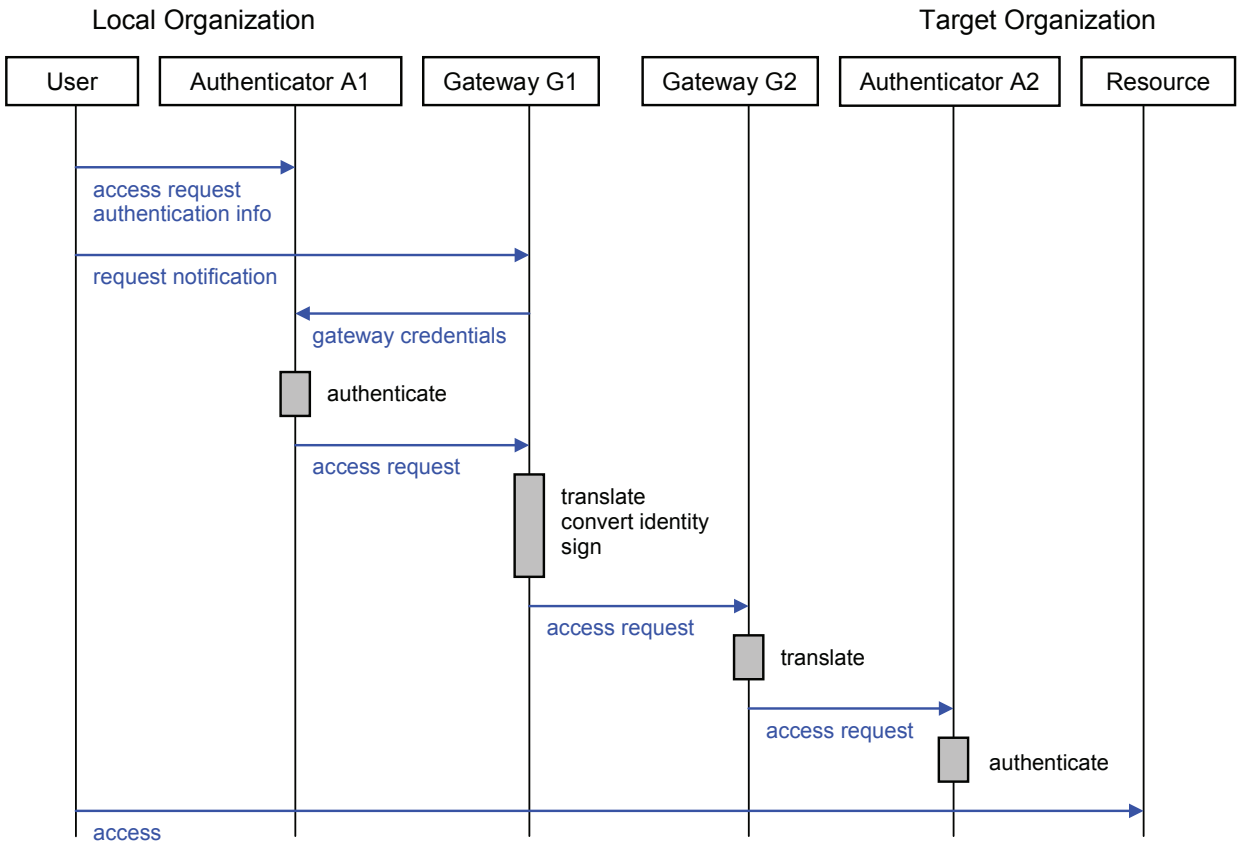


Fig. 4. One-way inter-cloud access

users communicate through a secure channel built by Kerberos GSS-API upon successful mutual authentication. Note that here Kerberos is our choice of mechanism to simulate an existing authentication mechanism in the real practice; the VPC scheme shall apply to any legacy authentication system and identity management. As described earlier in the paper, user identity is verified within his own enterprise and shall be anonymous to peer clouds; hence, there is no need for a dedicated collaborative identity management.

Authentication among VPC gateways across the collaborative cloud uses the Station-to-Station (STS) mutual authentication and key exchange protocol based on PKI. We create a CA within the collaborative cloud that issues signed digital certificates to VPC gateways as they are added to the cloud community. When a VPC gateway contacts another for an inter-cloud request, it first initiates the STS protocol, which includes exchanging certificates for verification and agreeing on a session key, to build a secure communication channel for further processing of the request. Each gateway also keeps a list of known peer gateways along with the services offered within the peer clouds, so that the gateway knows where to route each request.

We tested two scenarios on the VPC. In the first scenario, a user issues a request for a local data resource (i.e., download a file) in the private cloud. In this case, the user is authenticated by his local identity server normally (via Kerberos mechanism in our setting). Upon authorization of access based on the local policy, he then accesses the data directly from the private cloud. In this scenario, the user goes through the same process as he would without the VPC.

In the second scenario, a user issues a request for a data resource in a peer cloud. The access control mechanism in this scenario is as shown in Figure 4. The user request is intercepted by his home gateway, who re-directs the request to the local identity server. The user goes through local (Kerberos) authentication normally, after which, his request is forwarded to the VPC gateway. The VPC gateway proceeds to contact the peer VPC gateway in the target cloud and build a secure communication channel. The user's home VPC gateway processes the request before sending it through the channel, replacing the requestor identity with a pseudo user name to achieve anonymity. Upon receiving the request, the VPC gateway in the destination cloud checks its own local access policy and determines that the user is authorized to access the data requested. The gateway then forwards the request to the resource provider, who then sends the requested data to the user via the two gateways. This scenario shows how VPC can achieve inter-cloud access without altering user experience, that is, the user still goes through the same authentication process in his local server, and the remote authorization mechanism is fully transparent to him.

 ISBN: 978-1-61208-201-1

IV. Related Work

Cloud infrastructure commonly relies on virtualization machines so as to provide the properties of flexibility and application independence. When a user requests for resource properties (such as processor speed, time and memory size), the service provider will create a virtual machine satisfying the request.

Although virtual machines have become increasingly commonplace as a method of separating hostile or hazardous code from commodity systems, the potential security exposure from implementation flaws has increased dramatically. However, cloud security issues cannot be solved with just virtualization technologies [19]. Ormandy [20] investigated the state of popular virtual machine implementations for x86 systems, and assessed the security exposure to the hosts of hostile virtualized environments.

A. Intra-cloud security

Chow et al. [21] suggested to use trusted computing and computation-supporting encryption to enhance the security of cloud computing. Popovic and Hocenski [22] suggested considering privacy and security at every stage of a system design, while other researchers took care of trust [23], [24] and authorization [25].

Takabi et al. [26] proposed a comprehensive security framework for cloud computing environments. They also discussed challenges, existing solutions, approaches, and future work needed to provide a trustworthy cloud computing environment.

Demchenko et al. proposed an architectural framework for on-demand infrastructure service provisioning in [27], and discussed security mechanisms required for consistent DACI (Dynamically provisioned Access Control Infrastructure) operation using authorisation tokens in [28]. Shin and Akkan [29] proposed a domain-based framework for provisioning and managing users and virtualized resources in IaaS to support scalable management of users and resources, organization-level security policy, and flexible pricing model.

As a standard for identity management, SAML defines identity provider (IdP) and service provider (SP). The IdP focuses on identity management, access policy management, and security token generation, while SPs receive the remote security token, retrieve credential data, and reinforce user access policies locally. In practice, the schemes in compliance with IdP/SP model may focus on different properties, e.g., protocol flow [30], scalability [31], privacy [32], friendliness with device identity or user behavior [33], and SSO (Single Sign On) [34]. In all, SAML allows authentication so that a cloud can provide services to users both inside and outside the cloud.

B. Inter-cloud security

Riteau [35] built distributed large-scale computing platforms from multiple cloud providers, allowing to run software requiring large amounts of computation power so as to provide inter-cloud live migration and offer new ways to exploit the inherent dynamic nature of distributed clouds. Similarly, Nguyen et al. [36] presented a cloud architecture that allows users with different security authorizations to securely collaborate and exchange information using commodity computers and familiar commercial client software.

For a cloud community formed by different vendors or enterprises, Kretzschmar and Hanigk [37] intensified cloud security management domains, integrated various cloud security services of an organization and providing interoperability for the clouds. Moreover, Kretzschmar and Golling [38] identified functional components for a Security Manager architecture. These components, together with identified security data artifacts, are able to support the cloud provider community to some extent.

Bernstein et al. presented an InterCloud protocol to solve the cloud computing interoperability problem in [39], and also considered the InterCloud security such as identity management and access control in [40]. Generally, InterCloud is the focus of efforts especially in the public sector (e.g., USA Federal Government's Cloud Computing Initiative). It can be regarded as the second layer in the cloud computing stack [41]. In the inter-cloud layer, client-centric distributed protocols complement more provider-centric, large-scale ones in the intra-cloud layer. These client-centric protocols orchestrate multiple clouds to boost dependability by leveraging inherent cloud heterogeneity and failure independence. Celesti et al. [42] addressed the Identity Management (IdM) problem in the InterCloud context and showed how it can be successfully applied to manage the authentication needed among clouds for the federation establishment.

The above inter-cloud architectures or protocols enable to secure collaboration among clouds. However, they are self-contained, and may require modification of legacy authentication systems.

V. Conclusions and Future Work

Collaborative cloud is used to develop a dedicated task such as flight design such that the users can share resources in a confidential, authentic and transparent way. The paper presents a VPC gateway mechanism so as to build a secure channel for the users in the collaborative environment. With few modifications on the private clouds, it supports the resource sharing among private clouds and 3rd-party communication platforms.

In our prototype, we implemented the one-way inter-cloud access protocol for demonstrating the soundness of the proposed diagram only. The future work will be to develop the whole system, in particular to integrating with the standard IdP/SP protocol, and securing the 3rd platforms.

Acknowledgement

The authors would like to appreciate the anonymous reviewers for their constructive comments and suggestions. Dr. Hui Fang and Zhigang Zhao helped to build the cloud computing platform and VPC gateways respectively.

 ISBN: 978-1-61208-201-1

REFERENCES

[1] National Institute of Standards and Technology, DRAFT Cloud Computing Synopsis and Recommendations. http://csrc.nist.gov/publications/drafts/800-146/Draft-NIST-SP800-146.pdf.. Last Access on 30.03.2012.

[2] David Bernstein, Erik Ludvigson, Krishna Sankar, Steve Diamond, and Monique Morrow, "Blueprint for the Intercloud -Protocols and Formats for Cloud Computing Interoperability," IEEE International Conference on Internet and Web Applications and Services, 2009, pp. 328-336.

[3] Giorgio Nebuloni, "Accelerate Hybrid Cloud Success: Adjusting the IT Mindset," IDC, 04.10.2011.

[4] Anuradha Shukla, "China to invest £98 billion in cloud computing," ComputerWorld UK, 14.09.2011.

[5] E. Messmer, "Are security issues delaying adoption of cloud computing?" Network World, 27.04.2009. http://www.networkworld.com/news/2009/042709-burning-security-cloud-computing.html. Last Access on 30.03.2012.

[6] V. Tchifilionova, "Security and Privacy Implications of Cloud Computing Lost in the Cloud," Open Research Problems in Network Security, Lecture Notes in Computer Science, Vol. 6555, 2011, pp. 149-158.

[7] D. Velev, and P. Zlateva, "Cloud Infrastructure Security," iNetSec 2010, Lecture Notes in Computer Sciences 6555, 2011, pp. 140-148.

[8] R. Bhadauria, R. Chaki, N. Chaki, and S. Sanyal, "A Survey on Security Issues in Cloud Computing," ACM Computing Research Repository, vol. abs/1109.5388, http://arxiv.org/abs/1109.5388, 2011. Last Access on 30.03.2012.

[9] Craig Balding, "Go-Grid Security Breach," 30.03.2011. http://cloudsecurity.org/blog/2011/03/30/gogrid-security-breach.html. Last Access on 30.03.2012.

[10] Czaroma Roman, "Sony Data Breach Highlights Importance of Cloud Security," Cloud Times, 09.05.2011. http://cloudtimes.org/sony-data-breach-highlights-importance-of-cloud-security/. Last Access on 30.03.2012.

[11] IDC press, "Cloud Adoption Will Have Major Impact on European Software Market in 2011," 07.03.2011. http://www.idc.com/about/viewpressrelease.jsp?containerId=prDK22728011&sectionId=null&elementId=null&pageType=SYNOPSIS. Last Access on 30.03.2012.

[12] Robert Mullins, "IDC Survey: Risk In The Cloud," Network Computing, 16.06.2010.

[13] Jia Xu, Jia Yan, Liang He, Purui Su, and Dengguo Feng, "CloudSEC: A Cloud Architecture for Composing Collaborative Security Services," IEEE International Conference on Cloud Computing Technology and Science, 2010, pp. 703-711.

[14] S. T. Zargar, H. Takabi, and J. B.D. Joshi, "DCDIDP: A distributed, collaborative, and data-driven intrusion detection and prevention framework for cloud computing environments," International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2011, pp. 332-341.

[15] T. Sasaki, M. Nakae, and R. Ogawa, "Content Oriented Virtual Domains for Secure Information Sharing Across Organizations," ACM Cloud Computing Security Workshop, 2010, pp. 7-12.

[16] A. Kipp, L. Schubert, and M. Assel, "Supporting Dynamism and Security in Ad-Hoc Collaborative Working Environments", Identity in the Information Society, 2(2):171-187, 2009.

[17] Chih-Mou Shih, and Shang-Juh Kao, "Security Gateway for Accessing IPv6 WLAN," IEEE/ACIS Int'l Conf. on Computer and Information Science and Int'l Workshop on Component-Based Software Engineering, Software Architecture and Reuse, 2006, pp. 83-88.

[18] Ying-Dar Lin, Huan-Yun Wei, and Shao-Tang Yu, "Building an Integrated Security Gateway: Mechanisms, Performance Evaluations, Implementations, and Research Issues," IEEE Communications Surveys & Tutorials 4(1):2-15, 2002.

[19] M. Christodorescu, R. Sailer, D. L. Schales, D. Sgandurra, and D. Zamboni, "Cloud Security Is Not (Just) Virtualization Security," ACM Cloud Computing Security Workshop, 2009, pp. 97-102.

[20] Tavis Ormandy, "An Empirical Study into the Security Exposure to Hosts of Hostile Virtualized Environments," 15.04.2008. citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.105. Last Access on 30.03.2012.

[21] R. Chow, P. Golle, M. Jakobsson, R. Masuoka, J. Molina, E. Shi, and J. Staddon, "Controlling Data in the Cloud: Outsourcing Computation without Outsourcing Control," ACM Cloud Computing Security Workshop, 2009, pp. 85-90.

[22] K. Popovic, and Z. Hocenski, "Cloud computing security issues and challenges," Int'l Convention on Information and Communication Technology, Electronics and Microelectronics, 2010, pp. 344-349.

[23] Khaled M. Khan, and Qutaibah Malluhi, "Establishing Trust in Cloud Computing," IT Professional, 2010, 12(5):20-27.

[24] Jiyi Wu, Qianli Shen, Jianlin Zhang, and Qi Xie, "Cloud Computing: Cloud Security to Trusted Cloud," Advanced Materials Research, New Trends and Applications of Computer-aided Material and Engineering, vol. 186, 2011, pp. 596-600.

[25] P. Bryden, D. C. Kirkpatrick, and F. Moghadami, "Security Authorization: An Approach for Community Cloud Computing Environments," White Paper, Nov. 2009. http://www.techrepublic.com/whitepapers/. Last Access on 30.03.2012.

[26] Hassan Takabi, James B.D. Joshi, and Gail-Joon Ahn, "SecureCloud: Towards a Comprehensive Security Framework for Cloud Computing Environments," IEEE 34th Annual Computer Software and Applications Conference Workshops, 2010, pp. 393-398.

[27] Yuri Demchenko, Jeroen Van der Ham, Volodymyr Yakovenko, Cees de Laat, Mattijs Ghijsen, and Mihai Cristea, "On-demand provisioning of Cloud and Grid based infrastructure services for collaborative projects and groups," International Conference on Collaboration Technologies and Systems, 2011, pp. 134-142.

[28] Y. Demchenko, C. Ngo, and C. de Laat, "Access control infrastructure for on-demand provisioned virtualised infrastructure services," International Conference on Collaboration Technologies and Systems, 2011, pp. 466-475.

[29] Dongwan Shin, and Hakan Akkan, "Domain-based virtualized resource management in cloud computing," International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2010, pp. 1-6.

[30] S. Eludiora, O. Abiona, A. Oluwatope, A. Oluwaranti, C. Onime, and L. Kehinde, "A User Identity Management Protocol for Cloud Computing Paradigm," Int. J. Communications, Network and System Sciences, vol. 4, No. 3, 2011.

[31] Anu Gopalakrishnan, "Cloud Computing Identity Management," SETLabs Briefings, 7(7):45-55, 2009.

[32] Elisa Bertino, Federica Paci, and Rudolfo Ferrini, "Privacy-Preserving Digital Identity Management for Cloud Computing," IEEE Computer Society Data Engineering Bulletin, 2009, pp. 1-4.

[33] R. Chow, M. Jakobsson, and R. Masuoka, "Authentication in the Clouds: A Framework and its Application to Mobile Users," CCSW, 2010, pp. 1-6.

[34] Juniper Networks, "Identity Federation in a Hybrid Cloud Computing Environment Solution Guide," Oct. 2009. www.ictnetworks.com.au/pdf/8010035-en.pdf. Last Access on 30.03.2012.

[35] Pierre Riteau, "Building Dynamic Computing Infrastructures over Distributed Clouds," International Symposium on Network Cloud Computing and Applications, 2011, pp. 127-130.

[36] T. D. Nguyen, M. A. Gondree, D. J. Shifflett, J. Khosalim, T. E. Levin, and C. E. Irvine, "A cloud-oriented cross-domain security architecture," Military Communications Conference, 2010, pp. 441-447.

[37] M. Kretzschmar, and S. Hanigk, "Security management interoperability challenges for Collaborative Clouds," International DMTF Academic Alliance Workshop on Systems and Virtualization Management, 2010, pp. 43-49.

[38] M. Kretzschmar, and M. Golling, "Functional components for a Security Manager within future Inter-Cloud environments," International Conference on Network and Service Management, 2011, pp. 1-5.

[39] D. Bernstein, and D. Vij, "Intercloud Directory and Exchange Protocol Detail Using XMPP and RDF," World Congress on Services, 2010, pp. 431-438.

[40] D. Bernstein, and D. Vij, "Intercloud Security Considerations," IEEE International Conference on Cloud Computing Technology and Science, 2010, pp. 537-544.

[41] C.n Cachin, R. Haas, and M. Vukolic, "Dependable Storage in the Intercloud," IBM Research Report, RZ 3783, 21.10.2010.

[42] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "Security and Cloud Computing: InterCloud Identity Management Infrastructure," IEEE International Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises, 2010, pp. 263-265.

 ISBN: 978-1-61208-201-1

Process Discovery and Guidance Applications of Manually Generated Logs

Stefan Schönig, Christoph Günther, Stefan Jablonski
University of Bayreuth
Chair of Applied Computer Science IV
Bayreuth, Germany
{stefan.schoenig, christoph.guenther, stefan.jablonski}@uni-bayreuth.de

Abstract - **In this paper, we investigate the problem of the availability of complete process execution event logs in order to offer automatic process model generation (process discovery) possibility by process mining techniques. Therefore, we present the Process Observer project that generates manual logs and guides process participants through process execution. Like this, our project offers the possibility for the automatic generation of process models within organizations, without the availability of any information system. Process participants are encouraged to work with the Process Observer by various process execution support functions, like an auto-suggestion of process data and dynamic recommendations of following processes.**

Keywords - Process Mining, Process Monitoring, Activity Tracking, Guidance through Process Execution

I. INTRODUCTION

Business process management (BPM) is considered an essential strategy to create and maintain competitive advantage by modeling, controlling and monitoring production and development as well as administrative processes [1] [2]. Many organizations adopt a process based approach to manage various operations. BPM starts with a modeling phase, which is very time and cost intensive. It requires deep knowledge of the underlying application and long discussions with the domain experts involved in the processes in order to cover the different peculiarities of the process [3]. Since process modeling is an expensive and cumbersome task, we identify approaches that promise to reduce the modeling effort. One of them is process mining. Process mining utilizes information/knowledge about processes whilst execution. The idea is to extract knowledge from event logs recorded by information systems. Thus, process mining aims at the (semi-)automatic reconstruction of process models using information provided by event logs [4]. The computer-aided creation of process models offers huge potential of saving time. By deriving process models from event logs, the appropriateness of process models can be guaranteed to a certain extent, since they are constructed according to the way the processes have actually been executed. During the last decade, many techniques and algorithms for process mining have been developed and evaluated in different domains [5]. The basis for a successful generation of a process model through process mining is an existing and complete process execution log. This is also the big challenge for a successful application of process mining. First of all, not all processes are executed by information systems, i.e., they are executed "external" to computers. In such cases, there is no event log that represents a process available and process mining cannot be applied. In the case that information systems are already used to execute processes there must be guarantees that these event logs record process execution in such a way that processes can be reconstructed. Besides, these event logs must be analyzable in such a way that appropriate process models can be derived. It is obvious: the quality and availability of event logs determine the applicability of process mining techniques. Our research starts with the assumption that a complete and freely analyzable event log is usually not available. We regard this scenario as the most common one. Thus, one of the major aims of our research is to harvest process execution knowledge. This enables the assembly of a process execution log. This log is built up independently from the existence of information systems that are (at least partly) executing the processes. We developed a special software, the Process Observer (PO), that can be envisioned as a tool that permanently runs on the computers of process participants that asks the process participants “What are you doing right now?”. The participants then have to describe what they are doing. Here, the user does not need any process modeling skills. This is also one very important prerequisite since we assume that just few process participants do show process modeling skills. The recorded data is used by the PO to mine for process models. Of course, this process information can be enriched and complemented by event logs from information systems that are involved in the process execution. Gathering process execution information comes with the cost that process participants have to record what they are doing. Of course, this means additional work for the process participants. Therefore, the PO must offer a stimulus that motivates process participants to work with the PO. This stimulus is put into effect by a recommendation service. The PO continuously analyzes available process log data to guide the process users. This means, it suggests process steps that the user most probably should perform. We have experienced that this feature is especially important for users that are still not too familiar with the application; they are thankful that the PO recommends possible process steps. This dynamic recommendation service becomes more and more reliable the more process instances have been executed under the guidance of the PO. The execution of first instances of a process will therefore not considerably be supported. The effect becomes apparent when a couple of process instances have been executed. At the end of this introduction, we want

 ISBN: 978-1-61208-201-1

to classify the PO. As dimensions for this classification we take the two issues: attaining a process model and executing a process model. We already discussed the two principal approaches to attain a process model. They will be assessed with respect to the amount of effort a process participant has to or is able to invest. The first approach to attain a process model is to create it within a process modeling project. This task is very costly; it usually cannot be performed by process participants but requires process modeling experts. They identify the process through interviews with the domain experts and need to get a good overview over all possible process peculiarities to guarantee the completeness of the process model. Process models can also be attained by the application of process mining techniques. This approach is cheap since only little work from process modelers is required. However, it depends on the existence of event logs representing the execution of processes. These two approaches depict two extreme landmarks: on the one hand processes can be performed within information systems. On the other hand, information systems could not be involved at all. The PO bridges the contrary approaches of process execution und thus combines their benefits. It is connectable to process execution systems and can leverage them; also it provides execution support for "external" process execution.

In Section II, we will give an overview over related works. In Section III we will explain our concepts and the general approach. Furthermore, concrete implementation techniques will be shown in Section IV. Section V describes the influence of the PO on the current process lifecycle. In Section VI we will finally conclude and give an outlook on further research issues and applications.

II. RELATED WORK

The idea of automating process discovery through event-data analysis was first introduced by Cook and Wolf in the context of software engineering processes [6]. In the following years, Van der Aalst et al. developed further techniques and applied them in the context of workflow management under the term process mining [5]. Generally, the goal of process mining is to extract information about processes from event logs of information systems [7]. There are already several algorithms and even complete tools, like the ProM Framework [8], that aim at generating process models automatically. During the last decade, several algorithms have been developed, focusing different perspectives of process execution data. Van der Aalst et al. give a detailed introduction to the topic process mining and a recapitulation of research achievements in [5] and [7]. For the first prototype of the PO, we use the alpha-algorithm of [3]. However, for our future research activity we consider algorithms like the HeuristicsMiner [9] appropriate, because they are able to deal with noisy logs, i.e., incorrectly or incomplete logged information. Process mining algorithms rely on complete event logs from information systems. In the case of an incomplete log or even the unavailability of an information system, events can alternatively be recorded by manual activity tracking respectively task management methods. There are several approaches for activity tracking to generate weakly-structured process models by capturing data on personal task management [10] [11]. However, these approaches lack the use of process mining techniques during and after process run-time. In contrast to that we explicitly try to encourage user contribution to an evolving process model by using process mining methods. In order to discover identical processes between different data storages, we suggest using basic automatic ontology matching algorithms [12]. Process mining is considered as a part of Business Process Management (BPM). BPM relies on a life-cycle where different phases of the process are focused. The traditional approach consists of the following phases: process modeling, implementation, execution and evaluation, started by the modeling step. Despite the successful development and evaluation of the process mining algorithms named above, process mining is ranked among the process evaluation phase [1]. Consider, for example, Enterprise Resource Planning (ERP) systems such as SAP, OpenERP, Oracle, Customer Relationship Management (CRM) software, etc. These systems require a designed process model before they go into service [3]. In these situations, process mining could only be used for process rediscovery and not for real process discovery. Therefore, we aim at assigning process mining to the discovery phase by recording the complete process data covering all aspects of the perspective-oriented process modeling (POPM). In order to get a general idea about POPM perspectives, we recommend [13] and [14].

III. GENERATION OF PROCESS EXECUTION LOGS AND GUIDANCE THROUGH PROCESS EXECUTION

Process mining techniques allow for automatically constructing process models. The algorithms are analyzing a process execution log file, in the following referred to as (process) log; this log is usually generated by information systems (IS). However, there are processes that are not executed by information systems. This is an observation that is very important for the classification of our research. Thus, in order to define the application area of our project we have to introduce three different types of process execution support, classified upon the degree of logging and execution support (Fig. 1):

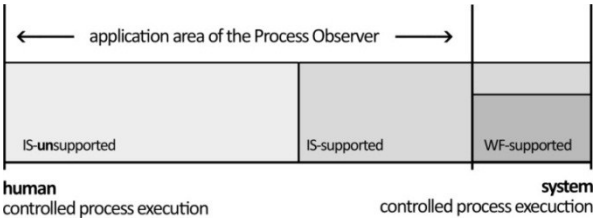


Figure 1. Application area of the Process Observer project

- *IS-unsupported*: Here, processes are executed without the support of any information system. Thus, there is no log for these processes. Furthermore, these processes are also not supported during execution. For example, there is no information system that guides a user through the process.

- *IS-supported*: Here, processes are executed by an information system. Processes of this type are (possibly) logged. However, the information system is not directly guiding users through the process. The user has to find his way through the information system by himself.
- *WF-supported*: Here, processes are executed by Workflow management systems (WFMS). WFMS build a subset of IS. Typically, they maintain a process log. Additionally, the process participants are guided through process execution with concrete recommendations of how to continue process execution (work list) [15].

The basis for the successful generation of process models through process mining is an existing and complete log. Thus, WF-supported processes are a great source for process mining. Nevertheless, the existence of a process log is the main prerequisite and also the major drawback for a successful application of process mining. Since we assume that in many applications, WF-supported processes will not be encountered the PO turns its attention to IS-supported and IS-unsupported processes (Fig. 1). In order to log IS-unsupported processes, we extend process execution by manual logging. We define the term manual logging as the user action of entering process execution data (e.g., process IDs, documents, and services) as well as of marking process execution events, among other things process start and completion. The action of manual logging is implemented by the PO Logging Client. Finally, our goal is to provide manual logging in such cases when processes are neither IS-supported nor WF-supported. The final aim is then to be able to apply process mining.

A. Aims of the Process Observer

The challenge of the PO is to provide a broader basis for process mining by implying IS-unsupported processes in logs. Therefore, the PO project aims at the adoption and generation of manual logs. The generated manual logs open the opportunity for the automatic generation of process models by process mining techniques even for applications that do not involve information systems. As manual logging is performed by process participants, it means additional work for them. Therefore, the PO must offer a stimulus that motivates process participants to support manual logging. Since the PO is particularly of interest for IS-unsupported and IS-supported processes, it offers a stimulus with respect to process execution guidance (this is what these two kinds of processes are lacking). The PO offers recommendations about how to continue a process execution and offers auto-suggest support. This kind of guidance during process execution is typically exclusively offered by WFMS.

B. Generation of Manual Logs

From now on, we generally assume that a complete and freely analyzable log is not available, i.e., we are focusing on IS-(un)supported processes. We regard this scenario as the most common one and it needs to be supported to apply process mining.

1) Manual Logging:

The generation of a manual log is initiated by the PO Logging Client. Process participants record what they are currently doing, i.e., they provide information about the process they are currently performing. It is very important that users do not need any process modeling skills to record this information.

An important issue is to determine what data the process participants should record. We recommend to record data based upon the different aspects of perspective oriented process modeling (POPM). We have experienced that most users are very familiar with the approach of describing process in the POPM method. Process participants have to enter data according to the following perspectives:

- *Functional* perspective: name of the current process step, the name of the corresponding superordinate process (if available)
- *Data* perspective: data, i.e., documents or generally information that was used by the current process step as well as the data or documents that were produced
- *Operational* perspective: tools, applications or services that were used during the execution of the currently executed process step
- *Organizational* perspective: information about the process executor (typically, this is that person that is logged into the PO Logging Client), the personal information is enriched by group and role memberships

Besides, process participants have to trace process execution: he has to declare that process execution starts, ends or is aborted.

2) Merging Logs:

The application of the PO Logging Client finally results in the generation of a manual log. In the case, that an information system is applied, there might also be an automatic log available. We harness this situation by combining the manual log with the automatic log. Doing this, missing process information of one of the logs can be completed by the other log. In order to be able to combine the two logs, conformance between the recorded data of both logs must be achieved. Therefore, we suggest a component for merging the logs, i.e., locating (matching) and unifying processes that were recorded in the manual log as well as in the automatic log. This results in one consistent log that contains the execution data of IS-unsupported as well as IS-supported processes.

C. Guidance through process execution

According to our classification in Fig. 1, many process executions are not assisted by a guidance component, i.e., the participants must decide for themselves which process step they want to perform next. Only WF-supported processes do provide this feature. In this subsection, we will show how the PO offers such guidance. It consists of two sub-features: dynamic recommendations and auto-suggest function.

 ISBN: 978-1-61208-201-1

1) Dynamic Recommendations:

Dynamic recommendations are generated in the following way: After the completion of a process step, the PO immediately starts a process mining algorithm analyzing available log data. It then tries to classify this current process execution into former process executions. If it is successful, the PO can recommend the execution of a subsequent process step according to the processes that have been executed formerly. This recommendation service becomes more and more reliable the more process instances have been executed under the guidance of the PO. When only a few or even none processes of this type have been executed so far, no recommendations can be made for the particular process. Especially when only a few process instances have been performed so far, the recommendation can be inconsistent. Then, process participants can ignore this recommendation. In order to know about the quality of the recommendation, the number of process instances the recommendation is based upon is displayed in the user interface.

Example: A process participant just completed a process step A. This step has already been completed and recorded 10 times before by other agents. On the one hand, step B was executed 7 times after step A; on the other hand, step C was executed 3 times after step A. The PO now starts process mining and generates a process model that contains the information that process A shows two subsequent processes B and C. Furthermore, the tool takes into account that step B occurred 7 times and step C occurred 3 times after step A in the log. Thus, a dynamic recommendation is shown to the user suggesting to continue with step B (70%) or step C (30%).

2) Auto Suggest Function:

The second aspect of guidance during process execution is provided by an auto-suggest function. This function helps the process participant to enter required information. The PO compares previously recorded process names, data, tool names, etc. with the currently entered term and auto-suggests terms. This function supports two issues: first, the user might nicely be supported through information provision; secondly, by suggesting already used terms, the probability of having to deal with too many aliases in the system is diminished to a certain extent.

Example: Agent 1 is executing a process "Drinking Coffee". Agent 1 starts the process by recording the process name, i.e., Agent 1 enters "Drinking Coffee". The agent starts and completes the process. The process gets a unique identifier and is recorded in the log. Later, Agent 2 also wants to drink coffee and executes this process with support of the PO. He starts by typing "Coffee" instead of "Drinking" in the process name row. This would easily result in the recording of a process name like “Coffee Drinking” or just "Coffee". So, aliases are produced without even recognizing. However, in this case an auto suggestion will appear, recommending to choose the process "Drinking Coffee". Agent 2 happily chooses the suggested process and thus ensures homogenous naming of the process step.

3) Visualization and manual mapping of processes:

Example: If the example from the former sub-section would occur as described, this would be ideal. However, in many cases same processes will be referenced by different aliases and thus stay unrecognized by the PO. In order to handle problems like this, the PO offers an administration interface, which allows process administrators to visualize recorded processes. Administrators can start process mining algorithms and thus generate process models visualizing observed processes. Doing this, different aliases of processes can be discovered. However, this must be done manually by the administrator. In order to map different aliases of the same process, the PO administration interface offers a mapping panel. This mapping can be declared valid for multiple processes (Fig. 2). After defining a mapping between processes, a repeated execution of process mining results in the visualization of the amended process model.

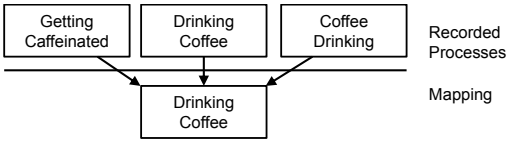


Figure 2. Sample mapping of recorded processes

D. Usage scenarios for the Process Observer

As a conclusion, we will give a short description of three different application scenarios of the PO.

1) Use Case 1 – Generation of manual logs:

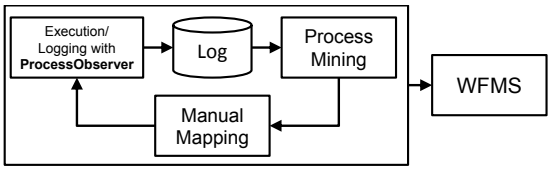


Figure 3. First use case – generation of manual logs

The first use case comprises the generation of a manual log (Fig. 3) without an information system being available. The participating agents are executing the corresponding processes under the guidance of the PO. The manual log is finally analyzed by process mining algorithms. The resulting process models can be fed into a WFMS if wanted and if available. Thus, processes can afterwards be executed by a WFMS.

2) Use Case 2 – Merging of logs:

The second use case comprises the application of the PO in parallel to an information system (Fig. 4). After the generation of a manual log, we have to merge the automatic and the manual log.

 ISBN: 978-1-61208-201-1

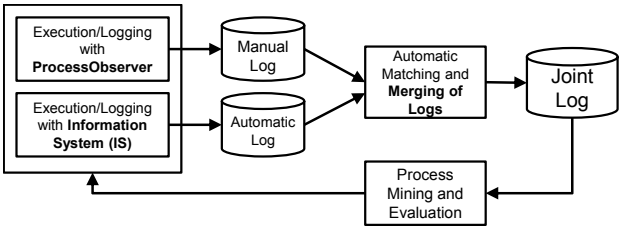


Figure 4. Second use case – merging of logs

The intention is to complete the log information mutually. Identical processes are merged to one single process. Process Mining is finally applied to the joint log. Identified processes can be fed back into information systems.

3) Use Case 3 – Running WFMS:

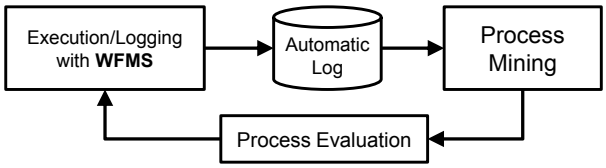


Figure 5. Third use case – running WFMS

The third use case assumes a fully-fledged WFMS running (Fig. 5). Here, manually logging is not necessary anymore because the WFMS includes all the processes being executed. It is important to define a threshold, when process management can shift from case 2 to case 3. Therefore, we define a value *matching_count* (1) as the number of matched processes from the manual log and the automatic log divided by the complete number of processes recorded in the manual log. The procedure of calculating this value is the following: the algorithm runs through both logs. It compares each process of the manual log with the processes of the automatic log. If an ontology matching algorithm identified two processes as equal, the numerator *#matched_processes* will be increased by 1. After finishing traversing both log files, the resulting value of *#matched_processes* is divided by the total number of recorded processes within the manual log.

$$matching\_count = \frac{\#matched\_processes}{\#recorded\ processes\ with\ Process\ Observer} \quad (1)$$

Like this, the calculated value reflects how many processes are already executed with support of the WFMS. Generally, an organization finally tries to execute all processes under the guidance of the WFMS, but the preferred value of *matching-count* can also alternatively be defined by the management. For a special organization a *matching-count* value of 0.9 may be enough. This means, 90% of the executed processes are implemented and supported by the WFMS. Like this, the right time of the application end of the PO can be declared by continuously calculating the *matching_count* (1) value.

IV. ARCHITECTURE AND IMPLEMENTATION

In this section, we will describe the architecture and implementation of the PO. In the first part, we will show implementation details of the PO Logging Client. After that, process mining implementation and data structures will be explained. Furthermore, we present the administration and mapping components.

A. Process Observer Logging Client

The core of the PO is constituted by the PO Logging Client. We decided to choose a web based implementation of the logging interface. This guarantees a great coverage of application scenarios, i.e., the PO can be used in almost all applications. If the users are working in a "normal" office, the PO can run on a stationary PC or notebook, if users are working "in the field", the PO could as well run on a mobile device (e.g., smartphone). For our prototype we chose an implementation based on Microsoft ASP.NET 4.0 and the MS SQL Server 2008 database, but surely any equally equipped database and server technology would be suitable. The core of the web application that implements the PO Logging Client is located on a web server connected to a database. Users have to identify themselves by logging in with their username and password. Users can be assigned to one or more organizational roles. Hence, recommendations and suggestions can be personalized to the users' roles. When users enter process names they want to log, these text strings are immediately sent to the PO to test for similar process names. The names of all processes containing a similar string are sent back to the client as a generic list. This list is finally displayed to the user as an auto suggestion list (Fig. 6). The user can select a process from this list. If none of the suggested processes is appropriate, the input process name is added as a new process. Accordingly, all other process data are captured (e.g., superordinate process, current process instance, used and produced data/documents and supporting tools). Finally, the user starts the process.

Figure 6. Example of auto suggestion list

B. Implementation of process mining, data structures and dynamic recommendations

As already described in Section III, the PO offers dynamic recommendations of how to continue after finishing a process step. Therefore, a process mining algorithm is executed after each process step. In our prototype we use the alpha algorithm of [3] in order to analyze the available logging information. The algorithm analyzes the log and builds up a dependency graph. Therefore, we used the graph data structure QuickGraph of

 ISBN: 978-1-61208-201-1

[16]. For implementation details concerning the alpha algorithm we refer to [3]. The logged execution information results in process models represented as graphs. A node is an instance of a class "Process" containing fields for process name, the executing originator role, used and produced data items as well as supporting tool items. Furthermore, the class contains two fields for the pre- and post-connectors which represent the semantic connection to previous and following processes. This information is also provided by the alpha algorithm. Once a process model has been generated as a graph, the PO can use it in order to display recommendations after a user has finished a process step. Therefore, the recently completed process is searched within the process model, i.e., the graph is traversed until the current process ID is identical to the recently completed one. After that, all available edges of this node are examined and their occurrence is counted. Like this, we generate a list, containing the processes that were executed after the recently completed one. Thus, a popup is displayed, giving the user the possibility to choose the following process step.

C. Administration interface

Additionally, the PO offers an administration interface that allows process administrators to visualize recorded processes as well as defining mappings between logged processes as described in Section III. The application consists of two panels, one for process model selection and visualization and the other one for defining mappings between processes. One could easily imagine additional applications, like agent-role assignments or dataflow applications. Those are planned for future versions.

1) Process visualization:

In order to visualize the generated process model we use basic graph visualization frameworks. In our prototype we used the Graph# framework [17] to display the QuickGraph data structures. The visualization procedure is started by examining the recorded event log for contained composite processes. A process is recognized as composite, if it was chosen as a superordinate process by a process participant during the logging phase of a process with the PO. The names of the composite processes are loaded in a tree view. The user selects a composite process that should be displayed from the tree view. The tree view shows the underlying process hierarchy. Processes that are contained within another one can be displayed by extending a process entry. After the selection of an entry, all event log information concerning the selected process is fetched from the database. After that, the alpha algorithm is applied to the resulting event log data. As stated before, the algorithm generates a dependency graph. This graph is finally assigned to the Graph# framework and displayed to the user. Here, the user has various possibilities to scroll within the visualization or to open the model of underlying composite processes by selecting the corresponding process nodes.

2) Mapping definition panel:

Furthermore, the administration interface offers a separate panel to define mappings between logged processes. Therefore, the database provides a separate mapping table with three columns: “superordinate process”, i.e., the super process within the mapping is valid, “target process”, i.e., the process on which another one is mapped and finally “mapped process”, i.e., the process which is mapped. Considering this data model, the mapping panel consists of three columns, too. They appear after the first things first principal. In the first list, the user selects the superordinate process within the mapping should be valid. After this selection, the target process list appears. The list is initialized with all processes occurring within the chosen superordinate process. Like this, the user can choose the target process for the defined mapping. Last but not least, the last list, i.e., a checkbox list, appears. It is again initialized with all processes of the corresponding super process. Here, the user checks all the corresponding boxes of the processes he would like to map on the target process chosen before. Finally, the mapping is applied to the database.

V. CHANGES WITHIN THE PROCESS LIFECYCLE THROUGH THE PROCESS OBSERVER

In this section, we will describe the impact of the PO on different phases in the process lifecycle. As already mentioned, the previous process lifecycle [1] consists of an initial modeling phase that is very time consuming. In this lifecycle, process mining is only used for the evaluation of the process being executed with support of a WFMS. As any WFMS needs at least one predefined process model in order to be operable [3], there is no possibility to support the intense process modeling phase with the automatic process discovery possibilities of process mining. The development of the PO offers the possibility to change this situation. With support of the PO, the lifecycle can be rearranged in the following way (Fig. 7). The initial step consists of process execution (as usual) accompanied by manual logging, i.e., the generation of a manual log, with the PO. This phase is followed by a process mining step. Afterwards, the results of process mining possibly have to be reworked in a process remodeling phase. The benefit of the application of the PO consists of the time saving between the previous process modeling phase and the less time consuming remodeling phase.

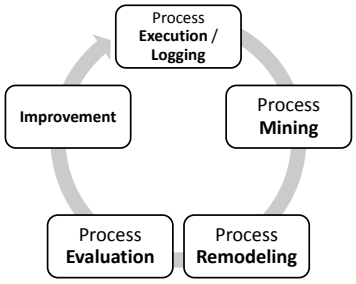


Figure 7. Adapted process lifecycle through the application of the PO

The previous modeling phase, i.e., the project of process discovery and process definition, had to be operated completely manual. The process management team had to do several interviews with agents, live observations of processes and the tracking of documents, for example. In contrast to that, process discovery with the PO is generally more automatable. Merely reworking effort is required in order to annihilate possibly occurring exceptions or execution errors. Based on the results of these first three steps, business processes can be evaluated and finally optimized.

VI. CONCLUSION AND OUTLOOK

In this paper, we discussed the problem of the availability of complete process execution event logs in order to offer automatic process model generation possibility by process mining techniques. Therefore, we presented the Process Observer (PO) project that generates manual logs and guides process participants through process execution. Like this, our project offers the possibility for the automatic generation of process models within organizations, without the availability of any information system. Process participants are encouraged to work with the PO by various process execution support functions, like the auto-suggestion of process data and dynamic recommendations of following processes. This kind of guidance during process execution is typically exclusively offered by WFMS. Our future research activity in the field of the PO will start with the development of matching methods in order to match and merge identical processes. We will also implement a module to transfer the recorded process data into the new the eXtensible Event Stream (XES) format [18]. Furthermore, we will face the problem of recording and logging processes in different granularities. This research faces one of the great challenges of process mining declared during the meeting of the IEEE Task force on process mining at the BPM conference in 2011. In order to deal with execution exceptions and wrongly logged processes, we will implement a heuristic process mining algorithm [9]. Like this, some of the manual mapping activity will be obsolete. Additionally, the control-flow mining algorithm should be featured by decision mining [4] in order to enrich the process models with decision information based upon data extensions. Furthermore, we are developing a new process discovery approach based upon explicit semantic definitions. Finally, we are looking forward to an extensive application of the PO in an organization, accompanied by a detailed documentation of the practice.

REFERENCES

[1] Zur Muehlen, M. and Ho D.: "Risk management in the BPM lifecycle". In: Business Process Management Workshops, LNCS, vol. 3812, 2006, pp. 454-466.

[2] Zairi, M.: "Business process management: a boundaryless approach to modern competitiveness". In: Business Process Management Journal, vol. 3, 1997, pp. 64-80.

[3] Van der Aalst, W., Weijters, T., and Maruster, L.: "Workflow mining: Discovering process models from event logs". In: IEEE Transactions on Knowledge and Data Engineering, vol.16 (9), 2004, pp. 1128-1142.

[4] Rozinat, A. and Van der Aalst, W.: "Decision mining in business processes". In: BPM Center Report BPM-06-10, BPMcenter.org, 2006.

[5] Van der Aalst, W. and Weijters, A.: "Process mining: a research agenda". In: Computers in Industry, vol. 53 (3), 2004, pp. 231-244.

[6] Cook, J.E. and Wolf A.L.: "Automating process discovery through event-data analysis". In: 17th International Conference on Software Engineering, Apr. 1995, Seattle, USA.

[7] Van der Aalst, W., Reijers,H., Weijters, A., Van Dongen, B., De Medeiros, A., Songa, M., and Verbeek, H.: "Business process mining: An industrial application". In: Information Systems, vol. 32 (5), 2007, pp. 713-732.

[8] Van Dongen, D., De Medeiros, A., Verbeek, H., Weijters, A., and Van der Aalst, W.: "The ProM framework: A new era in process mining tool support". In: Applications and Theory of Petri Nets, LNCS, vol. 3536, 2005, pp. 444-454.

[9] Weijters, A., Van der Aalst, W., and De Medeiros, A.: "Process mining with the heuristics miner-algorithm". Department of Technology, Eindhoven University of Technology, 2006, Eindhoven, The Netherlands.

[10] Witschel, H., Hu, B., Riss, U., Thönssen, B., Brun, R., Martin, A., and Hinkelmann, K.: "A collaborative approach to maturing process-related knowledge". In: Business Process Management, LNCS, vol. 6336, 2010, pp. 343-358.

[11] Stoitsev, T., Scheidl, S., Flentge, F., Mühlhäuser, M.: "From Personal Task Management to End-User Driven Business Process Modeling". In: Business Process Management, LNCS, vol 5240, 2008, pp. 84-99.

[12] Noy, N. and Musen, M.: "Algorithm and tool for automated ontology merging and alignment". In: Proceedings of the 17th National Conference on Artificial Intelligence, Aug. 2000, Austin, USA.

[13] Jablonski, S. and Goetz, M.: "Perspective oriented business process visualization". In: Business Process Management Workshops, LNCS, vol. 4928, 2008, pp. 144-155.

[14] Jablonski, S. and Bussler, C.: "Workflow management: modeling concepts, architecture and implementation". International Thomson Computer Press, London, 1996, ISBN 1850322228.

[15] Jablonski, S., Igler, M., and Günther, C.: "Supporting collaborative work through flexible process execution". In: 5th International Conference on Collaborative Computing, Nov. 2009, Washington DC, USA.

[16] Halleux, J.d.: "QuickGraph, Graph data structures and algorithms for .net". Last access: Mar. 2012. Available: http://quickgraph.codeplex.com..

[17] Microsoft: "Graph# - Layout algorithms and graph layout control for WPF applications". Last access: Mar. 2012. Available: http://graphsharp.codeplex.com.

[18] Verbeek, H., Buijs, J., Van Dongen, B., and Van der Aalst, W.: "XES, XESame, and ProM 6". In: Information Systems Evolution, Lecture Notes in Business Information Processing, vol. 72, 2011, pp. 60-75.

 ISBN: 978-1-61208-201-1

Fingerprint Scanners Comparative Analysis Based on International Biometric Standards Compliance

María del Carmen Prudente-Tixteco
Instituto Politécnico Nacional
Graduate School ESIME Culhuacan
México
mprudentet0900@ipn.mx

Gualberto Aguilar-Torres
Instituto Politécnico Nacional
Graduate School ESIME Culhuacan
México
gaguilar@ipn.mx

Linda Karina Toscano-Medina
Instituto Politécnico Nacional
Graduate School ESIME Culhuacan
México
ltoscano@ipn.mx

Gabriel Sánchez-Pérez
Instituto Politécnico Nacional
Graduate School ESIME Culhuacan
México
gasanchezp@ipn.mx

Abstract. **This paper discuses an analysis of the characteristics of fingerprint biometric scanners based on biometric standards compliance, taking into account of the features provided by manufacturers to conduct a comparative analysis of their physical characteristics, its certifications and standards supported. Because there is a huge fingerprint scanners variety, this paper studies only 30 devices from 13 different vendors using the characteristics obtained from the specification sheets for each device. A fingerprint device classification was performed based on the number of fingerprints that each device can capture, for example 4-4-2 devices or a single fingerprint capture and type of biometric standard compliant. As a result of this devices classification, 73% of them comply with at least one biometric standard or certification and 27% of them do not specify if they meet some kind of standard; however, they have the requirements to be considered for the acquisition of a fingerprint image.**

Keywords - fingerprint scanners; Biometric Standards; physical characteristics.

I. INTRODUCTION

A biometric system is an automated system to capture biometric sensor data from a user, extract feature data from that processed acquired data, compare the processed feature data with that contained in one or more biometric templates, decide how well they match and indicate whether or not an identification or verification of identity has been achieved [1][6][8].

Biometric systems are used to provide greater security for systems that are used as mechanisms for identifying and verifying people. Biometric sensors are devices that are located within a biometric system, which have the function of acquire data or images for biometric feature extraction.

The biometric technologies designers have the need to work with biometric standards that define the characteristics and minimum requirements to develop devices and appropriate models for the biometric data management for public and private entities, law enforcement and government areas.

The fingerprint scanner must produce images that exhibit good geometric fidelity, sharpness, detail rendition, gray-level uniformity and gray-scale dynamic range, with low noise characteristics. The images must be true representations of the input fingerprints, without creating any significant artifacts, anomalies, false detail, or cosmetic image restoration effects [2].

According to the International Standard Organization, the minimal requirements that a fingerprint scanner must meet are: image resolution, size, gray level color range, sample rate, light intensity and signal to noise ratio [1].

The requirements provide criteria to guarantee the image quality of fingerprint scanners and printers that input fingerprint images or generate fingerprint images [2]. Electronic images must be of sufficient quality to allow for: (1) conclusive fingerprint comparisons (identification or non-identification decision), (2) fingerprint classification, (3) automatic feature detection; and (4) overall Automated Fingerprint Identification System (AFIS) search reliability [8].

The paper is organized as follows. Section 2 relates on general international biometric standards that are contemplated for realization the analysis. Section 3 refers to requirements that were considered for the analysis based in international biometric standards. In Section 4, devices that are considered for analysis are listed, and, in Section 5, there are the results of comparative analysis between the devices according to international biometric standards compliant.

II. BIOMETRIC STANDARDS

The fingerprint scanners, in addition to meeting the minimum requirements for use in various applications, support different features provided by international biometric standards like ISO/IEC 19794-4, and ANSI/INCITS 381, this allows to meet interoperability between systems [6][7].

 ISBN: 978-1-61208-201-1

International biometric standards that support biometric fingerprint scanners are mainly focused on scanner physical characteristics, data transmission characteristics, management for biometric exchange formats and fingerprint image quality.

Some international biometric standards are described below that which are used to define the devices technical specification.

Electronic Fingerprint Transmission Specification (EFTS) by Federal Bureau of Investigation (FBI) the purpose of this document is to specify certain requirements to which agencies must adhere to communicate electronically with the FBI's IAFIS (Integrated Automated Fingerprint Identification System), electronic communications do not include fingerprints, and the requirements [2].

Personal Identity Verification (PIV) , this specification apply to fingerprint capture devices which scan and capture at least a single fingerprint in digital, softcopy form [3].

FIPS PUB 201: Personal Identity Verification (PIV) of Federal Employees and Contractors, this standard specifies the architecture and technical requirements for a common identification standard for Federal employees and contractors. The overall goal is to achieve appropriate security assurance for multiple applications by efficiently verifying the claimed identity of individuals seeking physical access to federally controlled government facilities and electronic access to government information systems [4].

BioAPI Specification or ISO/IEC 19784-1, this specification defines the Application Programming Interface (API) and Service Provider Interface (SPI) for standard interfaces within a biometric system that support the provision of that biometric system using components from multiple vendors. It provides interworking between such components through adherence to this and to other International Standards. The BioAPI specification is applicable to a broad range of biometric technology types. It is also applicable to a wide variety of biometrically enabled applications, from personal devices, through network security, to large complex identification systems [5].

ISO/IEC 19794-4 and ANSI 381, this standards specifies a data record interchange format for storing, recording and transmitting the information from one or more finger or palm image areas. There have a section of image acquisition requirements are made aware of the minimum requirements for selected image acquisition settings level desired [6][7].

ISO/IEC 19794-2 and ANSI 378, define interoperability is based on the definition of the rules or standards of minutiae extraction of the finger and the formats of the records that are common in many matching procedures. The minutiae are points located in a fingerprint image where a friction ridge begins, ends or splits into two or more peaks, these features can be used to identify a person [8][9].

Data Format for the Interchange of Fingerprint, Facial, & Other Biometric Information ANSI/NIST-ITL this standard defines the content, format, and units of measurement for the exchange of fingerprint, palmprint, facial/mugshot, scar mark & tattoo (SMT), iris, and other biometric sample information that may be used in the identification or verification process of a subject. The information consists of a variety of mandatory and optional items, including scanning parameters, related descriptive and record data, digitized fingerprint information, and compressed or uncompressed images [10].

III. REQUIREMENTS ANALYSIS OF FINGERPRINT SCANNERS

There are requirements that must be considered by providers of biometric technologies for the development of a fingerprint scanner.

The fingerprint comparison process requires a high fidelity image without any banding, streaking or other visual defects. Finer detail such as pores and incipient ridges are needed since they can play an important role in the comparison. Additionally, the gray-scale dynamic range must be captured with sufficient depth to support image enhancement and restoration algorithms [8].

Binary and grayscale fingerprint images to be exchanged shall be captured by an AFIS, live-scan reader, or other image capture device operating at a specific native scanning resolution. The minimum scanning resolution for this capture process shall be 19.69 ppmm plus or minus 0.20 ppmm (500 ppi plus or minus 5 ppi). Scanning resolutions greater than this minimum value and with a device tolerance of plus or minus 1% may be used [10].

Table I shows the preferred capture sizes, applicable to live scan systems. Scanner capture dimensions should never be less than 90% of those given.

TABLE I. PREFERRED CAPTURE SIZES

	Preferred Width (inches)	**Preferred Height (inches)**
Roll finger	1.6	1.5
Plain thumb	1.0	2.0
Plain 4-fingers (sequence check)	3.2	2.0
Plain 4-fingers (identification flat)	3.2	3.0

Table II shows maximum image dimensions of fingerprints [10].

 ISBN: 978-1-61208-201-1

TABLE II. MAXIMUM IMAGE DIMENSIONS

Finger position	Width		Length	
	(mm)	(in)	(mm)	(in)
Right or left thumb	40.6	1.6	38.1	1.5
Right or left index finger	40.6	1.6	38.1	1.5
Right or left middle finger	40.6	1.6	38.1	1.5
Right or left ring finger	40.6	1.6	38.1	1.5
Right or left little finger	40.6	1.6	38.1	1.5
Plain right or left thumb	25.4	1.0	50.8	2.0
Plain right four fingers	81.3	3.2	76.2	3.0
Left & right thumbs	81.3	3.2	76.2	3.0

The dynamic range define the image grayscale shall be encoded using the agreed precision necessary to meet the dynamic range requirement for a specific application. Grayscale finger image data may be store, recorded, or transmitted in either compressed or uncompressed form. Using a pixel depth of 8 bits (256 grayscale levels) each shall contained a single byte [6].

The ISO/IEC 19794-4 sets the standards for the acquisition of a fingerprint image by defining the specific requirements that must be considered for the data exchange format based on a biometric fingerprint image. Table III shows the levels of viewing requirements to the acquisition of a fingerprint image described by the standard ISO / IEC 19794-4.

TABLE III. REQUIREMENTS ACQUISITION OF A FINGERPRINT IMAGE

Levels set.	Resolution scanner.	Intensity of pixels.	Range dynamic. (grayscale)	Certification
30	500	8	80	None
35	750	8	100	None
31	500	8	200	EFTS/F

IV. FINGERPRINT SCANNERS

The biometric scanners vendors give the technical specification datasheet when a fingerprint scanner is sold. The main technical specifications are the following: resolution, image size, gray level, certifications, and standards compliance.

In some fingerprint scanner specification datasheets also give information about the percentage of geometric distortion.

A fingerprint devices classification was performed based on the number of fingerprints that each device can capture.

Tables IV, V and VI show the description of the devices that were selected for analysis depending on your fingers the number of images that can be acquired in a single capture.

TABLE IV. A SINGLE FINGER SCANNERS

Vendors	Product	Image size
Biometrika	Hiscan [11]	500x500
Cogent	CSD 200 [12]	480x320
Cogent	CSD 330 [13]	500x500
CrossMatch	Verifier 300 [14]	600x600
Dakty	NAOS-A [15]	236x354
Digital Persona	U.are.U 4000 [16]	390x355
Digital Persona	U.are.U 4500 [17]	390x355
Futronic	FS80 [18]	480x320
Futronic	FS88[19]	480x320
Futronic	FS90[20]	300x440
Lumidigm Mercury	M301 Fingerprint Reader [21]	342x274
Microsoft	Fingerprint Reader [22]	355x390
SecuGen	Hamster Plus [23]	260x300
SecuGen	Hamster IV [24]	258x336
SecuGen	ID USB SC/PIV [25]	258x336
Suprema	RealScan-S [26]	600x600
Suprema	SFR300S[27]	288x288
Suprema	BioMini & SDK [28]	288X320
Identix	DFR 2100 [29]	600x600

 ISBN: 978-1-61208-201-1

TABLE V. DUAL FINGER SCANNER

Vendors	Product	Image size
Cogent	CSD 450 [30]	800x750
CrossMatch	Verifier 310 [31]	900x900
CrossMatch	Verifier 320 [32]	800x750
Futronic	FS50 [33]	800x750
Suprema	RealScan-D [34]	900x900

TABLE VI. TENPRINT SCANNERS

Vendors	Product	Image size
ARH	AFS 510 Live Scanner [35]	1600x1500
CrossMatch	L_SCAN_Guardian [36]	1600x1500
CrossMatch	LScan500 [37]	1600x1500
Futronic	FS60 [38]	1600x1500
Mantra Softech	MFS500 [39]	1600x1500
Suprema	RealScan-10 [40]	1600x1500

V. RESULTS

The physical characteristics given by vendors that all devices meet are shown in Table VII:

TABLE VII. PHYSICAL CHARACTERISTICS

Image resolution	500 ppi
Pixel depth	8 bit
Grayscale	256

All fingerprint scanners considered for this comparative analysis meet the requirements acquisition of a fingerprint image taken by the ISO / IEC 19794-4 located on a level 31 set for the data exchange format based biometric the fingerprint image.

Some vendors do not specify information about the use of international biometric standards, but meet the requirements for the acquisition of a fingerprint image as stipulated in ISO / IEC 19794-4 as shown in Table VIII.

TABLE VIII. FINGERPRINT SCANNERS THAT NOT REPORT ANY BIOMETRIC STANDARDS

Firm	Product
Dakty	NAOS-A
Digital Persona	U.are.U 4000
Digital Persona	U.are.U 4500
Futronic	FS80
Futronic	FS90
Microsoft	Fingerprint Reader
Suprema	RealScan-S
Suprema	SFR300S

A. *Physical Characteristics.*

The devices that have a certification or meet some international biometric standards related to the minimum requirements of their physical characteristics are only 18 devices; Table VIII shows the devices that meet certification or international biometric standards.

The FBI EFTS Appendix F and PIV-071006 standards specify parameters that devices must meet to guarantee a correct acquisition of the fingerprint image. The requirements that have in common both standards are:

- Linearity
- Geometric Accuracy
- Spatial Frequency Response
- Signal-to-Noise Ratio
- Fingerprint Image Quality

The devices that can be used in identity and verification information systems into federal governments strictly satisfy FIPS 201 PIV certification.

In Table IX are listed the devices that meet or have a can fit certification standards which refer to physical characteristics.

Fig. 1, shows the total number of devices that meet each standard considered for the analysis concerning the physical characteristics.

 ISBN: 978-1-61208-201-1

TABLE IX. DEVICES THAT MEET PHYSICAL CHARACTERISTICS REQUIRED BY INTERNATIONAL BIOMETRIC STANDARDS.

Product	IQS. Appendix F of the EFTS.	FBI PIV-071006	FIPS 201 PIV
AFS 510	certification		
Hiscan		certification	
CSD 200		certification	Certification
CSD 330		certification	Certification
CSD 450		certification	Certification
Verifier 310		certification	
Verifier 320	certification	certification	
L_SCAN_ Guardian	certification		
LScan500	certification		
FS50		certification	certification
FS88		certification	comply
FS60	certification		
Hamster IV		certification	comply
ID USB SC/PIV			certification
RealScan-D	certification		
RealScan-10	certification		
BioMini & SDK		certification	
DFR 2100		certification	

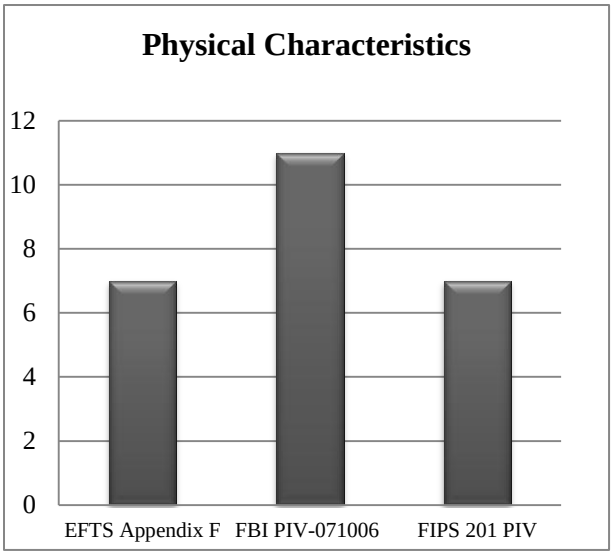


Figure 1. Number of fingerprint scanners that satisfy international biometric standards about physical characteristics.

B. Biometric Data Interchange Formats

The international biometric standards that satisfy with biometric data interchange formats are: ANSI/NIST-2000, ISO/IEC 19794-2/4 and ANSI 381/378.

The devices that satisfy the international biometric standard described above are 12; Table X and Fig. 2 show the number of devices that comply with each of the different standards.

The scanners that comply with the biometric data interchange formats are used for biometric acquisition of register fingerprint image and minutia data.

TABLE X. INTERNATIONAL BIOMETRIC STANDARDS DEVICES THAT MEET THE BIOMETRIC DATA INTERCHANGE FORMAT.

Product	ANSI/NIST-ITL-1-2000	ISO/IEC 19794-2/4	ANSI 381/378
AFS 510 Live Scanner	Comply	comply	
CSD 330		certification	
Verifier 300	Comply		
M301 Fingerprint Reader			comply
MFS500		comply	
Hamster Plus		comply	comply 378
Hamster IV		comply	comply 378
BioMini & SDK		comply 2	comply 378

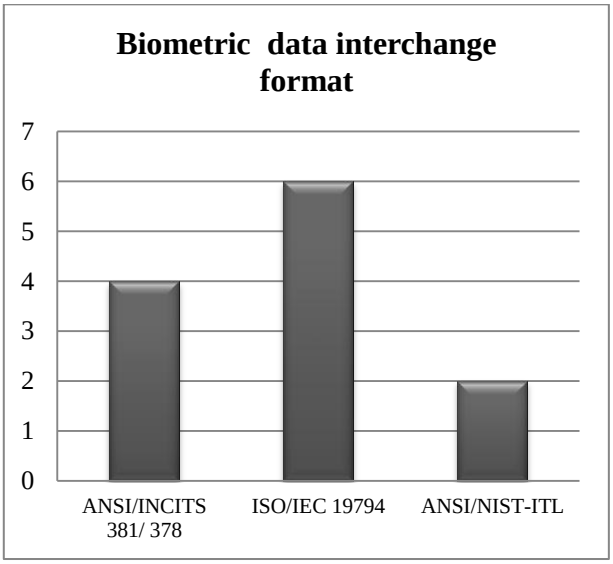


Figure 2. Number of fingerprint scanners that comply with biometric data interchange format standards.

 ISBN: 978-1-61208-201-1

C. Others

Some devices meet with the BioAPI standard, having a common structure for operation with different interfaces in a biometric system, these devices as show in Table XI.

TABLE XI. ISO/IEC 19784-1:2005

Firm	Product
ARH	AFS 510 Live Scanner
SecuGen	Hamster Plus
SecuGen	Hamster IV

D. General

In the comparative analysis of biometric fingerprint scanners, 30 datasheets specification were revised, where 73% compliance with any international biometric standard or have a certification.

Fig. 3 shows the number of devices that specify compliance with at least one biometric standard and the number of devices that do not specify.

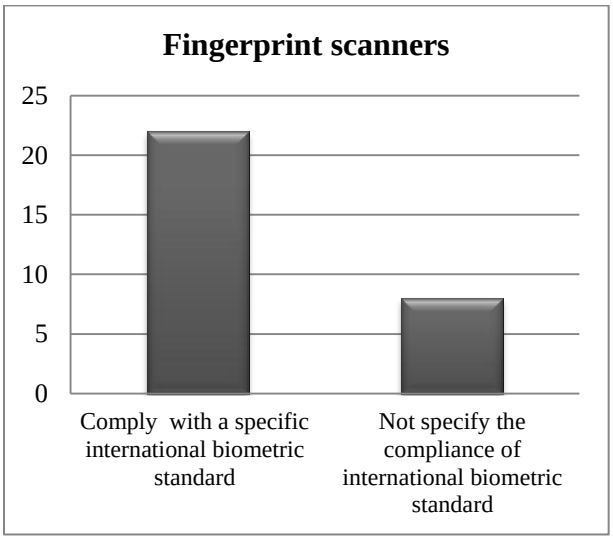


Figure 3. Devices number that satisfy or not with an international biometric standard.

From the 22 scanners that satisfy some certification or international biometric standard, the 63.63% devices complied with some physical characteristics standards, 18.18% with any biometric data interchange formats and the other 18.18% complied with both.

Fig. 4 shows the number of devices that satisfy biometric standard type of that were considered for this analysis.

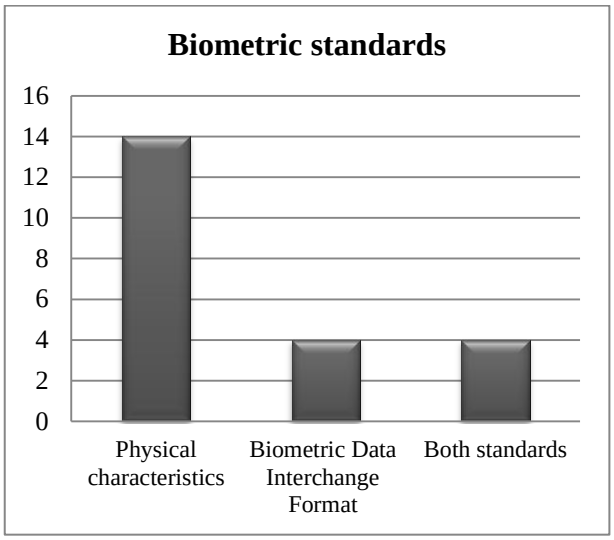


Figure 4. Devices that satisfy international biometric standards.

VI. Conclusion and Future Work

A comparative analysis was performed in relation to standards that meet the data acquisition devices biometric fingerprint taking into account the information provided in the specification datasheets for each of them. Most devices have a certification in relation to physical characteristics.

Devices that do not present information on compliance with standards meet the minimum requirements to be considered for the acquisition of a fingerprint image and are used in conjunction with software development kits, which use or comply with any type of international biometric standard for its use in various applications.

The international biometric standards are very important due they mark the minimum requirements that must take into account by devices manufacturers.

These minimum requirements must also be taken into account by people who use the scanner in various biometric applications. There are devices that meet the minimum requirements for physical characteristics, as well as compliance to unveil common structures in the biometric data interchange formats for interoperability in biometric systems.

As future work, tests will be done to devices that were considered for this analysis to verify compliance with the requirements of international biometric standard.

 ISBN: 978-1-61208-201-1

VII. REFERENCES

[1] International Standard ISO/IEC 19794-1:2005, "Biometric data interchange formats – Part 1: Framework", Information technology p. 3, 11.

[2] Federal Bureau of Investigation, Criminal Justice Information Services (CJIS), "Electronic Fingerprint Transmission Specification, Appendix F", May 2, 2005.

[3] Federal Bureau of Investigation, "Personal Identity Verification (PIV)". Image Quality Specifications for Single Finger Capture Devices, July 2006.

[4] FIPS PUB 201: Personal Identity Verification (PIV) of Federal Employees and Contractors", p. 1, 5, 15, March 2006.

[5] International Standard ISO/IEC 19784-1:2005, "Biometric application programming interface –Part 1: BioAPI specification", Information technology, p.1, 2005.

[6] International Standard ISO/IEC 19794-4:2005 "Biometric data interchange formats – Part 4: Finger Image", Information technology pp. 4-8, 2005.

[7] Standard ANSI INCITS 381:2004, "Information technology – Finger image format for data interchange", pp. 1-5, 2004.

[8] International Standard ISO/IEC 19794-2:2005 "Biometric data interchange formats – Part 2: Finger minutiae data", Information technology, pp. 1-4, 2005.

[9] Standard ANSI INCITS 378:2004, "Information technology -Finger minutia format for data interchange", pp. 1-10, 2004.

[10] ANSI/NIST ITL 1-2007, "Data Format for the Interchange of Fingerprint, Facial, & Other Biometric Information", pp. 25-40, April 20, 2007.

[11] Biometrika, "HiScan Fingerprint Scanner 1x1," [retrieved: January, 2012].

[12] 3M COGENT Ing, "CSD200, Single-digit Optical Scanner," [retrieved: December, 2011].

[13] 3M COGENT Ing, "CSD330, Single-digit Fingerprint Scanner," [retrieved: January, 2012].

[14] Cross Match Technologies, Inc. Verifier® 300 LC 2.0 "Single Finger Scanner with USB 2.0 Interface", 2008 [retrieved: January, 2012].

[15] Dakty GmbH, "Fingerprint NAOS-1," April 2009, [retrieved: January, 2012].

[16] DigitalPersona, U.are.U 4000B Reader "USB Fingerprint Reader," [retrieved: January, 2012].

[17] DigitalPersona, U.are.U 4500B Reader "USB Fingerprint Reader," [retrieved: January, 2012].

[18] Futronic's, "FS80 USB2.0 Fingerprint Scanner," [retrieved: January, 2012].

[19] Futronic's, "FS88 FIPS201/PIV Compliant USB2.0 Fingerprint Scanner," [retrieved: January, 2012].

[20] Futronic's, "FS90 USB2.0 Mini Fingerprint Scanner," [retrieved: December, 2011].

[21] Lumidigm, Inc., Venus Series Biometric Sensors Multispectral Fingerprint Imagers, "M301 Fingerprint Reader," [retrieved: December, 2011].

[22] Microsoft, "Fingerprint Reader v1.0", 2007 [retrieved: December, 2011].

[23] SecuGen Biometric Solution, "Hamster Plus", Hamster IV ID USB SC/PIV, 2011 [retrieved: January, 2012].

[24] SecuGen Biometric Solution, "Hamster IV", 2011 [retrieved: December, 2011].

[25] SecuGen Biometric Solution, "ID USB SC/PIV", 2011 [retrieved: December, 2011].

[26] Suprema, "Fingerprint Scanner RealScan-S", 2011 [retrieved: January, 2012].

[27] Suprema, "Fingerprint Scanner SFR300S", 2007 [retrieved: January, 2012].

[28] Suprema, "BioMini & SDK", 2011 [retrieved: January, 2012].

[29] Identity Solutions Biometrics Division, "DFR 2100/2130 Single Finger Reader", 2008 [retrieved: December, 2011].

[30] 3M COGENT Ing, "CSD450 Dual-Digit Optical Scanner," [retrieved: December, 2011].

[31] Cross Match Technologies, "Inc. Verifier® 310 LC Single/Dual Finger Scanner," [retrieved: January, 2012].

[32] Cross Match Technologies, Inc. Verifier® 320 LC "Two Finger Scanner for Flats and Rolled Prints," [retrieved: December, 2011].

[33] Futronic FS50 FIPS201/PIV Compliant USB2.0 "Two Finger Scanner," [retrieved: January, 2012].

[34] Suprema, "Fingerprint Scanner RealScan-D", Portable Dual Finger Live Scanner, [retrieved: December, 2011].

[35] ARH Inc. "Professional fingerprint scanning AFS 510 Live Scanner," [retrieved: January, 2012].

[36] Cross Match Technologies, Inc., "L_SCAN_Guardian Livescan and Fingerprint Enrollment System," [retrieved: December, 2011].

[37] Cross Match Technologies, Inc. "L SCAN® 500P "tenprint/palmprint," [retrieved: December, 2011].

[38] Futronic, "FS60 EBTS/F Certified ID Flat Fingerprint Scanner," [retrieved: December, 2011].

[39] Mantra Softech India Pvt. Ltd. MFS500 "4+4+2 Fingerprint Biometrics Ten Print Live Scanner," [retrieved: December, 2011].

[40] Suprema, Scanner RealScan-10, "Compact scanner for ten fingerprints," [retrieved: December, 2011].

 ISBN: 978-1-61208-201-1

Constructing Context-based Non-Critical Alarm Filter in Intrusion Detection

Yuxin Meng
Department of Computer Science
City University of Hong Kong
Hong Kong SAR, China
ymeng8@student.cityu.edu.hk

Wenjuan Li
Computer Science Division
Zhaoqing Foreign Language College
Guangdong, China
wenjuan.anastatia@gmail.com

***Abstract*—Currently, intrusion detection systems (IDSs) are being widely deployed in various computer networks aiming to detect all kinds of attacks. But the major problem is that a large amount of alarms are generated during their detection and most of them are non-critical alarms. This issue greatly increases the analysis workload and reduces the effectiveness of an IDS. We argue that this bottleneck stems primarily from the lack of contextual information to the intrusion detection systems. To mitigate this issue, we propose an architecture of context-based non-critical alarm filter to help filter out these non-critical alarms. In particular, our alarm filter consists of an *indexing component* to link input alarms to corresponding contextual information, an *analysis engine* aims to filter out non-critical alarms according to contextual information and a *monitor engine* to update index values. In the evaluation part, we explored the initial effectiveness of our non-critical alarm filter in a deployed network environment. The experimental results show that our alarm filter is promising and effective in filtering out non-critical alarms.**

***Keywords*-Intrusion detection; Network security; Non-critical alarm filter; Context-based system**

I. INTRODUCTION

Computer systems have become vulnerable to attacks with the rapid development of networks. According to the latest released *Internet Security Threat Report* from Symantec [7], the trend of malware attacks is increasing significantly. More than 286 million unique variants of malware were discovered and up to 6,253 new vulnerabilities were recorded.

To address these network threats, intrusion detection systems (IDSs) [1, 2] have been widely deployed into different kinds of organizations (e.g., assurance company) aiming to safeguard computer security and network environment. The security administrators can rely on them to detect and identify attacks and prevent future uses of known exploits and vulnerabilities. Moreover, the use of intrusion detection systems is a powerful complementary solution to firewall technology through defending against attacks and suspicious network traffic that are missed by the firewall.

In general, there are two traditional types of intrusion detection systems according to their detection techniques. One is the *signature-based intrusion detection systems* (also called *rule-based* IDS), which are mainly based on attack signatures to detect various attacks and threats. This kind of detection systems has to maintain a signature database and keep updating it to the latest version periodically. The signatures are usually extracted from the previously detected malicious network packets, therefore, the signature-based IDS can only identify known attacks. Take Snort [6, 12] as an example, this lightweight rule-based network intrusion detection system detects attacks by monitoring and analyzing network packets (e.g., UDP, TCP, IP). The common Snort rule format is shown as follows:

Action-type protocol-type Source-ip Source-port -> Destination-ip Destination-port (content:"attack signature"; msg:"attack msg";)

The other type of IDSs is called *anomaly-based intrusion detection systems*. Compared to the signature-based IDS, the anomaly-based IDS has the capability of identifying novel attacks. In reality, the anomaly-based approach will pre-build a normal profile to model the normal network traffic by training relevant systems with machine learning algorithms. During the detection, this approach aims to detect deviations through comparing current events with the normal profile. In actual deployment, the signature-based approach is more prevalent than the anomaly-based method in that the false alarm rate of anomaly-based systems is significantly higher than the signature-based detection systems since it is very hard to build a good normal profile in most cases [3].

Problem. Although the IDSs are proven to be effective in detecting network attacks, their generated large number of alarms greatly increase the analysis workload for a security officer. What is worse, most of these alarms are false alarms or non-critical alarms [8, 9]. The false alarm rate (or *non-critical alarm rate*) is a major limiting factor in encumbering the high performance of an IDS [5]. This issue primarily stems from the fact that current IDS detects not only the intrusions, but also unsuccessful attack attempts. Whereas it is hard for an IDS to decide the situation of an attack attempt (whether it is a successful attack or not), it has to report all detected attack attempts to security officers with the purpose of reducing security risk [4]. In this case, it is a big challenge for a security officer to analyze the information of real attacks without discarding all non-critical alarms since these non-critical alarms can greatly make negative effects on the final analysis results.

To more explicitly illustrate this problem in this paper, we

 ISBN: 978-1-61208-201-1

learn from the previous research work [10] and provide the definition of non-critical alarms as below:

Definition of non-critical alarms. A non-critical alarm is either not related to a malicious activity or not related to a successful attack. In other words, a non-critical alarm is either a false positive or a non-relevant positive.

Contributions. To improve the performance of IDSs by filtering out the non-critical alarms, we advocate that making the IDSs be aware of the contextual information of their deployed contexts is a promising method to achieve the improvement [10, 11]. In this paper, therefore, we develop and construct a context-based non-critical alarm filter to help filter out the non-critical alarms aiming to improve the effectiveness of IDSs and reduce the workload of a security officer. In particular, our proposed context-based non-critical alarm filter consists of three major components: namely, an indexing component, an analysis engine and a monitor component. For the indexing component, its main function is to link the input alarms to corresponding look-up tables which consist of contextual information. The analysis engine aims to compare the contextual information with the input alarms to determine whether the input alarms are critical or not. The monitor component is responsible for recording alarm information and updating the index values in the indexing component periodically. In terms of the indexed contextual information, our alarm filter can be adaptive to the specific network contexts.

To explore the feasibility and effectiveness of our context-based non-critical alarm filter, we implemented and evaluated this alarm filter under an established network environment with Snort. During the experiment, we converted all original Snort alarms to the type of *contextual alarms* (see Section IV). Then our alarm filter analyzed the *contextual alarms* by comparing to the stored contextual information and finally output the critical alarms. The initial experimental results show that our alarm filter is encouraging and effective in our network settings.

The rest of this paper is organized as follows: in Section II, we describe research papers that relate to false alarm reduction such as alert verification, alert correlation and machine learning based methods; Section III illustrates the architecture of our context-based non-critical alarm filter and gives an in-depth description of each component; Section IV presents our experimental methodology and shows the experimental results; limitations and future work are presented in Section V; finally, Section VI states our conclusion.

II. Related Work

A variety of solutions have been proposed aiming to reduce the number of non-critical alarms in intrusion detection. These efforts fall roughly into two general folders: *indirect reduction* such as *IDS signature enhancement*; and *direct reduction* including *alert correlation*, *alert verification* and kinds of *machine learning based approaches*.

For the signature-based IDSs, the false alarm rate (or *non-critical alarm rate*) depends heavily on the capability of their signatures. Therefore, *signature enhancement* is regarded as a promising approach to control the false alarm rate. Sommer and Paxson [14] proposed and designed a type of *contextual signature* as an improvement for the string-based signature matching. They then developed a signature engine for Bro as follows: low-level context by using regular expressions and high-level context by taking advantage of the semantic information from Bro's protocol analysis. In addition, Cost *et al.* [13] proposed *Vigilante*, a new approach to create a signature for the execution path of worms under an end-to-end environment. Followed by above work, Brumley *et al.* [15] improved *Vigilante* and provided the definition of *vulnerability signature* that is a representation for the set of inputs to satisfy a specific vulnerability condition. In the evaluation, they showed that this new type of signature achieved an improvement over existing signatures.

To directly reduce the non-critical alarms, the common approaches are *alert correlation*, *alert verification* and building *alarm filters* by using *machine learning algorithms*.

Debar and Wespi [16] proposed an aggregation and correlation algorithm to manage IDS alarms and relate these alarms together in order to output a condensed view of the reported security issue, and they also designed an aggregation and correlation component to handle alarms which were generated by probes. Then, Cuppens and Miege [17] introduced an architecture of CRIM, a cooperative module for IDSs to manage, cluster, merge and correlate alarms. The function of this module is to recognize alarms and create a new alarm from various alarms. The method of *alert verification* is to help determine whether an attack is successful or not, which is regarded as a pre-processing step for alert correlation in achieving good correlation results [19]. Gagnon *et al.* [10] evaluated the feasibility of using target configuration (i.e., operating system and applications) as contextual information for identifying non-critical alarms. Several other work (e.g., [21], [20], [32]) further showed that the alert verification improved the quality of alerts by effectively verifying the results of intrusion attempts and enhanced the performance of alert correlation.

Another widely used method in filtering out non-critical alarms is constructing an alarm filter through using machine learning algorithms. Pietraszek [31] described an adaptive alert classifier based on an analyst's feedback to help reduce false positives by using machine learning techniques. This classifier could discard alerts in terms of their classification confidence to reduce the workload of an analyst. Law and Kwok [18] proposed a method to decrease the number of false alarms by using a KNN classifier (*k-nearest-neighbor classifier*). Alharbt and Imai [23] illustrated an algorithm by using continuous and discontinuous sequential patterns to detect abnormal alarms. Davenport *et al.* [26] implemented a support vector machine to control false alarms.

 ISBN: 978-1-61208-201-1

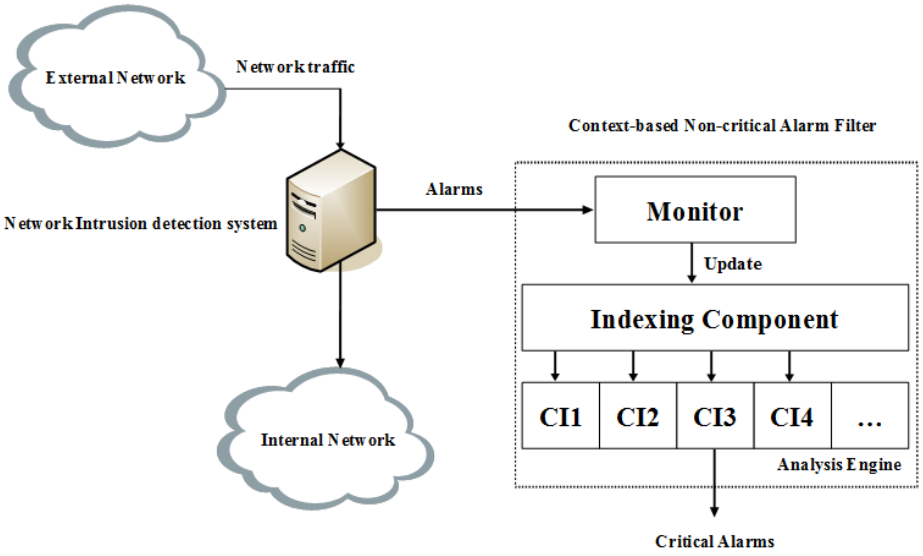


Figure 1. The architecture of context-based non-critical alarm filter with the deployment of network intrusion detection system.

Our approach is related to alert verification that aims to filter out non-critical alarms by considering the contextual information. The previous work (e.g., [10], [20]) was mainly explored the feasibility of alert verification while our work towards constructing a non-critical alarm filter. We acknowledge that our work is based on the previous results that it is appropriate and feasible to combine contextual information such as OS information and applications with IDS alarms. Differently, our work further develop this method in practice and towards automating this approach.

III. Context-based Non-critical Alarm Filter

In this section, we illustrate the architecture of our proposed context-based non-critical alarm filter. The alarm filter mainly contains three components: an indexing component, an analysis engine and a monitor component. A high-level diagram with these major components of this architecture is demonstrated in Fig. 1.

As illustrated in Fig. 1, the network intrusion detection system is deployed between an external network and an internal network in detecting network attacks by examining network packets. Its generated alarms are forwarded into our context-based non-critical alarm filter. There are three main components in the alarm filter: a monitor, an indexing component and an analysis engine. First of all, the *monitor component* records both *source IP address* and *destination port number* of an input alarm into a database and updates the indexing component periodically. Then the NIDS alarms are all forwarded to the *indexing component* in searching for the contextual information according to their index values. Finally, the *analysis engine* specifically compares the input alarms with relevant contextual information to identify whether the input alarms are critical or not. In addition, the non-critical alarms can be discarded or stored in another database for back-up and future analysis.

In the next three subsections, we give an in-depth description of these three major components respectively.

A. Monitor Component

The main task of the monitor is to collect statistical data and to update the index values in the indexing component. In this work, we use *source IP address* and *destination port number* as the index values. The construction of the monitor is shown in Fig. 2.

According to Fig. 2, when an alarm arrives, the monitor component first records its source IP address and destination port number into a database. The database is responsible for storing the source IP addresses and destination port numbers for all input alarms and updating the index values in the indexing component periodically. After recording the index values, then the input alarms are forwarded to the indexing component without any modification.

Based on the above descriptions, the database (as shown in Fig. 2) mainly contains two items: *source IP address* and *destination port number*. The major operations of the database are described as below.

- If the source IP address of the input alarm is brand new, then the database will create a new value corresponding to this IP address under the *item of source IP address*. For the new source IP address, the database can directly record its destination port number in the *item of destination port number*.
- While if the source IP address has been logged before, then the database will record the new *updating date* for this IP address. For the logged source IP address, the database has to check the destination port number.

 ISBN: 978-1-61208-201-1

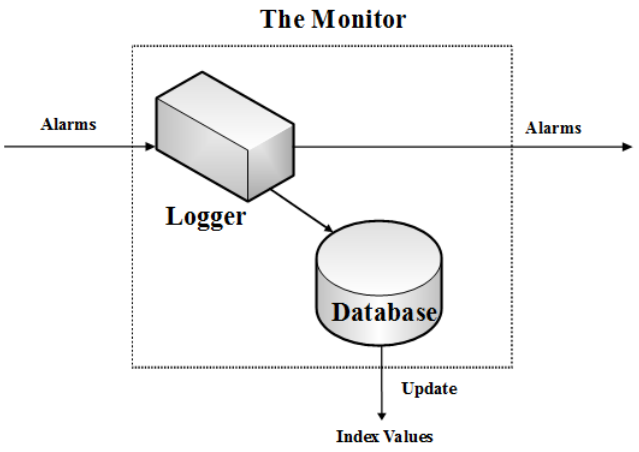


Figure 2. The construction of the monitor component in our alarm filter.

- If the destination port number has not been logged, then the database will create a new value corresponding to this destination port number under the *item of the destination port number*
- If the the destination port number has been logged, then the database will record the new *updating date* for this destination port number.

According to the specific *updating date* (which is determined by an administrator), the database can delete a source IP address if it is too dated to reduce the list length of the source IP address. Therefore, the database can be up-to-date and be adaptive to the alarm changes in real environment.

B. Indexing Component

The purpose of the indexing component[1] is to category incoming alarms based on their IP addresses and port numbers. There are two index items: *source IP address* and *destination port number*. Our scheme maintains a *complete CI database* that stores all available contextual information. In this component, the contextual information can be indexed in terms of recorded alarms' IP addresses and port numbers. For example, if an alarm was matched in the previous comparison, then its source IP address and destination port number will be recorded and be linked to that contextual information.

When alarms pass through the monitor and arrive at this component, the component will first check the *source IP addresses* of the input alarms in terms of the look-up table. If a match is identified, then the component will look for the *item of destination port number* in the look-up table and try to find another match. If a matched *destination port number* is also detected, then the relevant alarms will be forwarded to the analysis engine based on the above two items.

Otherwise, if a dis-match is either identified in the *item of source IP address* or in the *item of destination port number*, the relevant alarms will be regarded as *fresh alarms* (which have not been logged before) and have to be compared with the *complete CI database*. After the comparisons, the *source IP address* and *destination port number* of these *fresh alarms* will be recorded.

[1]In fact, this component can be incorporated into the analysis engine in real deployment, but it provides the key connection between the monitor and the analysis engine. Due to its importance, we consider it as a major component in the architecture of our alarm filter.

C. Analysis Engine

The analysis engine aims to compare the input alarms with relevant contextual information. The contextual information is the key element to our alarm filter, we mainly consider two major types: *Networking features* and *Target configuration*.

- *Networking features* consist of many different kinds of network information such as network topology, protocol specifications. These features can reflect the characteristics of distinct network environments.
- *Target configuration* usually refers to the information obtained from operating systems or applications. The information is used to help determine whether a target system is vulnerable to a given attack. For example, a Windows-based virus or worm cannot be running under a Linux system.

The basic information of known exploits can be extracted from various vulnerability databases such as Security Focus [29], National Vulnerability Database [24], Common Vulnerabilities and Exposures [27], Open Source Vulnerability Database [28]. What is more, the use of scanners [22] is an alternative if the information is not available in these vulnerability databases.

In this work, we use the *operating system (OS)* and *application (APP)* as the contextual information in the analysis engine. The evaluation steps are listed as follows:

1) If the target OS is marked as non-vulnerable to this exploit, then relevant alarms are non-critical alarms.
2) If the target APP is marked as non-vulnerable to this exploit, then relevant alarms are non-critical alarms.
3) If the target OS is marked as vulnerable to this exploit, then relevant alarms are *potential* critical alarms.
4) If the target APP is marked as vulnerable to this exploit, then relevant alarms are *potential* critical alarms.
5) If 3) and 4) are both fulfilled, then relevant alarms are regarded as critical alarms.

IV. Evaluation

In this section, we constructed an experimental network environment by using Snort (version 2.9.0.5) [6], Wireshark [30] and packet generator [25] to evaluate the performance of our context-based non-critical alarm filter. The network environment is illustrated in Fig. 3.

As shown in Fig. 3, the network traffic goes through from the source network (Internet) to the target network (Internal

 ISBN: 978-1-61208-201-1

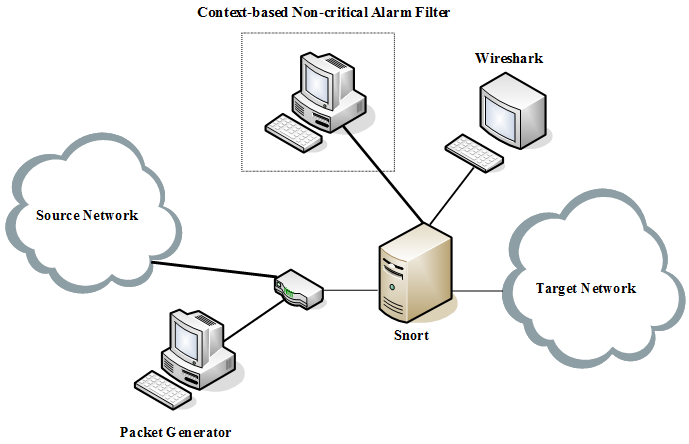


Figure 3. The deployment of experimental environment.

network). The Snort is deployed between these two networks aiming to identify attacks by checking the network packets. Our context-based non-critical alarm filter is deployed close to the Snort and all the Snort alarms will be forwarded to our alarm filter. The Wireshark is used to record the network packets and provide statistical data for analyzing experimental results.

A. Experimental Methodology

In the experiment, we use the Snort alarms as the basis for our evaluation but these original alarms do not contain any contextual information. In this case, we develop the type of *contextual alarms* by adding the *application and OS information* to the original Snort alarms before forwarding them to the analysis engine. The conversion is completed by an additional module which is designed to convert the original Snort alarms to *contextual alarms*.

By means of the concept of *contextual alarms*, the evaluation process is as below. First of all, the target network will communicate with the source network (i.e., using QQ, MSN and browsing the internet). The Wireshark will monitor and record all the network packets. Then, we used the packet generator to simulate some malicious packets by modifying the contents of three packet types: ICMP, TCP and UDP according to Snort rule database such as *icmp.rules*, *telnet.rules* and *scan.rules*.

Through sending out malicious packets, Snort can generate a number of alarms (including both real alarms and non-critical alarms) by examining the network packets. All the generated alarms will be forwarded into our alarm filter and converted to the *contextual alarms* by adding the information of target applications and OS. In the analysis engine, all *contexture alarms* will be compared with relevant contextual information followed by the evaluation steps (see Section III). The outputs of our alarm filter are critical alarms.

B. Evaluation Results

By understanding the experimental methodology, we give several examples of the *contextual alarms* in Table I. There are totally 8 items as follows: *packet type*, *source IP address*, *destination IP address*, *source port number*, *destination port number*, *alarm description*, *application information (APP)* and *OS information*.

In Table I, the first *contextual alarm* is related to TCP packets that come from the source IP "197.218.177.1" to the destination IP "172.16.114.15", the source port number is 20 and the destination port number is 80. The application information of this alarm is "IE application" and the target OS is "Windows". The description of this alarm is "ATTACK-RESPONSES 403 Forbidden". The other two contextual alarms are similar to the first one.

Table I
SEVERAL EXAMPLES OF CONTEXTUAL ALARMS.

Packet type	TCP	TCP	ICMP
Sour. IP	197.218.177.1	197.218.176.1	197.218.177.15
Dest. IP	172.16.114.15	172.16.114.15	172.16.112.2
Sour. Port	20	80	–
Dest. Port	80	4000	–
Description	ATTACK-RESPONSES 403 Forbidden	ATTACK-RESPONSES 403 Forbidden	ICMP Echo Reply
APP	IE	QQ	–
OS	Windows	Linux	Windows

In addition, Table II gives some examples of the contextual information in the analysis engine.

Table II
SEVERAL EXAMPLES OF CONTEXTUAL INFORMATION IN THE ANALYSIS ENGINE.

Sour. IP	197.218.177.1	197.218.176.1	197.218.177.1
Dest. IP	172.16.114.15	172.16.114.24	172.16.112.2
Packet Type	TCP	TCP	ICMP
Dest. Port	20	80	21
Description	ATTACK-RESPONSES 403 Forbidden	ATTACK-RESPONSES 403 Forbidden	ICMP Echo Reply
APP	IE	QQ	–
OS	Windows	Linux	Windows

The contextual information in the analysis engine contains 7 items: *source IP address*, *destination port number*, *destination IP address*, *packet type*, *alarm description*, *application information (APP)* and *OS information*.

Following by the experimental methodology, we conduct the experiment and show the initial experimental results in Fig. 4.

Analysis: As shown in Fig. 4, our alarm filter greatly reduces the number of Snort alarms in our deployed network environment. For instance, in the 10th hour of our experiment, the number of Snort alarms is decreased by 48.6% after adding the contextual information. The filtration accuracy of non-critical alarms is nearly 95% in terms of

 ISBN: 978-1-61208-201-1

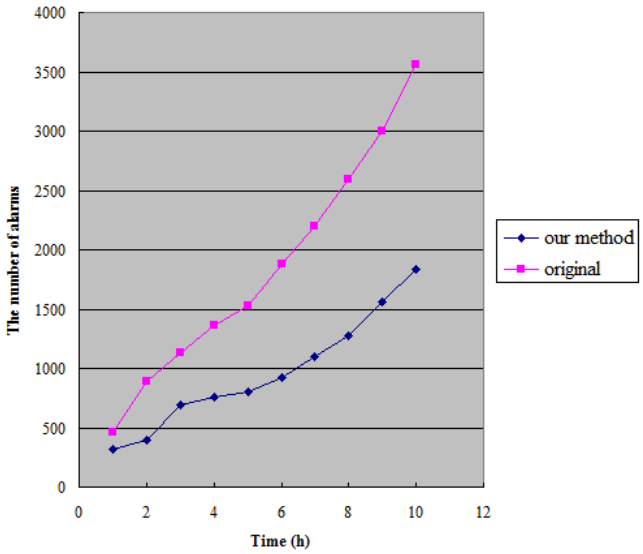


Figure 4. The alarm filtration results with regard to our context-based non-critical alarm filter in the experiment.

the packet records logged by the Wireshark. The reason for some missed non-critical alarms is that there is no relevant contextual information stored in the analysis engine or the contextual information is not found in some *contextual alarms* (i.e., the third contextual alarm in Table I, the target APP is not recognized).

In this experiment, the false filtration rate (FFR)[2] of our alarm filter is 0 in that we only filter out the alarms which are regarded as non-critical alarms according to APP and OS information. While the negative filtration rate (NFR)[3] of our alarm filter is about 10% in this experiment since we regard input alarms as critical alarms by default if the contextual information is not stored. On the whole, the initial experimental results show that our context-based non-critical alarm filter is encouraging in the network environment.

V. Limitations and Future Work

This is an early work on constructing non-critical alarm filter. Based on our initial experimental results, there are some issues that we can improve in the future work.

- *Anomaly-based detection system*. In our current work, we only investigate the performance of our scheme on Snort alarms (which are generated from a signature-based IDS). For the anomaly-based detection system, we leave it as an open problem for our future work to investigate the performance of our scheme on the alarms from the anomaly-based detection systems.
- *Contextual information*. In this paper, our work uses OS operating system and application types as the contextual information. However, we acknowledge that it is hard to identify the information accurately in some cases. We consider it as an open problem for our future work, to explore other types of contextual information.

Therefore, our future work could include considering and combining more applicable contextual information into our alarm filter to help better filter out the non-critical alarms, or constructing a more powerful alarm type to facilitate our alarm comparisons. In addition, the future work could also include comparing our scheme with other research work in this domain or deploying our alarm filter in a real network environment to further explore its performance.

[2]FFR: A critical alarm is regarded as a non-critical alarm.
[3]NFR: A non-critical alarm is regarded as a critical alarm.

VI. Conclusion

The large number of non-critical alarms is a big problem with regard to IDSs. In this paper, we propose a context-based non-critical alarm filter to help filter out these non-critical alarms. In particular, our proposed alarm filter consists of three main components: an indexing component, an analysis engine and a monitor component. The indexing component uses the source IP address and the destination port number as the index values to link the input alarms to corresponding contextual information. Then, the analysis engine compares the contextual information with the contextual alarms which are converted from the original alarms. The monitor component is used to update the index values in the indexing component. In the experiment, we explore the performance of the alarm filter under a constructed network environment. The initial experimental results show that our alarm filter is encouraging and effective to reduce the non-critical alarms in our network deployment and lighten the analysis burden for security analysts.

References

[1] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-Time," *Computer Networks* 31(23-24), pp. 2435-2463, 1999.
[2] K. Scarfone and P. Mell, Guide to Intrusion Detection and Prevention Systems (IDPS), NIST Special Publication 800-94, Feb 2007.
[3] R. Sommer and V. Paxson, "Outside the closed world: On using machine learning for network intrusion detection," *In IEEE Symposium on Security and Privacy*, pp. 305-316, 2010.
[4] T.H. Ptacek and T.N. Newsham, "Insertion, evation, and denial of service: Eluding network intrusion detection," *Technical Report*, Secure Networks, January 1998.
[5] S. Axelsson, "The base-rate fallacy and the difficulty of intrusion detection," *ACM Transactions on Information and System Security*, pp. 186-205, August 2000.
[6] Snort - The Open Source Network Intrusion Detection System. http://www.snort.org/. (Accessed on November 2011)
[7] Symantec Corp., Internet Security Threat Report, Vol. 16. http://www.symantec.com/business/threatreport/index.jsp
[8] J. McHugh, "Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory," *ACM Transactions on Information System Security*, pp. 262-294, 2000.

 ISBN: 978-1-61208-201-1

[9] R.P. Lippmann, D.J. Fried, I. Graf, J.W. Haines, K.R. Kendall, D. Mcclung, D. Weber, S.E. Webster, D. Wyschogrod, R.K. Cunningham, and M.A. Zissman, "Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation," *In DARPA Information Survivability Conference and Exposition*, pp. 12-16, 2000.

[10] F. Gagnon, F. Massicotte, and B. Esfandiari, "Using Contextual Information for IDS Alarm Classification," *In Proceedings of the 6th Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA'09)*, pp. 147-156, 2009.

[11] R. Lippmann, S. Webster, and D. Stetson, "The Effect of Identifying Vulnerabilities and Patching Software on the Utility of Network Intrusion Detection," *In Proceedings of Recent Advances in Intrusion Detection*, pp. 307-326, 2002.

[12] M. Roesch, "Snort-lightweight intrusion detection for networks," *In Large Installation System Administration Conference*, pp. 229-238, 1999.

[13] M. Cost, J. Crowcroft, M. Castro, A. Rowstron, L. Zhou, L. Zhang, and P. Barham, "Vigilante: End-to-end containment of Internet worms," *In Proceedings of 20th ACM Symposium on Operating System Principles (SOSP)*, pp. 133-147, 2005.

[14] R. Sommer and V. Paxson, "Enhancing Byte-Level Network Intrusion Detection Signatures with Context," *In Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*, pp. 262-271, 2003.

[15] D. Brumley, J. Newsome, D. Song, H. Wang, and S. Jha, "Towards automatic generation of vulnerability based signatures," *In IEEE Symposium on Security and Privacy*, pp. 2-16, May 2006.

[16] H. Debar and A. Wespi, "Aggregation and correlation of intrusion-detection alerts," *In Proceedings of Recent Advances in Intrusion Detection (RAID)*, pp. 85-103, October 2001.

[17] F. Cuppens and A. Miege, "Alert correlation in a cooperative intrusion detection framework," *In Proceedings of IEEE Symposium on Security and Privacy*, pp. 202-215, May 2002.

[18] K.H. Law and L.F. Kwok, "IDS False Alarm Filtering Using KNN Classifier," *In Proceedings of the International Workshop on Information Security Applications*, pp. 114-121, 2005.

[19] P. Ning and D. Xu, "Learning Attack Strategies from Intrusion Alert," *In Proceedings of the ACM Conference on Computer and Communications Security*, pp. 200-209, October 2003.

[20] J. Zhou, A.J. Carlson, and M. Bishop, "Verify Results of Network Intrusion Alerts Using Light-weight Protocol Analysis," *In Annual Computer Security Applications Conference*, pp. 117-126, December 2005.

[21] C. Kruegel and W. Robertson, "Alert verification: Determining the success of intrusion attempts," *In Proceedings of the Workshop on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA)*, pp. 25-38, July 2004.

[22] Nessus Vulnerability Scanner, http://www.nessus.org/

[23] A. Alharbt and H. Imai, "IDS False Alarm Reduction Using Continuous and Discontinuous Patterns," *In Proceedings of International conference on Applied Cryptography and Network Security*, pp. 192-205, 2005.

[24] National Vulnerability Database (NVD), http://nvd.nist.gov

[25] Colasoft Packet Builder: Packet Generator. (Accessed on July 2011) http://www.colasoft.com/packet_builder/

[26] M.A. Davenport, R.G. Baraniuk, and C.D. Scott, "Controlling false alarms with support vector machines," *In Proceedings of the International Conference on Acoustics, Speech and Signal (ICASSP)*, pp. 589-592, May 2006.

[27] Common Vulnerabilities and Exposures (CVE), (Accessed on November 2011) http:// cve.mitre.org

[28] Open Source Vulnerability Database (OSVDB), (Accessed on November 2011) http:// osvdb.org/

[29] Security Focus, http://www.securityfocus.com/. (Accessed on November 2011)

[30] Wireshark, http://www.wireshark.org. (Accessed on November 2011)

[31] T. Pietraszek, "Using adaptive alert classification to reduce false positives in intrusion detection," *In Proceedings of Recent Advances in Intrusion Detection (RAID)*, pp. 102-124, 2004.

[32] C. Mu, H. Huang, and S. Tian, "Intrusion Detection Alert Verification Based on Multi-level Fuzzy Comprehensive Evaluation," *In Proceedings of International Conference on Computational Intelligence and Security (CIS)*, pp. 9-16, Lecture Notes in Computer Science, 2005.

 ISBN: 978-1-61208-201-1

Improving Attack Aggregation Methods Using Distributed Hash Tables

Zoltán Czirkos, Márta Rencz, Gábor Hosszú
Department of Electron Devices
Budapest University of Technology and Economics
Budapest
{czirkos,rencz,hosszu}@eet.bme.hu

***Abstract*—Collaborative intrusion detection has several difficult subtasks to handle. Large amount of data generated by intrusion detection probes has to be handled to spot intrusions. Also, when correlating the pieces of evidence, the connection between them has to be revealed as well, as it may be the case that they are part of a complex, large-scale attack. In this article, we present a peer-to-peer network based intrusion detection system, which is able to handle the intrusion detection data efficiently while maintaining the accuracy of centralized approaches of correlation. The system is built on a distributed hash table, for which keys are assigned to each piece of intrusion data in a preprocessing step. This method allows one to make well-known correlation mechanisms work in a distributed environment.**

Keywords-collaborative intrusion detection; attack correlation; peer-to-peer; distributed hash table.

I. INTRODUCTION

In the earliest days of the Internet, services on the network were all based on trust. As e-commerce emerged, network hosts became victim of a wide range of everyday attacks. Due to the high amount of confidental data and resources that can be exploited, the possibilites and open nature of the Internet opened serious security questions as well.

The attacks network administrators fight against are both human and software controlled. They get more and more sophisticated, originating or targetting ocassionally multiple hosts at the same time. A large number of nodes can be simultaneously scanned by attackers to find vulnerabilities. Automatized worm programs replicate themselves to spread malicious code to thousands of vulnerable systems, typically of home users. Others compromise hosts to build botnets, which can deliver millions of spam e-mails per day.

As the manifestation of attacks, e.g., the evidence that can be observed is spread across multiple hosts, these large-scale attacks are generally hard to detect accurately. To recognize such, one has to first collect or *aggregate* the evidence, then *correlate* the pieces of information collected. A collaborative intrusion detection system has to analyze the evidence from multiple detector *probes* located at different hosts, and even on different subnetworks. However, this poses several problems to solve:

- large quantities of possible evidence collected,
- including inadequate data for precise decision making,
- communication and reliability problems,

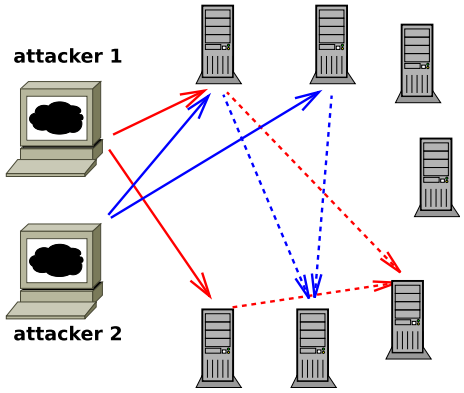


Figure 1. Messages carrying attack information in the Komondor system. If any probes in the network detect a suspicious event, it sends a report to the DHT. The nodes of the DHT act as correlation units as well, and are able to collect these reports.

- frequent change of intrusion types and scenarios.

Some of these troubles are specific for the isolated, host-based detection systems, while others occur only in case of the network scale intrusion detection. Despite of all these difficulties it is still worth collecting and correlating evidence available at different locations for the efficiency and accuracy boost of both detection and protection.

In this paper, we present a collaborative intrusion detection system, which organizes its participants to a *peer-to-peer (P2P) overlay network,* as seen in Figure 1. For intrusion data aggregation, a *distributed hash table* (DHT) is used, which is built on the Kademlia topology. This is used to balance the load of both aggregation and correlation of events amongst the participants. The organization of nodes in the overlay network is automatic. Should some nodes quit or their network links fail, the system will reorganize itself.

The rest of this paper is organized as follows. In Section II, we first review existing research of collaborative intrusion detection systems. Then we present the architecture of our intrusion detection solution based on the Kademlia DHT overlay in Section III. The results of our intrusion detection method and statistics of detection are highlighted in Section IV. Research is concluded in Section V.

 ISBN: 978-1-61208-201-1

II. Related Work

Attackers use various ways for intrusion of computer network systems depending on their particular goals. These methods leave different tracks and evidences, called the *manifestation of attacks* [1]. To discuss the internals of a collaborative intrusion detection system, we use the following terms [2]:

- *Suspicious events* are primary events, that can be detected at probes. Not necessarily attacks by themselves, but can be part of a complex attack scenario.
- *Attacks* are real intrusion attempts, which are used to gain access to a host or disturb its correct functioning. Usually these are made up from several suspicious events at once.

The activity of an SSH (Secure Shell, a remote login software) worm program can be seen as an example of an attack. These worms use brute-force login attempts using well-known user names and simple passwords [3], directed against a single host. The attempts are events that make up the attack in this case. Multiple failed login attempts usually indicate an attack, while a single failed attempt is usually only a user mistyping his password.

A. Centralized Collaborative Intrusion Detection

To achieve *collecting* and *correlating* events detected by a number of detector probes, various collaborative intrusion detection systems (CIDS) have been proposed, for which a detailed overview can be found in [4].

The earliest collaborative detection systems used a centralized approach for *collecting* the events. The Internet Storm Center *DShield* project collects firewall and intrusion detection logs from participants, uploaded either manually or automatically [5]. The log files are then analyzed centrally to create trend reports.

The *NSTAT* system [6] on the other hand is more advanced in the sense that its operation completely real-time. In NSTAT, the detection data generated by the probes is preprocessed and filtered before being sent to a central server for correlation. This system analyzes the order of events using a state transition mechanism with predefined scenarios to find out the connection between them.

The advantage of centralized methods is that the server is able to receive and process all data that could be gathered. Processing, i.e., correlation can be carried out with several different methods. SPICE [7] and CIDS [8] group events by their common attributes. The LAMBDA system tries to fit events detected into pre-defined and known scenarios [9]. The JIGSAW system maps prerequisites and consequences of events to find out their purposes [10].

B. Hierarchical and P2P Collaborative Intrusion Detection

By using hierarchical approaches, the scalability problem of centralized intrusion detection systems can be handled. The *DOMINO* system is used to detect virus and worm activity. It is built on an unstructured P2P network with participants grouped into three levels of hierarchy [11]. The nodes on the lowest level generate statistics hourly or daily, therefore they induce only a small network traffic.

The *PROMIS* protection system (and its precedessor, Netbiotic) uses the JXTA framework to build a partly centralized overlay network to share intrusion evidence [12]. The nodes of this system generate information for other participants about the frequency of detected suspicious events. This information is used to fine-tune the security settings of the operating system and the web browser of the nodes. This creates some level of protection against worms, but also decreases the usability of the system.

The *Indra* system is built on the assumption that attackers will try to compromise several hosts by exploiting the same software vulnerability [13]. If any attempts are detected by any participant of the Indra network, it alerts others of the possible danger. Participants can therefore enhance their protection against recognized attackers, rather than developing some form of general protection.

The scalability and single point of failure problems of centralized solutions can also be solved by using structured P2P application level networks. The P2P communication model enables one to reduce network load compared to the hierarchical networks presented above. The CIDS system [8] is a publish-subscribe application of the Chord overlay network. Nodes of this system store IP addresses of suspected attackers in a blacklist, and they subscribe in the network for notifications that are connected to these IPs. If the number of subscribers to a given IP address reaches a predefined threshold, they are alerted of the possible danger. The Chord network ensures that the messages generated in this application will be evenly distributed among the participants [14].

C. Structured P2P Networks

Structured P2P networks generally implement *distributed hash tables* [15]. DHTs store $\langle key; value \rangle$ pairs and allow the quick and reliable retrieval of any $value$ if the key associated to that is known precisely. This is achieved by using a *hash function* and mapping all data to be stored to the nodes selected by the distance of the hashed keys and their NodeIDs, which are chosen from the same address space. The connections between nodes are determined by their NodeID selected upon joining the network. They are selected so that the number of steps between any two node is usually in the order of $\log N$, where N is the count of all nodes.

DHTs all implement routing between their nodes in the application level to build the topology desired. The *Kademlia* network uses a binary tree topology [16], in which the distance is calculated using the XOR function. All nodes have some degree knowledge of the successively smaller subtrees of the network they are *not* part of. For any of these

 ISBN: 978-1-61208-201-1

subtrees they have routing tables called k-buckets, which store IP addresses of nodes that reside in distant subtrees. When a node looks up a selected destination, it successively queries other nodes, which are step by step closer to the destination. The queried nodes answer by sending their k-buckets to the source. As nodes closer to the destination have greater knowledge of their neighbors, the lookup will get closer every step, as discussed in [16]. The distance in the XOR metric is halved with every message, so the number of messages is $\log_2 N$ with N being the number of nodes in the tree.

III. The Komondor System Architecture

In this section, our intrusion detection system named *Komondor* is presented. Its most important novelty is that it uses the Kademlia DHT to store intrusion data and to disseminate information about detected events. Having analyzed the collected events, Komondor correlation units may start an alert procedure notifying other nodes of the possible danger if necessary.

A. Distributing Load Among Multiple Correlation Units

The Komondor peer-to-peer application level network consists of multiple nodes. All nodes have the *responsibility* of collecting and correlating intrusion data. They also report attacks discovered to other nodes of the network, as seen in Figure 1. All participants of the Komondor network serve as intrusion detection units and correlation units as well.

The Komondor network is designed to enable the previously mentioned correlation methods to be used in a distributed manner:

- Pieces, which are correlated should be sent to the same correlation unit, so that it can gather all the information about the attack.
- Pieces of evidence, which are part of distinct ongoing attacks should preferably be sent to different correlation units. This reduces load and improves overall reliability of the system.

Komondor achieves this goal by *assigning keys to preprocessed intrusion data*, as seen in Figure 2. Keys assigned are used as storage keys in the DHT as well. For different attackers or attack scenarios, different keys are selected, and this way data is aggregated at different nodes of the Komondor overlay.

Correct key selection is critical, since pieces of evidence, which might be correlated to each other must be assigned the same key and sent to the same Komondor node for correlation. Note that these pieces do not have to be detected by the same probe, yet they can be aggregated by the same correlation unit. The Komondor system is essentially a *middle layer inserted into the intrusion detection data path.* The nodes of the DHT are the correlation units, which have to implement the same correlation methods as their centralized counterparts. However, the correlation procedure is started as soon as the preprocessing stage with the key selection, and it is finalized at the correlation units.

The detected and preprocessed data of suspicious events is stored in the Komondor overlay. In this system, the key assigned at the preprocessing stage of detection is used as a *key for DHT operations* as well. The value parts of the $\langle key; value \rangle$ pairs stored are any other data, which might be useful for detection or protection. As all nodes use the same key selection mechanism and the same hash function, events related to each other will be stored at the same node, as seen in Figure 1. This way the algorithm ensures that the aggregator node has perfect knowledge of all events related to the attack in question.

The reason why a structured overlay – Kademlia – was selected for the Komondor system is that it has the advantages of distributed and centralized detection systems as well. Event data collected has to be sent to a single collector node only (this would not be possible with an unstructured overlay, as those have no global rule to map a key to a node.) Moreover, when Komondor nodes are under multiple but unrelated attacks, the network and computational load of both aggregation and correlation is distributed among nodes. The Komondor system neither has a single point of failure: the responsibility of correlating particular events is transferred to another node in this case. The overlay can also be used to disseminate other type of information as well, for example the attack alerts, which enable nodes to create protection.

B. Kademlia as the DHT Topology of Komondor

The nodes of Komondor create a *Kademlia DHT* overlay. This is the topology, which can adapt its routing tables to the dynamic properties of traffic generated by the intrustion detection probes. As discussed below, other DHTs wouldn't be able to adapt their routing tables to the dynamic properties of this kind of traffic.

Storing information of events generated by the probes generates significant overlay traffic, which will load not only detector and collector nodes, but other nodes along the path from the former to the latter one as well, as routing between nodes is handled on the application level. If the *events are in correlation* with the same attack, the *key chosen is likely to be the same*, making the distribution of keys highly uneven. However, by using Kademlia, network traffic can be significantly reduced in this scenario. The reason for this is that the routing algorithm of Kademlia is very flexible: any node can be put to the routing tables of any other node while still obeying the rules of the routing protocol. Routing tables of other DHT overlays like CAN or Chord are much more rigid, and therefore the routing algorithm of those cannot optimize the number of messages for the store requests with the same key.

Table I compares the number of messages generated in intrusion detection for Kademlia and Chord, with the latter

 ISBN: 978-1-61208-201-1

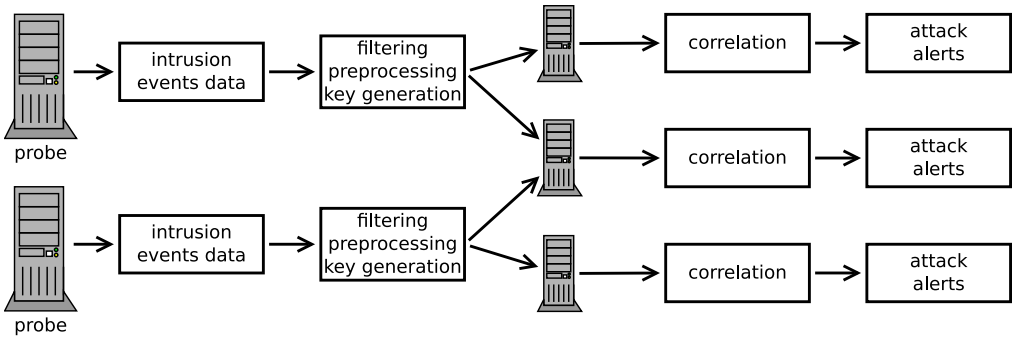


Figure 2. Distributed collection and distributed correlation of intrusion evidence from various probes. The Komondor system assigns keys to pieces of evidence so that data can be stored efficiently in a DHT. By using these keys, computational load of correlating can be distributed among several units.

Overlay	**Chord**	**Kademlia**
Routing algorithm	recursive	iterative
Node lookup	0	$\log_2 N$
First event stored	$\log_2 N$	$1 + \log_2 N$
n **events with the same key**	$n \cdot \log_2 N$	$n + \log_2 N$
Average number of messages per event	$(n \cdot \log_2 N)/n$	$(n + \log_2 N)/n$
Average number of messages with $n \to \infty$	$\log_2 N$	1

Table I
NUMBER OF MESSAGES IN STRUCTURED OVERLAYS FOR INTRUSION DETECTION

being an example for hacing rigid routing tables. Chord uses a *recursive routing mechanism,* which means that messages are forwarded by overlay nodes along the path from the source to the destination of the message. If Komondor would be built on Chord, the number of messages generated in the overlay would be in the order of $\log_2 N$ for each detected event, where N is the node count of the overlay.

Kademlia uses an iterative algorithm. To store a $\langle key; value \rangle$ pair, a Kademlia node first looks up the IP address of the destination node by successively querying nodes closer to the destination. After finding out its address, data is sent directly from the source and the destination. This also implies that the payload of the message is contained in every message for Chord, and only in the last message for Kademlia. For Kademlia, the node has to first look up the address of the destination, which also takes $\log_2 N$ messages. Having done that, it requires one more message $(+1)$ to send the payload as well. If multiple events are to be stored, which are detected by the same probe (this is a likely scenario for a node that is under attack), the *lookup procedure can be optimized away*, as the key and therefore the collector node is the same, too. For sending data of n events, the number of messages generated is only $n+\log_2 N$ for Kademlia and $n \cdot \log_2 N$ for Chord, which is worse at the factor of n for the latter one. The limit of messages per event drops to 1 for Kademlia in this common intrusion detection scenario.

IV. RESULTS AND DISCUSSION

In this section, we present statistics of intrusion attempts detected using the implemented Komondor system. The statistics are evaluated to show which types of attacks this system can be used to detect.

The present Komondor implementation used the open-source *Snort intrusion detection system* [17] to detect intrusion events. However, it could collaborate with other intrusion detection solutions as well. The key selected for each event was the *IP address* of the attacker, as found in the Snort log file. It was also used for correlation. We selected common event types from the Snort database and also tagged events with a severity score. Intrusion alert was triggered when the sum of these scores reached a threshold level. This simple correlation method enabled us to determine the efficiency and reliability of the Komondor system for known attack types presented here. The number of probes in the system varied from 7 to 10, each with their own IP address but on the same subnetwork. Data presented here was collected in a three year interval.

A. Attack Intervals and Number of Events

Figure 3 shows invalid passwords detected for SSH login attempts on various hosts [3]. Every dot on the graph is an individual attack. The y axis shows the number of events or the number of invalid passwords detected. The duration of an attack is the time interval between the first and the last event detected, and is on the x axis. Several attackers were detected by multiple Komondor probes, because the SSH worm that was trying to gain access to the subnetwork tried to login all on-line hosts it found. The number of probes which detected an attack in question is shown by the color of the dots. (In the case of multiple probes detecting an attacker, the event number on axis x is an average per probe.)

 ISBN: 978-1-61208-201-1

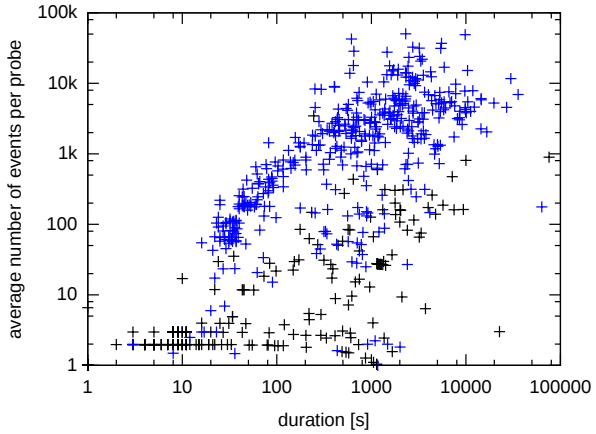


Figure 3. Number of invalid password events detected for various attacks (y axis) plotted by the duration of the attack (x axis). The color of the dots represent the number of probes a specific attacker was detected by.

Type of attack	Attacks	Protection	Ratio
phpMyAdmin scan	107	71	66%
MSSQL overflow	4355	15	0%
SSH connection lost	490	321	65%
SSH failed password	546	219	40%
SSH invalid user	51	47	92%
FTP failed login	46	2	4%

Table II
NUMBER OF ALL ATTACKS AND ATTACKS FOR WHICH PROTECTION COULD BE BUILT BY KOMONDOR, FOR EACH ATTACK TYPES.

Attacks, which were detected by one probe only (black dots) have much less events associated to them. The 1 100 attacks shown on the graph have as much as 450 of them stacked up in the $(1; 1)$ point. These evidently came from human interaction. Attacks detected by multiple probes usually suggest automatic worm programs using dictionary attacks against the detector hosts.

This experience suggests that distributed intrusion detection can benefit from the advantages of DHTs:

- Attackers could be detected by several probes at the same time. When multiple hosts are attacked, recognizing an attacker using any evidence from any probe of the Komondor network, several hosts could be protected using firewalls at the same time, which might promptly be attacked, too.
- Attack evidence came from multiple probes. One attack is likely to be associated to thousands or tens of thousands of events, which must be stored and processed in the overlay. This type of load can be dealt with the DHT fairly well, as it can select different collector nodes for each individual attack and therefore balance the load.
- When detecting an event, which generates the same key, the Kademlia DHT can significantly reduce network traffic, as the IP address of the collector nodes have to be looked up only once. When the IP address is obtained, the system works as if it were using a centralized approach with the same benefits as those.

B. Attack Types Detected by Komondor

Table II shows various attack types and the efficiency for the Komondor system regarding protection. The *protection* column shows the number of attacks for each type, for which the attack continued after it was blocked on the firewall, and the activity of the attacker was detected by another Komondor node of the same subnetwork. For these attacks, the collaborative intrusion detection can greatly enhance the protection of hosts.

Figure 4 shows event numbers and attack durations for different worms attacking SQL servers. The y axis has two scales for each graph. The scales of the left hand side show attack durations (red plot), and the right hand side scale shows the number of events (blue plot). Attacks are sorted by duration. Every value on the x axis is an attack for which the duration and the number of events is shown right under each other.

A worm, which scanned the Web servers for vulnerabilities via HTTP requests is shown on the right hand side subfigure. For any event detected, the IP address of the attacker can be recognized by the correlation units. The left hand side graph presents the properties of the Slammer worm, which penetrates outdated MSSQL servers. This worm does not issue more attempts in a short time interval to the same host, and selects IP addresses of victims randomly. For detecting this type of attacks, the PROMIS and CIDS systems could be used more effectively.

V. CONCLUSION

Attacks on the Internet mean constantly growing problem for network administrators. Sophisticated attacks have evidence spread across multiple hosts and subnetworks. To detect these attacks promptly and correctly, data must be aggregated and analyzed automatically. In this article, the novel Komondor intrusion detection system is presented, which enables current attack correlation methods to be upgraded to work in a distributed environment, thereby making them feasible for large-scale deployment. This is achieved by inserting a middle layer into the intrusion detection data path, which utilizes the Kademlia overlay.

The novelty of the method presented is attaching a key to the detected events, which key is then used to send the events for correlating to several correlation units that are organized as a DHT. This mechanism can be used to reduce network and computational load and increase reliability of the system, while still retaining the advantages of centralized approaches of intrusion detection. By mapping the detected

 ISBN: 978-1-61208-201-1

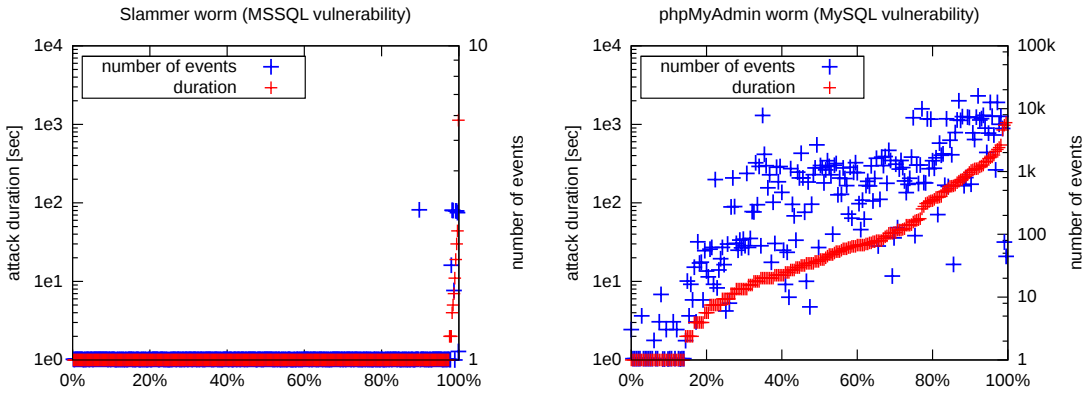


Figure 4. Attack intervals and number of events for different worm activities detected by the Komondor system. The left hand side shows a worm, which scanned our Web servers via HTTP in order to find a phpMyAdmin installation to gain access to MySQL databases. On the right hand side the activity of the infamous Slammer worm is shown, which penetrates MSSQL servers.

events to nodes in the system, all nodes are assigned the same level of responsibility as well. Our further research will focus on considering the different computational and network capacity of nodes to prevent those with slow connections or CPUs from being overloaded by intrusion detection data.

ACKNOWLEDGEMENT

The work reported in the paper has been developed in the framework of the project "Talent care and cultivation in the scientific workshops of BME". This project is supported by the grant TÁMOP - 4.2.2.B-10/1–2010-0009.

REFERENCES

[1] D. Mutz, G. Vigna, and R. Kemmerer, "An Experience Developing an IDS Stimulator for the Black-Box Testing of Network Intrusion Detection Systems," in *In Annual Computer Security Applications Conference, Las Vegas, NV*, 2003, pp. 374–383.

[2] H. Debar, "Intrusion Detection Systems-Introduction to Intrusion Detection and Analysis," *Security and privacy in advanced networking technologies*, p. 161, 2004.

[3] C. Seifert, "Analyzing Malicious SSH Login Attempts," http://www.symantec.com/connect/articles/analyzing-malicious-ssh-login-attempts, Nov. 2010, retrieved: March, 2012.

[4] C. Zhou, C. Leckie, and S. Karunasekera, "A Survey of Coordinated Attacks and Collaborative Intrusion Detection," *Computers & Security*, vol. 29, no. 1, pp. 124–140, 2010.

[5] "Internet Storm Center," http://www.dshield.org/, retrieved: March, 2012.

[6] R. Kemmerer, "NSTAT: A Model-based Real-time Network Intrusion Detection System," *University of California-Santa Barbara Technical Report TRCS97*, vol. 18, 1997.

[7] A. Valdes and K. Skinner, "Probabilistic Alert Correlation," *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection*, pp. 54–68, October 2001.

[8] C. V. Zhou, S. Karunasekera, and C. Leckie, "A Peer-to-Peer Collaborative Intrusion Detection System," in *Networks, 2005. 13th IEEE International Conference on*, vol. 1.

[9] F. Cuppens and R. Ortalo, "LAMBDA: A language to model a database for detection of attacks," in *Recent advances in intrusion detection*. Springer, 2000, pp. 197–216.

[10] S. Templeton and K. Levitt, "A Requires/provides Model for Computer Attacks," in *Proceedings of the 2000 workshop on New security paradigms*. ACM, 2001, pp. 31–38.

[11] V. Yegneswaran, P. Barford, and S. Jha, "Global Intrusion Detection in the DOMINO Overlay System," in *Proceedings of NDSS*, vol. 2004, 2004.

[12] V. Vlachos and D. Spinellis, "A PRoactive Malware Identification System based on the Computer Hygiene Principles," *Information Management and Computer Security*, vol. 15(4), pp. 295–312, 2007.

[13] R. Janakiraman, M. Waldvogel, and Q. Zhang, "Indra: A Peer-to-peer Approach to Network Intrusion Detection and Prevention," in *Enabling Technologies: Infrastructure for Collaborative Enterprises. WET ICE 2003.* IEEE, 2003, pp. 226–231.

[14] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications," *ACM SIGCOMM Computer Communication Review*, vol. 31, no. 4, pp. 149–160, 2001.

[15] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-peer Content Distribution Technologies," *ACM Computing Surveys (CSUR)*, vol. 36, no. 4, pp. 335–371, 2004.

[16] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," *Peer-to-Peer Systems*, pp. 53–65, 2002.

[17] "Snort – Open-source Intrusion Detection System," http://www.snort.org/, retrieved: March, 2012.

 ISBN: 978-1-61208-201-1

Engineering Security Protocols with Modelchecking – Radius-SHA256 and Secured Simple Protocol

Florian Kammüller, Glenford Mapp, Sandip Patel, and Abubaker Sadiq Sani
Middlesex University
Computer Communications Group
f.kammueller@mdx.ac.uk, mapp@mdx.a.cuk, sp1264@live.mdx.ac.uk, ss1234@live.mdx.ac.uk

***Abstract*—This paper presents Radius-SHA256, an adaptation of the Radius protocol for remote authentication for network access to the secure hash function SHA-256 and a Secure Simple Protocol. Both protocols have been formalized in the Avispa model checker, an automated verification tool for security of protocols. The work on Radius utilizes the existing formalization of the standard Radius protocol thereby establishing general validity and transferability of the established security proof and showing how refactoring can be applied in security protocol engineering. The development of a secured version of the SP protocol shows how gradually adding cryptographic keys to a transport protocol can introduce verified security while maintaining a level of trust in the adapted protocol.**

Keywords-Security protocols; Model Checking; Cryptographic Hashes; Simple Protocol.

I. INTRODUCTION

Radius, a remote authentication protocol used for building up secure communications of clients with networks via network access servers, uses the message digest function MD5, a hash function which has meanwhile been proven to have security weaknesses. By contrast, the hash function SHA-256 still remains unchallenged. Although seemingly straightforward and thus tempting, simply replacing MD5 by SHA-256 in the Radius protocol must be considered potentially harmful since authentication protocols are extremely sensitive to minor changes as the history of attacks shows. In December 2008, an attack on the SSL protocol has been demonstrated based on the previously discovered collisions of the MD5 hash function [10]. The engineers of that attack recommend the discontinuation of use of SSL based on MD5. Fortunately, for SSL the use of the hash function is already by design a choice point. For Radius, this flexibility is not yet established; this is the subject and result of this paper. Triggered by the alarming history of attacks of security protocols, formal verification techniques have long been deemed to be a way out.

Model checking, a push-button technology for mathematical verification of finite state systems has been discovered to be a suitable tool for security analysis of authentication protocols [4]. Ever since, this technology has proved to be useful for the engineering of secure protocols, e.g., for adaptation of the Kerberos protocols to mobile scenarios [3].

We investigate whether Radius-SHA256 – our proposed adaptation of the Radius protocol – can provide better security guarantees than its original. To provide evidence based on mathematical rigor we use the Avispa model checker. Fortunately, we can rely on the rich data base of this tool providing a model of the original protocol. By adapting this model to our Radius-SHA256 and checking that the original security guarantees still hold, we prove two things (a) that Radius-SHA256 is secure and (b) that the security guarantees have general validity, i.e., they can be carried over to protocols Radius-X for hashes X. The latter result corresponds to a reduction of Radius security to the security of the underlying hash function.

The Simple Protocol (SP) [5] is a new protocol that is currently being developed by the Ycomm group [13]. As a second engineering exercise, we report on a *secured version of the SP protocol*. This exercise shows how a new development of a special purpose protocol can profit from a simultaneous modelling and analysis with a dedicated modelchecker like Avispa.

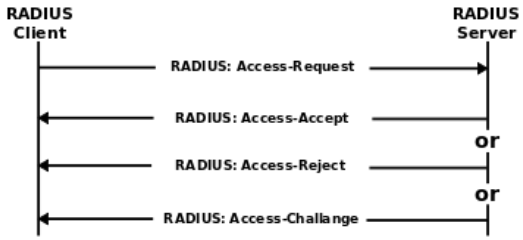
This paper is based on the Masters Theses of two of the authors [6], [8]. In this paper, we first provide the prerequisites of this project: brief introductions to the Radius protocol, the Simple Protocol, Avispa model checking, and hashes (Section II). From there, we develop our new version Radius-SHA256 by introducing its model in Avispa in detail (Section III) and illustrate how this model can be efficiently used to verify security goals (Section III-D). Next, we show how a protocol can be extended step by step introducing cryptographic keys to add authentication and secure it (Section IV). We finally offer conclusions and an outlook (Section V).

II. BACKGROUND

A. Radius

One of the major issues with networks is their security and one response to this challenge are authentication protocols. Radius is a popular protocol providing security to communication channels. Radius stands for *Remote Authentication Dial in User Service* and serves to secure communication between *Network Access Servers* (NAS) and so-called Radius

 ISBN: 978-1-61208-201-1



servers. Radius satisfies the AAA (Authentication, Authorization and Accounting) protocol standards in both local and roaming situations. In January 1997, Radius standards were first introduced in RFC 2058 and Radius accounting in RFC 2059. After that RFC 2138 and RFC 2139 were published and they made the previous RFC obsolete. They both were made obsolete in turn by RFC 2865 and RFC 2866 respectively.

Assume that there is an Internet service provider (ISP) and he has two NAS. A NAS allows a user to connect directly to the ISP's network and be accepted by a core router which directly connect with ISP's network backbone. When a user wants to access his services, he sends a request to the NAS which forwards the user request to the main server to check the supplied credentials. This process is called authentication.

After authentication, the NAS has to check the access list of the user and then decide which services are permitted to this user. The RADIUS server then replies to the NAS with Access Reject, Access Challenge, or Access Accept as illsutrated in above Figure [12]. This information is forwarded by the Radius server to the NAS. This is called authorization. Once a user is authenticated and authorized successfully, the NAS creates a connection between the user and the main server through which both can exchange their information. This secure connection is called a session. All the information regarding the session will be saved by the NAS for its accounting purposes. It includes start time of session, termination time of session, size of total received and sent data, amongst other information for accounting.

B. Simple Protocol

A new trend in next generation networks is the divergence between local area networks (LAN) and wide area networks (WAN) because there is still an increase of efficiency to be expected in LANs. Additionally, the ubiquity of computing devices and common usage of mobile devices asks for a flexibility that is better supported with fixed core networks and flexible wireless networks at the periphery. A reconsideration of the TCP/IP seems appropriate since adaptation of TCP to the often heterogeneous requirements of local wireless networks is not easy. The Simple Protocol (SP) [5] is intended to be used in combination with TCP but TCP for the WAN and SP for the LAN communication. SP is part of a wider development of the Y-Comm framework [13] – a new architecture for mobile heterogeneous networking.

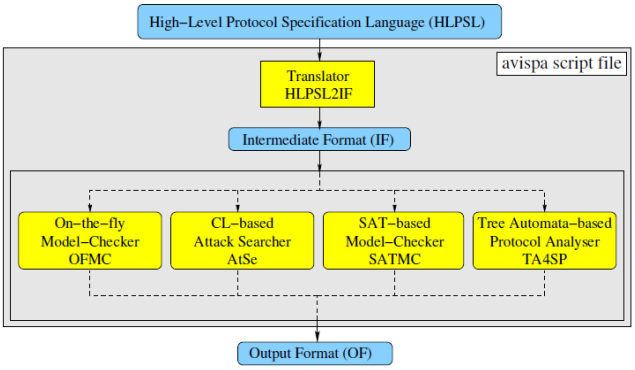


Figure 1. Avispa: language formats and tool architecture [1]

A specially LAN-centric transport protocol has different requirements from a WAN transport protocol, e.g., TCP, since performance issues differ. These requirements mark the design decision that define SP [5]. Since most LAN communications consist of messages or transactions, SP supports a message-based communication in contrast to TCP streams. The higher speed available in LAN is exploited by using a larger window size for SP than WAN protocols: SP supports 4MB message sizes by default and can even be increased. In order to keep packet processing simple, SP uses a small number of connection states as well as packet types. Flexibility is achieved by allowing Quality of Service (QoS) to be set using the packet types.

In this paper (Section IV), we summarize briefly how the Avispa support helped in designing a secure extension of SP by hybrid cryptography. Extending the initial connection part of the protocol, we add public-key based authentication while simultaneously exchanging symmetric session keys for the following secured data exchange of SP. The protocol achieves authenticity by public keys while preserving its efficiency to an extent through the use of faster symmetric key encryption.

C. Avispa

Avispa stands for Automated Validation of Internet Security-sensitive Protocols and Applications [1]. To model and analyze a protocol, Avispa provides its own High-Level Protocol Specification Language (HLSPL). In order to check security, Avispa translates the given HLSPL specification in the intermediate format IF, which is then the basis for four different verification machines that can be applied to model check security properties on a protocol expressed as depicted in Figure 1. Avispa uses Dolev-Yao channels annotating them as a type as `channel(dy)`. This means that the attacker is assumed to be able to do eavesdropping, intercepting and faking on these channels. Protocols can be very naturally specified in Avispa using the *role* concept. Every principal is modeled as such a role which enables encapsulating its communication parameters, local variables and constants. Based on that, a role describes state changes by defining transitions

 ISBN: 978-1-61208-201-1

between states that may depend on pre- and postconditions of the current state. Roles can furthermore be instantiated in other roles. This enables the composition of the single roles representing the single principals into a protocol session while synchronizing them on their communication. It also enables specifying an attacker. Once the protocol is thus specified predefined HLSPL propositions, most prominently secrecy and authentication can be automatically verified. We introduce more details on HLSPL constructs, their IF translation, and the verification features when applying them to formalize Radius-SHA256 in the following section.

D. Hash Functions

Hash functions – also known as message digests or compression functions – map arbitrary length inputs to fixed size outputs. They are considered as cryptographic hash functions if they provide the following three properties: (a) they cannot be inverted, i.e., given $y = H(x)$, the input x cannot be found, (b) it is impossible to find collisions, i.e., we cannot find x, y with $H(x) = H(y)$, and (c) given an input hash pair it is impossible to find another input with the same hash value, i.e., for $H(x) = y$ we cannot find x' such that $H(x') = y$. The latter two properties resemble each other expressing the idea of *collision resistance* but the second one is stronger.

These basic properties of good hashes give rise to use them for cryptography. However, since a hash is a deterministic function it has as such not the same quality as an encryption algorithm: anyone can apply the hash. However, a hash can be easily combined with a shared secret to provide authentication which is often used for so-called message authentication codes (MAC). For example, let Kcs be a shared secret. Then, $H(Kcs)$ can be used as an authentication token since only principals who have access to Kcs can produce this token.

III. RADIUS-SHA256

In this section, we present the protocol Radius-SHA256 as derived from the classical Radius of RFC2865/66 by replacing MD5 by SHA-256. At the abstract protocol level this replacement seems simple but in order to ensure that this change of the original protocol preserves the security properties, we start from the formal presentation of the original Radius protocol and develop the new Radius-SHA256 on that formal basis. This enforces a detailed investigation of the necessary adjustment to the old – no longer secure – version of Radius and in addition enables comparison to the previously established security guarantees showing whether they still hold. From an engineering perspective, this procedure corresponds to a kind of refactoring of a protocol specification: re-engineering the previous security specification enables re-invocation of the previous verification by rerunning security check routines.

```
role client(C,S: agent,
   Kcs: symmetric_key,
   SHA256: hash_func,
   Success, Failure: text,
   Access_accept,Access_reject: text,
   SND, RCV: channel(dy))
played_by C def=
local  State: nat,
   NAS_ID, NAS_Port: text,
   Chall_Message: text
const  kcs: protocol_id,
   sec_c_Kcs : protocol_id
init State := 0
transition
   t1. State  = 0 ∧ RCV(start) ⇒
       State':= 1 ∧ NAS_ID':=new()
       ∧ NAS_Port':=new()
       ∧ SND(NAS_ID'.NAS_Port'.SHA256(Kcs))
       ∧ secret(Kcs,sec_c_Kcs,C,S)
   t2. State  = 1 ∧ RCV(NAS_ID.Access_accept) ⇒
       State':= 2 ∧ SND(NAS_ID.Success)
   t3. State  = 1 ∧ RCV(NAS_ID.Access_reject) ⇒
       State':= 3 ∧ SND(NAS_ID.Failure)
   t4. State  = 1 ∧ RCV(NAS_ID.Chall_Message') ⇒
       State':= 4 ∧ SND(NAS_ID.Chall_Message'_Kcs)
                  ∧ witness(C,S,kcs,Kcs)
   t5. State  = 4 ∧ RCV(NAS_ID.Access_accept) ⇒
       State':= 5 ∧ SND(NAS_ID.Success)
end role
```

Figure 2. Client role of Radius-SHA256 in HLSPL

We introduce the protocol Radius-SHA256 by its formal model in HLSPL, the specification language of Avispa. Its level of abstraction is sufficient to comprehend just the major gist of the protocol. This model contains four roles: client, server, session, and environment. The idea is that the client role represents the NAS and the server role represents the Radius server. In applications, client and server might as well be represented by proxies depending on the type of network. For the formal presentation of the protocol, we simplify by summarizing the scenario as a client-server session. As a session, we consider the time period of a client-server communication. The attacker is modeled by the role of the environment that specifies the basis for the attacks on protocol executions.

Each of these components client, server, session and environment is modeled by a so-called "role" in HLSPL. Client (Section III-A) and server (Section III-B) define the two matching sides of the protocol; their composition as defined in the role session only gives the full protocol (see Section III-C and Figure 4), which can again be instantiated to model legal session and attacker.

A. Client-side Protocol

The client role is specified in Figure 2. This role definition defines the protocol by specifying the necessary entities, like identifiers, messages and used cryptographic primitives, e.g., the symmetric key `Kcs` in its header. Note, here how we define `SHA256` to be a *hash function* in this header by using

 ISBN: 978-1-61208-201-1

```
role server(C,S: agent,
            Kcs: symmetric_key,
            SHA256: hash_func,
            Success, Failure: text,
            Access_accept,Access_reject: text,
            SND, RCV: channel(dy))
played_by S def=
local State: nat,
      NAS_ID, NAS_Port : text,
      Chall_Message     : text
const kcs: protocol_id,
      sec_s_Kcs : protocol_id
init State := 11
transition
t1. State  = 11 ∧ RCV(NAS_ID'.NAS_Port'.SHA256(Kcs)) ⇒
    State':= 12 ∧ SND(NAS_ID'.Access_accept)
                ∧ secret(Kcs,sec_s_Kcs,C,S)
t2. State  = 12 ∧ RCV(NAS_ID.Success) ⇒
    State':= 13
t3. State  = 11 ∧ RCV(NAS_ID'.NAS_Port'.SHA256(Kcs)) ⇒
    State':= 14 ∧ SND(NAS_ID'.Access_reject)
t4. State = 14 ∧ RCV(NAS_ID.Failure) ⇒
    State':= 15
t5. State  = 11 ∧ RCV(NAS_ID'.NAS_Port'.SHA256(Kcs)) ⇒
    State':= 16 ∧ Chall_Message':=new()
                ∧ SND(NAS_ID'.Chall_Message')
t6. State  = 16 ∧ RCV(NAS_ID.Chall_Message_Kcs) ⇒
    State':= 17 ∧ SND(NAS_ID.Access_accept)
                  ∧ request(S,C,kcs,Kcs)
t7. State  = 17 ∧ RCV(NAS_ID.Success) ⇒
    State':= 18
end role
```

Figure 3. Server role of Radius-SHA256 in HLSPL

the Avispa keyword `hash_func`. This function is applied in the first transition of the following client-side of the protocol specification. In detail, the steps of the protocol are defined as *state transitions* that are conditional on logical conditions of a current state `State` $\in \{1, \ldots, 5\}$: each of the five rules in the `transition` section in Figure 2 defines a *precondition* for this current state (to the left of the implication arrow $\Rightarrow$) and a *postcondition* on the post state `State'` of a transition after the $\Rightarrow$. The conditions are conjoined by logical conjunction with $\wedge$. The initial state is `State` zero. For example, the first transition `t1` in Figure 2 can be read as follows. If the precondition holds, i.e., the current state is "state 0" and the role receives on its input channel `RCV` the message start, then the transition `t1` is enabled. If this transitions fires, the post-state is "state 1" and the message `NAS_ID.Success` is sent on the output channel `SND`. The following transitions can be read in the same manner. Since the client represents only one principal in this protocol, we need to need to define the server side of the protocol to complement it.

B. Server-side Protocol

Figure 3 now shows the definition of the second principal in the model of Radius-SHA256: the Radius-server. The transitions defined in the role server correspond to the transitions of the client. Each `SND` on one side corresponds to a `RCV` on the other side. However, in order to put these building blocks together, we first have to define the composition. This is done in a further role for the session, presented in the following section.

C. Session and Attacker

The two roles of client and server are combined by defining a role for the session. Session uses the `composition` keyword to couple the two instances of client and server synchronized by common parameters.

```
role session(C,S: agent,
             Kcs: symmetric_key,
             SHA256: hash_func,
             Success, Failure: text,
             Access_accept,Access_reject: text) def=
local
  S1, S2 : channel (dy),
  R1, R2 : channel (dy)
composition
  client(C,S,Kcs,SHA256,Success,Failure,
         Access_accept,Access_reject,S1,R1) ∧
  server(C,S,Kcs,SHA256,Success,Failure,
         Access_accept,Access_reject,S2,R2)
end role
```

The synchronization couples the transitions of the client with the server over their connecting channels. For example, the message `SND(NAS_ID.Success)` of `t2` in client is now being sent over `S1` and coupled via `R2` to the message `RCV(NAS_ID.Success)` of server. The composition that is defined in the role session actually defines the protocol between the roles client (Section III-A) and server (Section III-B) by instantiating their channels such that they mutually connect; the overall protocol is best illustrated graphically (see Figure 4).

The environment represents the attacker and uses a composition, now in turn of two session instances, where the first is one between two agents `c1` and `s1` and the second generalizes the first agent to be `i` – an unspecified agent that triggers the search for intruder possibilities incorporating agents. Note, also that the `SHA256` is given openly to the environment signifying that the attacker knows it and can use it which formalizes the idea of a hash that it is applicable by everyone (see Section II-D).

```
role environment() def=
const c1,s1: agent,
      sha256: hash_func,
      succs, fails: text,
      acc_acp, acc_rej: text,
      kcsk, kisk, kcik: symmetric_key,
      kcs: protocol_id
intruder_knowledge = {c1,s1,sha256,kisk,kcik,
                      succs, fails, acc_acp, acc_rej}
composition
session(c1,s1,kcsk,sha256,succs,fails,acc_acp,acc_rej)
∧
session(i, s1,kisk,sha256,succs,fails,acc_acp,acc_rej)
end role
```

 ISBN: 978-1-61208-201-1

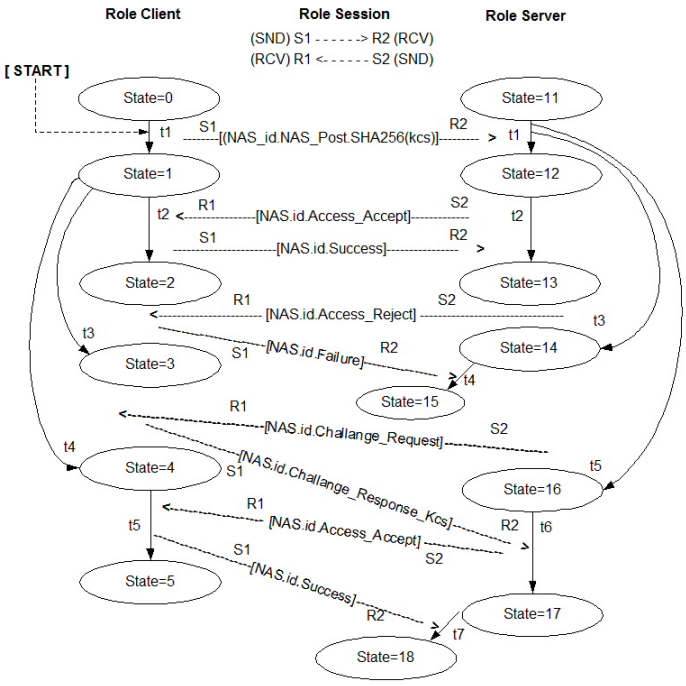


Figure 4. Composition of client and server yields protocol.

The representation is abstract enough to be comprehensible while being in places a bit superficial. We dig deeper down into the lower levels of the Avispa model in the next section to investigate the influence of the hash function on the Radius-SHA256.

D. Security Verification

This section now illustrates how the actual model checking process of the Avispa tool automatically translates the high level protocol model in HLSPL defined in the previous section and performs a complete state analysis over the resulting internal Kripke structure representing this model. The verification is relative to a set of security properties specifying the goals of the authentication that we will illustrate first.

E. Security Properties and Verification Process

Given the implementation of the protocol as described in the previous section, we can now use the inbuilt features of Avispa to verify security in a push-button manner. Avispa provides two features for protocol verification: secrecy of keys and authentication. The secrecy of the server and client keys and authentication of client and server are given as verification commands to Avispa as follows.

```
goal
  secrecy_of sec_c_Kcs, sec_s_Kcs
  authentication_on kcs
end goal
```

The meaning of these two formulas can be illustrated more closely by inspecting their translation into the IF format. We apply all four back-ends OFMC, CLAtSE, SATMC, and TA4SP of the Avispa tool to the Radius-256 specification. For the full IF representation and the performance details of the analysis see [6]. The main observation is that the original security guarantees shown for Radius can be carried over to the protocol Radius-SHA256 by *simply replacing* the hash function MD5 by SHA-256 in the specification. The above secrecy and authentication properties verify just the same.

To understand the effect that the choice of a particular hash function, i.e., MD5, SHA-256, or any other cryptographic hash function has on the security guarantees, we need to inspect the IF version in more detail. First of all, a hash function application in HLSPL like `SHA256(Kcs)` is translated into IF as `apply(SHA256,Kcs)`. According to the Avispa semantics [1], this `apply` operator is reserved for the application of hash functions which manifests itself in the following type.

```
apply(F,Arg) apply: message × message → message
```

However, there seems to be no further semantics attached to the type. The defining properties of a cryptographic hash function are provided implicitly by defining the intruder knowledge for hashes as follows.

```
step gen_apply (PreludeM1,PreludeM2) :=
     iknows(PreludeM1).iknows(PreludeM2) ⇒
     iknows(apply(PreludeM1,PreludeM2))
```

Since the `apply`-operator can produce a hash, the intruder can apply a hash himself but Avispa's intruder semantics provides no rule to inverse a hash function nor any rule enabling collision detection for the intruder.

F. Evaluation and Generalization

Wrapping up the discussed security verification we see that the verification of Radius-SHA256 yields exactly the same guarantees as the classical Radius of RFC 2865/6. In this final section, we show up the consequences of this mechanized verification.

Primarily, the re-engineered modelling and verification for the Radius-256 protocol in Avispa shows that the guarantees of secrecy of keys and mutual authentication that have already been shown for the classical Radius version MD5 equally hold for Radius-256.

Next, our construction process reveals that the exchange of MD5 by another hash function in the Avispa model is simply replacing one (presumably) secure cryptographic hash function by another. As we have observed in the previous section, the Avispa semantics of a hash models hash functions abstractly. Thus, we observe that the verification depends only on the general assumption that *some* hash function is used in the protocol. Therefore, the derived result can be generalized to *all* secure hash functions.

Theorem 1: The Avispa guarantees of secrecy of keys and authentication of the Radius protocol hold for all secure cryptographic hash functions.

 ISBN: 978-1-61208-201-1

Note, however, that this verification does presuppose a secure hash function. That is, the proved result is not valid if the assumed cryptographic strength of the hash function is flawed, like in the case of MD5.

Since the Avispa model cannot cover the implicit part of the hash function security proof, the analysis does not reveal possible attacks. However, the aforementioned attack on SSL [10] could be used as a guideline to produce a similar attack on the classical Radius protocol based on MD5. On the other hand, the generalization presented in this paper is not trivial: its proof relies on the re-engineering of the Radius for SHA-256 and the observation that this re-engineering is applicable to any secure hash function.

IV. Secure Simple Protocol

The protocol SP consists of two parts: the connection part and the data transmission. The connection part establishes a communication between processes A and B to prepare a data transmission according to these established connection parameters. Thereby, a "connected" state is reached during which data may be transmitted before the connection is closed again. During data transmission, SP uses synchronization numbers (`SYNC_NO`) for each message and acknowledgments replying those message numbers to ensure safe transmission. This sequential message numbering can be used as well to secure the protocol against replay attacks, i.e., resending of previously intercepted messages by adversaries. However, to ensure this security, we need to keep the message numbers secret. To do that, we establish a session key in the connection part of SP. We assume that a global public key infrastructure provides certified identities, that is for every principal X on the network we have a signed pair $(K_X, X)_{K_C^{-1}}$ of a public key K_X associated to the principal's identity (for example the MAC of his device). This key-identity pair is signed with the secret key of the certification authority K_C^{-1} and can be verified by both parties A and B even off-line.

Now given this setup, the secure-SP connection part extends the basic exchange of request and reply (`REQ`, `REP`) by additional time stamps T, nonces N (where indices $\in \{A, B\}$ indicate the sender and receiver), sender, and a symmetric session key K_S for the future data transmission. The contents of the following two messages are encrypted using the public keys K_A and K_B so that only the intended recipient A or B can read the message contents.

$$\begin{aligned} A \mapsto B &: \texttt{REQ} + \{\texttt{SYNC\_NO}_A, T_A, A, N_A\}_{K_B} \\ B \mapsto A &: \texttt{REP} + \{\texttt{SYNC\_NO}_B, T_B, B, N_B, N_A, K_S\}_{K_A} \end{aligned}$$

If this two step challenge response protocol succeeds, a connection between A and B is established. In the course of that connection, A and B can now exchange messages whose `SYNC_NO` and shared secrets N_B and T_A are cryptographically protected by the symmetric key K_S that has been exchanged.

$$A \mapsto B \quad : \quad \{\texttt{SYNC\_NO}_A, N_B, T_A, A\}_{K_S} + \text{data message}$$

Note, that the authentication of A to B is only complete after the third step, i.e., the first data transmission, where A shows possession of the private key K_A^{-1} by decrypting and re-encrypting N_B, T_A, and K_S. This protocol has been formalized and successfully verified with Avispa. Confidentiality and integrity of the data communication part holds as long as the session keys are not broken. This additional assumption is necessary and explicit in Avispa: it is beyond the scope of the protocol verification since we abstract from key length and duration of use. The same applies for the above mentioned public key infrastructure.

The secured SP protocol's communication part bears a strong resemblance to the (corrected) Needham-Schroeder asymmetric authentication protocol. This is no surprise, as the NS-asymmetric protocol is the essence of remote authentication.

V. Conclusions

In this paper, we have shown that an adapted version of the Radius protocol using SHA-256 instead of MD5 provides exactly the same security guarantees as the RFC version based on MD5. The verification is a fully automatic analysis in the Avispa toolkit, a specialized model checker for security protocols. We could generalize this result to guarantee security for Radius protocols using secure hash functions, even other than SHA-256. We furthermore illustrated on the example of the simple protocol SP that modelchecking can be used to stepwisely introduce security to a transport layer protocol. The verification process has shown the feasibility of model checking as an engineering tool.

Although the authors of [9] provide a model for the Radius protocol as defined in the RFC, they have failed to sufficiently generalize their results. In some sense, our approach resembles a *refactoring* of the formal model: refactoring is a technique from software engineering supporting the change in software without affecting desired properties; we change the formal model of Radius by replacing MD5 by SHA-256 *without losing desired security properties*. In the process of following the earlier design, we discovered that the model is by no means limited to the classical Radius but can indeed be generalized to a more secure Radius-SHA256, and that this generalization can be extended to arbitrary hashes.

The generalization or refactoring could be an interesting concept to explore because for the working security engineer it provides an easy to use extension making the rather complex model checking process easy to access and provide a practical tool to allow more flexibility in network security engineering. Apart from facilitating the process of protocol engineering, this could also advocate the use of formal

 ISBN: 978-1-61208-201-1

specification and automated model checking in the domain of network security.

REFERENCES

[1] Avispa v1.1 User Manual. Available at http://www.avispa-project.org, [retrieved: April, 2012], 2006.

[2] I.-G. Kim and J.-K. Choi. Formal Verification of PAP and EAP-MD5 Protocols in Wireless Networks: FDR Model Checking. *AINA*. 2004.

[3] Y. Kirsal-Ever. Development of Security Strategies using Kerberos in Wireless Networks. PhD Thesis, Middlesex University, 2011.

[4] G. Lowe. Breaking and Fixing the Needham-Schroeder Security Protocol. *Information Processing Letters*, Elsevier, 1995.

[5] yRFC2: The Simple Protocol (SP) Specification, 12.1.2012. http://www.mdx.ac.uk/Assets/yrfc2_sp_protocol.doc, [retrieved: April, 2012].

[6] S. Patel. Implementation and Analysis of Radius Protocol using Avispa. Master's Thesis. Middlesex University, 2011.

[7] R. Rivest. Message-Digest MD5. *Network Working Group, RFC: 1321*. http://www.kleinschmidt.com/edi/md5.htm[retrieved: April, 2012], 1992.

[8] A. S. Sani. Verifying the Secured Simple Protocol in AVISPA. Master's Thesis, Middlesex University, 2012.

[9] V. Sankhla. Formalisation of Radius in Avispa. http://www.avispa-project.org/library/RADIUS-RFC2865, [retrieved: April, 2012], University of Southern California, 2004.

[10] Rogue CA certificate signed by a commercial Certification Authority. http://www.win.tue.nl/hashclash/rogue-ca/\#sec71 [retrieved: April, 2012], Presented at the 25th Chaos Communication Congress, Berlin 2008.

[11] X. Wang and H. Yu. How to Break MD5 and Other Hash Functions. *Advances in Cryptology, Eurocrypt 2005*. LNCS **3439**, Springer 2005.

[12] Wikipedia RADIUS, http://en.wikipedia.org/wiki/RADIUS, [retrieved: April, 2012].

[13] The Ycomm Framework. Offical Web-Site, Middlesex University. http://www.mdx.ac.uk/research/areas/software/ycomm_research.aspx, [retrieved: April, 2012].

 ISBN: 978-1-61208-201-1

Firewall Analysis by Symbolic Simulation

Arno Wagner
Consecom AG
Zurich, Switzerland
Email: arno.wagner@consecom.com

Ulrich Fiedler
Bern University of Applied Sciences
Biel, Switzerland
Email: ulrich.fiedler@bfh.ch

***Abstract*—When doing Layer 4 security analysis on a chain of firewalls, the analyst is faced with the problem of combining them into a unified representation in order to verify reachability though the chain and possibly compare it with a security policy. Doing this manually is labor-intensive and becomes infeasible if firewalls with large configurations are part of the chain. To automate the unification process, we have created the Consecom Network Analyzer that uses symbolic simulation with an interval representation to generate a unified equivalent firewall in a normalized, simple and flat form. We show the suitability of this approach for firewalls with large configurations by giving benchmarks based on deployed rule-sets. We also demonstrate the effects of different optimization techniques on run-time and memory footprint. The Consecom Network Analyzer has already been used successfully for security reviews.**

Keywords-Firewall Analysis; Symbolic Simulation.

I. Introduction

This paper describes the *Consecom Network Analyzer* (CNA), which is the result of a collaboration between academia and industry. The CNA is a tool-set that greatly reduces the effort, and thereby cost, for practical firewall security analysis in the presence of large firewall configurations.

A firewall security analysis is one type of network security review. It is often done on network Layer 4, for example for TCP and UDP traffic. Figure 1 shows the basic scenario. The typical steps to be done include:

1) Normalize firewall configurations
2) Identify critical network paths
3) Identify firewalls along each critical path
4) Determine network reachability on critical paths
5) Compare reachability and security requirements
6) Identify non-compliant firewall rules

The primary motivation for creating the CNA lies in steps 4, 5 and 6. In step 4, the CNA calculates the reachability in a unified simple format that has firewall rules attached as trace information. If a formalized or easy to formalize security policy is available, it can be compared automatically to the actual network reachability. As such a security policy is often not available in practice, step 5 may still need to be done manually.

Figure 2 shows the typical application scenario. The Rule-Set Converter is not part of the core CNA system and has to be adapted for each different firewall description format. The CNA uses a normalized symbolic Layer 4 format internally that is based on intervals. As core contribution of this paper, we show this representation is suitable for calculating reachability even in the presence of large firewall configurations. To this end, we present benchmark calculations on deployed rule-sets. The CNA has been used successfully in firewall security reviews.

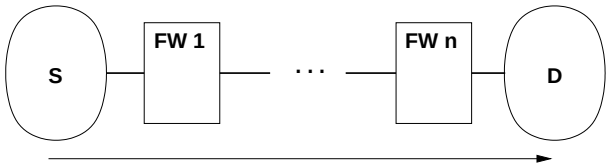


Fig. 1. Unidirectional reachability along a critical network path.

The paper is organized as follows: Section II introduces our network and firewall model, and the symbolic representation used. Section III gives the operations used for single firewalls. Section IV explains how to calculate unidirectional reachability. A complexity analysis is sketched briefly in Section V. Section VI describes the implementation, while Section VII states benchmark results and the effects of different optimization techniques. Section VIII explains how to extend the approach to two-sided reachability and to automated comparison with a policy. The paper finishes with a discussion of related work in Section IX and a conclusion in Section X.

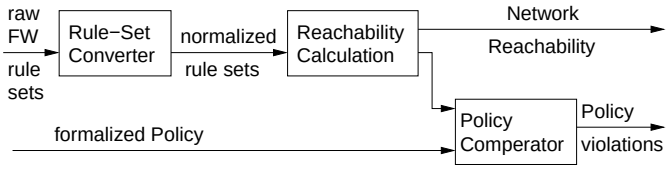


Fig. 2. Typical analysis data-flow with the CNA.

II. Approach

The reachability calculation process starts with a representation of the initial reachability (disregarding firewalls), which will often be unconstrained. This initial reachability is then successively reduced by applying firewall configurations. The end-result is a flat, unified representation of the firewall-chain, restricted by the initial reachability.

 ISBN: 978-1-61208-201-1

A. Network Model

We are primarily interested in network reachability as restricted by firewalls. Given a source network S, sequence of firewalls $\mathrm{FW}_1, \ldots, \mathrm{FW}_n$ and a destination network D (see also Figure 1), we say that D is *reachable* from S if there are network packets that can traverse $\mathrm{FW}_1, \ldots, \mathrm{FW}_n$ without being dropped by any FW_i. Note that some attacks will need *two-sided reachability*. For example services used over TCP can usually only be attacked if response packets can traverse the firewall sequence in reverse order. See Section VIII-A for a discussion on how to check two-sided reachability.

We restrict the packet information visible for firewalls to IP addresses and ports, which results in a Layer 4 model. Each protocol is treated separately, although it is possible to mix protocols, for example by doing a forward analysis with TCP and a backward analysis with ICMP in order to determine whether an ICMP response to a TCP packet would get through. Routing is out of scope for this work, as we do not see it as a security mechanism; see Section IV-A for a brief discussion.

B. Subspaces, Boxes and Intervals

Reachability is represented by subspaces of

$$M = \text{src IP} \times \text{src port} \times \text{dst IP} \times \text{dst port}$$

We organize these subspaces into sets (lists) of axis-aligned hyperrectangles in M, also called *axis aligned boxes* [1] (or simply *box* for short), with

$$\begin{aligned} A &\subseteq M \text{ is represented as} \\ A &= \{b_1, \ldots, b_n\} \text{ with } b_i \in M \text{ and } b_i \text{ is a box.} \end{aligned}$$

In this paper, boxes will always be axis-aligned. A box can be represented as a 4-tuple of intervals, which allows symbolic computations. This representation is similar to the one used in [2].

Box example:
$b = (10.0.0.0 - 10.0.0.255, 1024 - 65535, 10.1.1.1, 80)$

We use intervals with wrap-around, where IP and port number spaces are regarded as circles. This facilitates representing complements. Figure 3 gives graphical examples of three boxes in two dimensions represented this way.

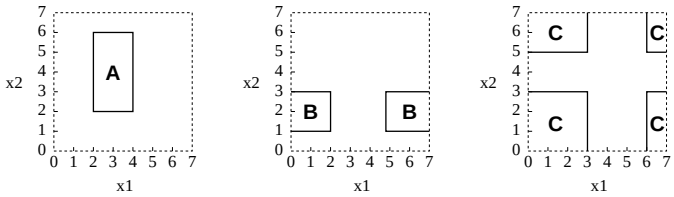


Fig. 3. Example: Boxes A,B and C in two dimensions.

C. Firewall Model

The CNA uses a simple firewall model, where each firewall consists of a linear sequence of rules r that each have a box describing their applicability and one of the target actions *accept* or *drop*, with a default *drop* at the end of sequence. This corresponds to the "simple" model of [3].

D. Rule Application and Set Operations

In order to apply a firewall rule $r = (b, <\text{action}>)$ to a subspace $A = \{b_1, \ldots, b_n\} \subseteq M$, we intersect b with the different b_i in turn and apply the action to the result $A \cap \{b\} = \{b \cap b_1, \ldots, b \cap b_n\}$.

The usual set operations are defined on boxes and, by extension, on subspaces of M. Some deserve additional comments.

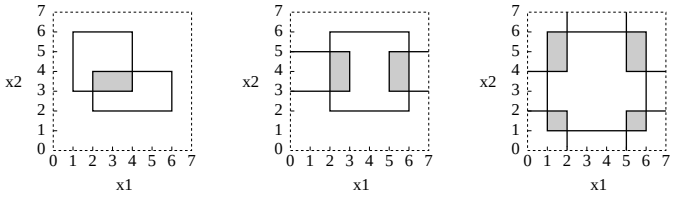


Fig. 4. Box intersection, as used in rule application, shown for two dimensions.

Intersection: Intersecting two boxes in d dimensions can have up to 2^d result boxes. Figure 4 illustrates this in two dimensions. For $b_1, b_2 \in M$, the intersection $b_1 \cap b_2$ may consist of up to 16 boxes as M has 4 dimensions.
Box complement: The complement of an interval is derived by adjusting the boundaries. The complement of a box is derived by complementing each interval in turn and setting all other intervals to full range. Hence, a 4-dimensional box has four boxes as its complement.
Subtraction: Calculating $a - b$ for boxes a and b is done by using the relation $a - b = a \cap \bar{b}$ from set calculus.

III. Restricting Reachability by a Single Firewall

The core operations used in determining reachability through a single firewall are `apply_firewall()` and `apply_rule()`, shown in Figure 5 in simplified form. The task of `apply_firewall()` is to take a given reachability description, stated as a set of boxes, called here a *Work Set* (WS) and, using the rules of the firewall, determine both an *Accept Set* (AS), which is the part of the WS that can pass the firewall, and a *Drop Set* (DS) that is the part of the WS that cannot pass the firewall. AS and DS are represented as sets of boxes. The function `apply_rule()` forms the basis of `apply_firewall()` and implements calculation of the intersection I between the given rule and WS. The intersection I is then added to the AS for an *accept* rule or to the DS for a *drop* rule.

 ISBN: 978-1-61208-201-1

Building on these two operations, more complex operations can be constructed. Note that `apply_rule()` may attach trace information to boxes, for example to document rule application. If desired, the full history of each box can be recorded in the trace. This allows to determine the specific firewall rules that are responsible for a box being in the final reachability, and represents information needed in any report about firewall configuration problems.

```
apply_firewall(WS, FW):
  AS := ∅      /* Accept Set */
  DS := ∅      /* Drop Set   */
  for r ∈ in FW:     /* r: box of a rule */
    I := apply_rule(WS, r)
    WS := WS - I     /* reduce Work Set  */
    if r is accept:  AS := AS ∪ I
    if r is drop:    DS := DS ∪ I
  return(AS, DS)

apply_rule(WS, r):
  I := ∅
  for b ∈ WS:        /* b is a box */
    i := b ∩ r
    I := I ∪ i
  return(I)
```

Fig. 5. Pseudo-code for `apply_firewall()` and `apply_rule()` (simplified).

IV. Unidirectional Reachability Computation

Pseudo-code for the calculation of one-direction reachability through a sequence of firewalls is given in Figure 6. We will typically choose the initial reachability as unrestricted. Starting with full, unconstrained reachability will ensure the final results only rely on the given firewall configurations. A more restricted initial reachability can still be used when appropriate. Ports are unconstrained in the initial reachability.

A. Comments on Routing

Routing can usually not be regarded as security feature in practice and is not seen as one by many customers. There are several reasons for this:

- The primary task of routing is to get packets to a specific destination, while the primary task of a firewall is to prevent packets reaching a specific destination. Routing configuration and firewall configuration hence have diametrically opposed primary tasks and this is reflected in procedures and mind-sets.
- Due to the different primary tasks, often the teams responsible for routing and for firewalls are different.
- While firewall configurations are handled securely and all updates are done with the security model in mind, routing configurations are typically changed with the network model in mind and handled in a less secure fashion. Routing is hence easier to compromise.
- Sometimes customers cannot even specify the IP ranges of `S` and `D` precisely, but have precise firewall information. This may sound surprising, but if routing delivers more to a physical target network than expected, this is not necessarily a problem. For firewalls, it is a critical error.
- Routing works on Layer 3, while firewalls work on Layer 4. Mixing the two complicates things and increases maintenance effort.

Overall, it is far more practical to separate routing and firewalls and to require that all restrictions on reachability must be implemented by firewalls placed into the critical network paths. This is especially true for customers with complex firewall configurations.

It should be noted that with this approach, the question arises whether a specific firewall actually is on the critical network paths it is supposed to be on. Answering this question requires a network topology analysis and is outside of the scope of this work.

It should also be noted that network scanning always takes routing into account. This is a fundamental limitation of network scanning.

```
in:  S, D  /* Source, Destination networks */
     FW1, ..., FWn   /* firewalls          */
out: ASn             /* final reachability */
     DS1,...,DSn     /* Drop Sets          */

WS1 := S × <all> × D × <all>
(AS1, DS1) := apply_firewall(WS1, FW1)
WS2 := AS1
(AS2, DS2) := apply_firewall(WS2, FW2)
WS3 := AS2
...
(ASn, DSn) := apply_firewall(FWn − 1, WSn − 1)
```

Fig. 6. Pseudo-code for calculating unidirectional reachability with `apply_firewall()` for the scenario shown in Figure 1.

V. Algorithmic Complexity

We briefly sketch the complexity analysis idea. For a worst-case scenario, start with one box and a single firewall with n drop rules. Each drop rule can split (asymptotically) at most one element of the Work Set into a maximum of 2^d (with dimension $d = 4$) non-overlapping parts that are kept in the working set. Hence, each rule increases the size of the working set by a maximum of 16, giving an overall space complexity of the result of $16 * n \in O(n)$. As each successive rule application has to work on 16 more boxes, time complexity is $1 * 16 + 2 * 16 + \ldots + n * 16 = 16 * (1 + 2 + \ldots + n) = \frac{16}{2} n(n-1) \in O(n^2)$. A very similar argument applies to accept rules and mixed rule-sets.

In comparison, in [4], the authors need worst case effort $O(n^4)$ to build a Firewall Decision Diagram (FDD) for n firewall rules with for our firewall model. It is reasonable to

 ISBN: 978-1-61208-201-1

No	FW / FW seq.	rule-set size raw	nor-malized	opt.	benchmark results Python baseline		input opt.		trace reduction		core-loop ported to C	
1	S	27	2'000	180	8min	12MB	6s	6MB	5s	6MB	0.2s	6MB
2	M	67	23'000	8300	-	-	546min	84MB	222min	48MB	48s	19MB
3	L	170	27'000	3100	-	-	34min	26MB	20min	18MB	4s	10MB
4	S, M				100min	32MB	294s	14MB	240s	13MB	3s	14MB
5	M, S				-	-	544min	81MB	336min	48MB	49s	19MB
6	M, L				5000min	187MB	660min	77MB	250min	56MB	69s	22MB
7	S, M, L				205min	58MB	370s	16MB	305s	16MB	4s	16MB

TABLE I
BENCHMARKS (TCP)

expect that this worst-case is extremely unlikely to happen in practice.

In [3], the authors claim a worst case complexity of $O(n)$ for processing a firewall with n rules in their "simple model". However, they assume constant effort for set operations on their accept (A) and drop (D) sets. While the BDDs used are typically very efficient set-representations, they do not reach $O(1)$ worst-case effort for set operations and the correctness of the given complexity analysis seems doubtful.

VI. Implementation

The CNA is implemented in Python 3 [5] with C extensions. This allows a clean and flexible OO design and facilitates targeted optimizations. IP addresses and port numbers are represented directly by Python integers. Boxes are represented as Python 8-tuples (representing 4 intervals) and encapsulated into class objects in order to allow attachment of traces, annotations and firewall rule actions. Subspaces are represented as Python lists. The pure-Python prototype is relatively slow and has high memory consumption, but can already be used for security reviews involving firewalls with small and medium-sized rule-sets.

VII. Optimizations and Benchmarks

First, note that in the absence of Network Address Translation (NAT), which is rarely deployed in security critical networks, firewalls can be arbitrarily reordered, as exactly those packets that make it through *all* of them are part of the final reachability space. In particular, a good selection of the first firewall to be processed can have significant performance benefits. Benchmarks must therefore always be seen together not only with the relevant firewall configurations, but also their processing order.

A. Benchmarks

In order to determine performance and to examine the performance impact of different optimizations, we give a selection of benchmark results[1] in Table I. Times are CPU times including input data parsing. Memory sizes are the whole process memory footprint, excluding shared areas (libraries). The calculations were done using Linux (Debian Squeeze 32bit) on an Athlon64 X2 5600+ CPU, using only one CPU at a time. Memory was set to the 4GB memory model and the machine was running kernel 2.6.38 from kernel.org without any special optimizations. Python versions used include 3.0 and 3.1 with no significant differences in performance between the two.

Lines $1, 2, 3$ of Table I describe the firewall configurations used. These are firewall configurations deployed in the real world. They have a flat form (no sub-chains) and a default-drop policy.

Lines 4ff. of Table I give benchmarks for different firewall combinations. The order of the firewalls is important as the first one has to be completely represented in memory, which causes effort $O(|\mathrm{FW}_1|^2)$ (where $|\mathrm{FW}_k|$ is the number of rules in firewall FW_k). The effort for each additional firewall in the chain is $O((|\mathrm{WS}_i|+|\mathrm{FW}_i|)\cdot|\mathrm{FW}_i|)$ and hence higher in the worst case. But when starting with a firewall with small rule-set, we observed that a later combination with a firewall with a large rule-set does often not increase the WS size significantly, as most rules of the larger firewall do not apply. For that case, the complexity goes effectively down to $O(|\mathrm{WS}_i| \cdot |\mathrm{FW}_i|)$, which is a lot smaller than $O(|\mathrm{FW}_i|^2)$ if $|\mathrm{FW}_i|$ is is large but $|\mathrm{WS}_i|$ is small. If the firewall processed first has a much larger rule-set than the others, we have observed that it will often dominate the runtime.

The columns "rule-set size" give the number of rules in the raw input in vendor format, the normalized number of rules without optimization and the optimized rule-set size. Benchmarks are given only for TCP for brevity, UDP and ICMP analysis have comparable results. We do not have benchmarks for comparison against a policy, as we do not have a sufficiently formalized policy and hence looking directly at reachability was more efficient. Comparison with a policy would incur effort comparable to adding one more firewall configuration in the size of the negated policy specification. The idea is that nothing must be able to pass though the given firewall chain *and* an additional firewall representing the negated policy, with the negated policy

[1] As with all benchmarks, it should be noted that the stated results only give a rough idea about runtime, memory footprint and effects of different optimizations.

 ISBN: 978-1-61208-201-1

representing all forbidden traffic.

As can be seen in Table I, each evaluated optimization step has significant impact on observed run-time. The final implementation with all optimizations included has very reasonable performance even in the presence of firewalls with large rule-sets.

B. Firewall Evaluation Sequence Optimization

The benchmarks demonstrate that the selection of the first firewall to be processed has a huge impact on performance. For the first firewall, the Work Set grows for each rule application, while for later firewalls only rules that have a non-empty intersection with the Work Set can increase Work Set size. Our experiences show that the most restrictive firewall configuration should be processed first. In many scenarios, this will be the smallest firewall configuration, measured in number of rules.

C. Rule-Set Representation Optimization

Firewall configuration in vendor-formats often allow more complex specifications, such as lists of multiple sources, destinations or services. Decomposing such input rules into rules using a single box each can results in a number of normalized rules that is a lot higher than needed. The reason is that many resulting rules will be overlapping or adjacent in such a way that they can be combined. The column "opt." under "rule-set size" in Table I states the reduced number of rules after optimization and the column "input opt." gives the improved run times and memory footprints. The runtime for the input optimization itself is small, as it only works with a focus of one raw input rule at a time.

Note that global box combination would be possible, but combining boxes from different raw rules has two problems: First, if both *accept* and *drop* rules are present, the combination algorithm has to take rule sequence into account. And second, in this approach a box cannot be labeled with the single raw firewall rule it originated from. This makes the identification of policy-violating rules in the end-result difficult.

D. Trace Reduction

While the original prototype retained traces for all operations that changed a box, it turns out these full traces are only beneficial for debugging. In a security analysis, only *accept* and *drop* actions are relevant and hence it is enough to add trace information to a box when it is added to an Accept Set or Drop Set. It is not necessary to trace when boxes are reduced or split in the WS. Hence, traces were reduced accordingly. This also means that there can be at most one trace entry per firewall in each box contained in the result. The column "trace reduction" in Table I states the additional performance gains. Note that trace reduction was benchmarked with input optimization applied as well.

E. Core-Loop Ported to C

In a last step, the core loop function `apply_rule()` was ported to C and embedded into the Python code. Contrary to Figure 5, WS, AS and DS are passed to `apply_rule()` and are manipulated in-place according to the rule action. This puts expensive operations, such as data-structure manipulations, into the C code. No other special optimizations were done for the C code and in particular the standard GNU libc memory allocator was used. The column "core-loop ported to C" in Table I states final performance figures. Note that trace reduction and rule-set representation optimization was applied as well.

In addition, we performed a benchmark calculation for deployed firewall configuration "XL". It has a normalized rule-set size of 2.8 million rules, which reduces to 300'000 rules after input optimization. Raw rule number is 95. Representing configuration XL in memory took 20h of CPU time and resulted in a memory footprint of 900MB. This shows that firewall configurations of this size can still be processed with the CNA with reasonable effort.

VIII. Advanced Analysis

A. Computing Two-Sided Reachability

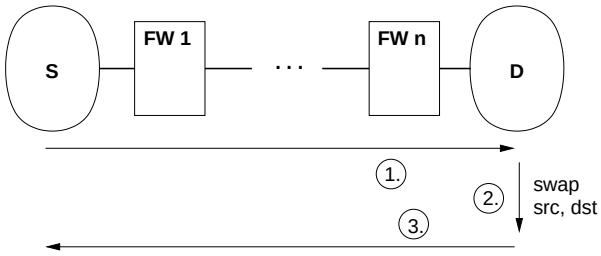


Fig. 7. Calculating bidirectional reachability.

Two-sided Reachability allows determining whether an agent in the source network S can use a service offered in the destination network D that needs a *connection*, for example any service offered over TCP. It also allows limited comparison with scan results (for example from `nmap` [6]), which are sometimes used to verify a firewall deployment. Figure 7 gives the idea on how to obtain a two-sided reachability result.

B. Verifying Policy Compliance

Policies can be represented as an undesired reachability U, with the meaning that if anything in $U \subseteq M$ is actually reachable through the firewalls, then the policy is violated. How policies are obtained and converted into this format is outside of the scope of this work.

To test policy compliance, the actual network reachability A on each critical network path is calculated. Let V be the policy-violating reachability. Then $V = A \cap U$. If V is non-empty, all elements of V represent violations. The non-compliant firewall rules can be identified by looking at the trace information attached to elements of V, which

 ISBN: 978-1-61208-201-1

they inherit from A. Other compliance tests are possible and can be implemented when needed.

IX. Related Work

Reachability Analysis: One alternative to using the CNA is network scanning, for example with nmap [6]. It should be noted however that this suffers from the limitations that routing affects scanning and that normal scanning cannot find undesired unidirectional reachability.

Algorithmic Firewall Analysis: It is possible to formalize firewall functionality with a suitable logic and then use approaches from automated theorem proving to derive properties and check against violation of conditions. Work in this area includes FIREMAN [3] by Yuan, Mai, Su, Chen, Chuah and Mohapatra, which uses a BDD (Binary Decision Diagram) representation. The idea of using BDDs is developed further by Liu and Gouda [4], [7], with the introduction of Firewall Decision Diagrams (FDDs).

The query-engine of Mayer, Wool and Ziskind [8] uses a different approach. It answers questions on whether a specific packet would traverse a set of firewalls by using a rule-based simulator. This is mostly useful to determine the impact of specific firewall configuration changes. Its value in a complete firewall security analysis is limited. The Margrave Tool [9] uses a similar approach.

Commercial Tools: A commercial firewall analyzer is offered by AlgoSec [10]. This tool seems to be targeted at maintenance and administration of large numbers (up to 1000) of firewalls. Commercial firewall maintenance tools with limited audit capabilities are also offered by Tufin [11] and FireMon [12].

X. Conclusion and Future Work

We have designed and implemented the CNA (Consecom Network Analyzer), a tool that calculates network reachability through a series of firewalls given as a Layer 4 abstraction by symbolic simulation. The primary use is for real-world security audits that examine firewalls with large rule-sets. While using set operations to model firewalls is simple, to the best of our knowledge we are the first to demonstrate that an abstraction based on intervals for reachability and firewall rules is efficient enough to calculate reachability through large deployed firewall configurations in practically useful time and with moderate memory footprint, while at the same time retaining the capability to annotate each result sub-set with a full trace of the applied firewall rules. Automated result annotation is essential when analyzing firewall chains that include firewalls with a large number of rules.

One possible direction for future work is optimizing the CNA. First, the representation of the Work Set can be improved. Using ideas from geometric search, the Work Set could be organized into a data-structure that efficiently allows searching for all boxes that intersect a given box. This could speed up rule application significantly. A second possible optimization direction is optimization of memory management. Run-times and memory footprint could be improved by reducing traces to a fixed size format and by providing a custom allocator to the core-loop. Finally, the CNA could be adapted to handle *IPv6* in the future. This may need specific performance optimizations. We plan to defer IPv6 adaption until there is market demand.

Acknowledgement: We thank the Swiss KTI and Consecom AG for funding parts of this work and the anonymous reviewers for helpful suggestions on how to improve the paper.

References

[1] "Wikipedia: Hyperrectangle," http://en.wikipedia.org/wiki/Hyperrectangle, last visited January 2012.

[2] P. Eronen and J. Zitting, "An expert system for analyzing firewall rules," in *Proc. 6th Nordic Worksh. Secure IT Systems*, 2001, pp. 100–107.

[3] L. Yuan, J. Mai, Z. Su, H. Chen, C.-N. Chuah, and P. Mohapatra, "FIREMAN: A Toolkit for FIREwall Modeling and ANalysis," in *IEEE Symposium on Security and Privacy*, 2006, pp. 199–213.

[4] A. X. Liu and M. G. Gouda, "Diverse firewall design," in *IEEE Transactions on Parallel and Distributed Systems, 19(8)*, August 2008.

[5] "Python Homepage," http://python.org/, last visited January 2012.

[6] "Nmap Security Scanner," http://nmap.org/, last visited January 2012.

[7] A. X. Liu and M. G. Gouda, "Firewall policy queries," in *IEEE Transactions on Parallel and Distributed Systems, 20(6)*, June 2009.

[8] A. J. Mayer, A. Wool, and E. Ziskind, "Offline firewall analysis," *Int. J. Inf. Sec.*, vol. 5, no. 3, pp. 125–144, 2006.

[9] T. Nelson, C. Barratt, D. J. Dougherty, K. Fisher, and S. Krishnamurthi, "The Margrave Tool for Firewall Analysis," in *USENIX Large Installation System Administration Conference (LISA)*, 2010.

[10] "Algosec Homepage," http://www.algosec.com/, last visited January 2012.

[11] "tufin Homepage," http://www.tufin.com/, last visited January 2012.

[12] "FireMon Homepage," http://www.firemon.org/, last visited January 2012.

 ISBN: 978-1-61208-201-1