IARIA

SECURWARE 2011

The Fifth International Conference on Emerging Security Information, Systems and Technologies

ISBN: 978-1-61208-146-5

August 21-27, 2011

Nice/Saint Laurent du Var, France

SECURWARE 2011 Editors

Masaru Takesue, Hosei University, Japan

Rainer Falk, Siemens AG - München, Germany

SECURWARE 2011

Foreword

The Fifth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2011),held between August 21-27, 2011 in Nice/Saint Laurent du Var, France, was a multi-track event covering topics related to theory and practice on security, cryptography, secure protocols, trust, privacy, confidentiality, vulnerability, intrusion detection and other areas related to low enforcement, security data mining, malware models, etc

Security, defined for ensuring protected communication among terminals and user applications across public and private networks, is the core for guaranteeing confidentiality, privacy, and data protection. Security affects business and individuals, raises the business risk, and requires a corporate and individual culture. In the open business space offered by Internet, it is a need to improve defenses against hackers, disgruntled employees, and commercial rivals. There is a required balance between the effort and resources spent on security versus security achievements. Some vulnerability can be addressed using the rule of 80:20, meaning 80% of the vulnerabilities can be addressed for 20% of the costs. Other technical aspects are related to the communication speed versus complex and time consuming cryptography/security mechanisms and protocols.

Digital Ecosystem is defined as an open decentralized information infrastructure where different networked agents, such as enterprises (especially SMEs), intermediate actors, public bodies and end users, cooperate and compete enabling the creation of new complex structures. In digital ecosystems, the actors, their products and services can be seen as different organisms and species that are able to evolve and adapt dynamically to changing market conditions.

Digital Ecosystems lie at the intersection between different disciplines and fields: industry, business, social sciences, biology, and cutting edge ICT and its application driven research. They are supported by several underlying technologies such as semantic web and ontology-based knowledge sharing, self-organizing intelligent agents, peer-to-peer overlay networks, web services-based information platforms, and recommender systems.

To enable safe digital ecosystem functioning, security and trust mechanisms become essential components across all the technological layers. The aim is to bring together multidisciplinary research that ranges from technical aspects to socio-economic models.

We take here the opportunity to warmly thank all the members of the SECURWARE 2011 Technical Program Committee, as well as the numerous reviewers. The creation of such a broad and 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 SECURWARE 2010. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the SECURWARE 2011 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that SECURWARE 2011 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the area of information security, security systems and technologies.

We hope Côte d'Azur provided a pleasant environment during the conference and everyone saved some time for exploring the Mediterranean Coast.

SECURWARE 2011 Chairs

Advisory Chairs
Juha Rőning, University of Oulu, Finland
Catherine Meadows, Naval Research Laboratory - Washington DC, USA
Petre Dini, Concordia University, Canada / China Space Agency Center - Beijing, China
Reijo Savola, VTT Technical Research Centre of Finland, Finland
Masaru Takesue, Hosei University, Japan
Mariusz Jakubowski, Microsoft Research, USA

Industry Liaison Chair
Rainer Falk, Siemens AG - München, Germany

Research/Industry Chair
Mariusz Jakubowski, Microsoft Research, USA

SECURWARE 2011

Committee

SECURWARE Advisory Chairs

Juha Rőning, University of Oulu, Finland
Catherine Meadows, Naval Research Laboratory - Washington DC, USA
Petre Dini, Concordia University, Canada / China Space Agency Center - Beijing, China
Reijo Savola, VTT Technical Research Centre of Finland, Finland
Masaru Takesue, Hosei University, Japan
Mariusz Jakubowski, Microsoft Research, USA

SECURWARE 2011 Industry Liaison Chair

Rainer Falk, Siemens AG - München, Germany

SECURWARE 2011 Research/Industry Chair

Mariusz Jakubowski, Microsoft Research, USA

SECURWARE 2011 Technical Program Committee

Imad Abbadi, Oxford University, UK
Ali Ahmed, University of Najran, Saudi Arabia
Hasan Ibne Akram, Munich University of Technology - Garching b. München, Germany
Claudio Agostino Ardagna, Università degli Studi di Milano, Italy
Carlos Aguilar, Université de Limoges, France
Ioannis Askoxylakis, FORTH-ICS & FORTHcert, Greece
Steve Barker, King's College/London University, UK
Ilija Basicevic, University of Novi Sad, Serbia
Feng Bao, Institute for Infocomm Research, Singapore
Georg T. Becker, University of Massachusetts Amherst, USA
Francisco Jose Bellido Outeiriño, University of Cordoba, Spain
Jun Bi, CERNET, China
Catalin V. Birjoveanu, "Al.I.Cuza" University of Iasi, Romania
Carlo Blundo, Università di Salern, Italy
Wolfgang Boehmer, Darmstadt University of Technology (TUD), Germany
Ravishankar Borgaonkar, Technical University Berlin, Germany
Ahlem Bouchahda, SUP'COM Tunis, Tunisia
Jeremy Briffaut, ENSI Bourges, France
Dariusz Caban, Wroclaw University of Technology, Poland
Christian Callegari, University of Pisa, Italy
Stelvio Cimato, Università degli studi di Milano – Crema, Italy
Nora Cuppens, Telecom-Bretagne, France
Frédéric Cuppens, Telecom-Bretagne, France
Ernesto Damiani, University of Milan, Italy

Table of Contents

Lightweight Detection of Spamming Botnets

Masaru Takesue
Dept. Applied Informatics, Hosei University, Tokyo 184-8584 Japan
E-mail: takesue@ami.ei.hosei.ac.jp

***Abstract*—A botnet is one of the largest problems against the Internet society because it is used to mount Distributed Denial of Service (DDoS), to steal users credentials, to send spam email, and so on. To cope with the problem, this paper presents a method of detecting spamming botnets, exploiting the information in a hash table of spams produced in our spam filter. To partition the spams based on the similarity of their messages, we cluster the spams in the hash table in three steps by 1) removing the collision (due to the hashing) in each bucket of the table, 2) merging the clusters obtained in step 1), using the fingerprints of message bodies, and 3) further merging the second-step clusters based on the spam-specific words in the *Subject* headers of spams. We identify a bot using the IP address in the first internal *Received* header that is prepended to the list of *Received* headers by the first internal server of a receiving organization. By simulation, we can cluster about 18,000 real-world spams in about 4,000 seconds with no misclustering on our commodity workstation. The active IP space for bots to send spams is almost the same as the one reported in the literature, except of a slight expansion.**

***Keywords*-Spam, bot, botnet, clustering, fingerprint, spam-specific word.**

I. Introduction

A botnet consists of the home and office computers infected with bot malware and controlled by a botmaster. Botnets are one of the largest problems against the Internet society since they are used to mount serious attacks such as Distributed Denial of Service (DDoS) and users' credentials stealing, and to misuse and waste network resources and recipients' invaluable time by sending spam email (e.g., about 88% to 92% of spams originate from botnets [1]). The size of a botnet ranges from tens of hosts to more than ten thousand hosts [2]; surprisingly, a botnet that Symatec discovered consists of about 400 thousand hosts [3].

Since a large percentage of emails originate from botnets, an analysis of spam email is effective to identify bots and botnets that are spamming [2][4], though there are several approaches to the identification of bots and botnets that are not necessarily spamming (as described in Section II). An email message consists of several kinds of headers and a body. One *Received* header is prepended to the list of *Received* headers every time the email is relayed. Then the list would provide information about the route from the origin to the recipient of the email.

The botmaster usually obfuscates the route information so that it is difficult to identify the origin of spamming bot. The only exception is *the first internal line*, i.e., the *Received* header prepended to the list of *Received* headers by the first internal server of the organization the recipient belongs to [5]. The first internal line gives the IP address of the outside machine that directly delivered the email across the Internet to the receiving organization. The *Received* headers before the first internal line can be falsified by the sender.

This paper presents a method of detecting botnets. For a continuous detection of spamming bots, we integrate botnet detection into spam filtering since currently every organization or user installs a spam filter. Then we can detect spamming botnets worldwide if the botnets detected in the individual spam filters are periodically sent to, for instance, a Botnet Analysis Center to figure out spamming botnets worldwide. We assume our spam filter [6] in the paper.

The botnet detection integrated in spam filtering has two problems: First, we can use only restricted kinds of information for the detection because it is produced in the filter. Second, to identify the member bots of each botnet, we partition the spams classified by the filter into clusters each corresponding to a botnet. Then the clustering has to be lightweight enough for a home or office computer since the clustering of a large number of objects generally needs a very large computing power [7][2].

Our filter detects spam based on the fingerprints of message bodies and spam-specific words in the *Subject* headers of the received email. We start the clustering with a hash table of spams produced in the spam filter, since each bucket of the table is a linked list of spams that have the same hash value. We cluster spams in a hierarchical way to reduce the computation power required in the clustering.

After the clustering, we detect the IP addresses in the first internal lines of the spams in each cluster to identify the botnets and their members. Experimental results show that the clustering of about 18,000 real-world spams can be performed with no misclustering in about 4,000 seconds on our commodity workstation. The active IP space for bots to send spams is almost the same as the one reported in [8], though some new subspaces appear to be emerging. The size distribution of detected botnet significantly differs, depending on the size of partial IP addresses (i.e., the upper 8, 16, 24, and 32 bits) used in the botnet detection.

In the rest of the paper, Section 2 describes related work. Section 3 outlines our spam filter and proposes the scheme for clustering spamming bots. Section 4 describes experimental results, and Section 5 concludes the paper.

II. Related Work

To communicate with the bots in a botnet, its botmaster needs a command and control (C&C) channel. The C&C channels with the Internet Relay Chat (IRC) are currently dominant, though the channels with the HTTP or P2P are appearing to remedy the weak point of centralized structure of IRC channels. We classify the approaches to bot detection into four categories; network-based, C&C-based, DNS-based, and spam-based approaches.

The approach based on network traffic detects bots by observing 1) the session parameters such as a source address/port, a destination address/port, the number of packets and bytes, the start and end time of the session, and the transport layer protocol used [9], 2) the network activity on suspicious IRC and HTTP traffic such as port scanning, spamming, and binary down loading [10], or 3) Windows API socket function calls and their arguments [11].

C&C-based approach exploits 1) behavioral characteristics of IRC bots such that bots are idle most of the time and respond faster than a human upon receiving a command [12] and 2) regularity or invariant characteristics in botnet behavior such as one-to-many connection, simultaneous and immediate responses from bots in a small fixed period, and synchronized malicious actions by bots [13], or 3) flow characteristics of IRC C&C traffic such as bandwidth, duration, and packet size and timing [14].

In a DNS-based approach, the connections to botmasters are redirected to a local sinkhole that performs the 3-way TCP handshake with bots to detect their IP addresses [15]. Another DNS-based technique infers bots by observing lookup behavior into DNS blackhole lists (DNSBLs) [16], since botmasters tend to query these lists to inspect if their bots are blacklisted. Yet another technique detects botnets by identifying the features of a group activity in the DNS queries that a large number of bots in a botnet simultaneously send [17].

The last approach, spam-based one, is most relevant to our approach. Spam campaigns, i.e., coordinated mass emailing of spam, are used as the primary indicator to detect the membership in a single botnet [2]. In identifying the membership, 1) email messages are clustered into spam campaigns, using a number of fingerprints, 2) dynamic IP addresses are inferred into a single one of the same machine, using a probabilistic method, and 3) spam campaigns are merged into a single botnet to cope with the cases where a number of spam campaigns are initiated by the same botnet. These steps run on a cluster of 120 computers since the steps poses formidable computing challenge for a single computer. In [4], URLs embedded in email content are used to detect botnet-based spam emails and botnet membership. Then botnet hosts are identified by regular expressions of URL-based signatures to reduce the false positive rate against polymorphic URLs.

III. Our Method of Detecting Spamming Bots

This section first outlines our spam filter, next presents our strategy for clustering and describes our methods for clustering spams and detecting spamming bots and botnets.

A. Outline of our spam filter

Our spam filter is a cascade of three rule-based filters, each of which produces or collects information required for filtering [6]. Since the filter in a stage of the cascade sends the email it cannot classify to the next stage, the false (positive and negative) rates gradually decrease while the email passes through the cascade.

Let a *token* refer to the hash value of a 50-byte character string produced with the hash function described in [18], and the *fingerprint* (*FP*) of an email be a set of the all tokens for overlapping 50-byte character strings (sliding by one byte at a time) in the email's message body. Then the first filter in our cascade characterizes an incoming email by a set of 32 tokens, called *partial fingerprint* (*pFP*), that have the smallest least-significant 8 bits of the tokens in the FP for the email. The pFPs for earlier spams are stored in the filter and used to classify an incoming email, comparing its pFP with those saved in the filter.

The second filter classifies email, consulting the lists respectively of spam and legitimate email addresses in the *From* headers; this filter also checks the mail address format in the headers. The last filter classifies emails, using the lists respectively of spam-specific and legitimate email-specific words in the *Subject* headers. Evaluated with about 20,000 real world emails, our cascaded filters achieve the false negative rate of 0.025 with no false positive and is lightweight, i.e., classifies about 93 emails per second.

For a detected spam, our spam filter stores an entry consisting of the pFP and other items in a bucket of a hash table, called *eMail Hash Table* (*MHT*); an MHT bucket is a list of entries. The bucket including a specific spam is indexed by yet another fingerprint, the *summary FP* (*sFP*; also referred to as *origin index*), that equals $\sum_{i=0}^{31} \mathrm{token}_i$, where token_i is the i-th token in the spam's pFP.

Let the *dominant spam* for an incoming email stand for a spam whose pFP saved in the filter most matches the email's pFP and the number of matching tokens is not less than a threshold N_{dom}. To decrease the false rates, the filter puts a spam and its origin index into the MHT bucket with the dominant spam's origin index (referred to as *dominant index*) when the spam has the dominant spam, or stores only the spam in the bucket with its origin index otherwise.

B. Strategy for clustering

Let S denote the numbers of spams to be clustered, and T be the average number of tokens per FP. If we would perform spam clustering by FP, the clustering needs $S^2 T \log T$ comparisons, where S^2 is for comparing each spam with another one, and $T \log T$ is for comparing each token in the FP for a spam with all tokens in the FP for another spam, assuming the FPs organized into search trees.

The average size of a message body in our collected $S \simeq 2 \times 10^4$ emails is equal to about 4×10^3 bytes, so that $T \simeq 4 \times 10^3$. Then the total size of FPs equals about 800×10^6 bytes. Assuming one file I/O per 1,000 bytes and the I/O delay (millisecond order per disk seek) of 10^6 times greater that the CPU clock cycle time (nanosecond order), then the clustering of S spams directly by FPs in a single step would need $(S^2 T \log T)/10^3 \simeq 10^9$ seeks. This is equivalent to the time for about 10^{15} CPU clocks, and hence, about 10^6 seconds on a current home/office computer; note that one day equals about 8.6×10^4 seconds.

We reduce the computing power above through these strategy, assuming that in the bots clustering, we can use only the information collected in the spam filter:

1. Perform clustering in a number of steps.
2. Use different metrics in different clustering steps.
3. Produce clusters based on the similarity only of the *leaders* (i.e., the representative spams) of clusters.
4. Verify the clustering correctness in each step by the similarity of FPs.

Strategy 1 will reduce the computing power because the clustering is separately applied on each cluster produced in the previous step. Strategies 2 and 3 will decrease the computing power, with a penalty of misclustering. Strategy 4 compensates our clustering method for the probable misclustering due to Strategies 2 and 3.

C. Clustering of spam

For spam clustering, we modify our original filter so that the *Subject* headers and the first internal lines of the detected spams are also saved in the MHT entries. Moreover, we store the spams' fingerprints FPs in a number of files, call *FP files*. In addition, we copy all produced MHT entries into another hash table, called *Spam Cluster Table* (*SCT*), since the original filter deletes stale MHT entries every 1000 spams to save the memory resource.

We start our spam clustering with SCT since each SCT bucket is a preliminary cluster of spams since it is a list of spams with similar message bodies in the sense that they produce the same origin or dominant index. We define a pair of spams as being similar if the rate of the number of identical tokens in their FPs to the number of tokens in one of the FPs is greater than or equal to a threshold, $R_{\rm sim}$. Then our spam clustering proceeds as listed below and shown in Fig. 1; an SCT bucket is shown in Fig. 1.(a) and the *leader* of a cluster is the head spam in the cluster (see Fig. 1.(b)):

1. Cluster the spams in each SCT bucket by (partially) resolving the collisions (i.e., by further hashing the spams) due to the indexing by sFP (Fig. 1.(b)).
2. If a cluster C_d produced in step 1 is located in the bucket with the leader's dominant index d, merge cluster C_d with a cluster C_ℓ in the bucket with the leader's origin index ℓ if the leader is similar to some spam in cluster C_ℓ (see Fig. 1.(c)).
3. Merge the clusters generated in step 2 if their leaders have a number of common spam-specific words greater than or equal to a threshold, $N_{\rm swd}$.

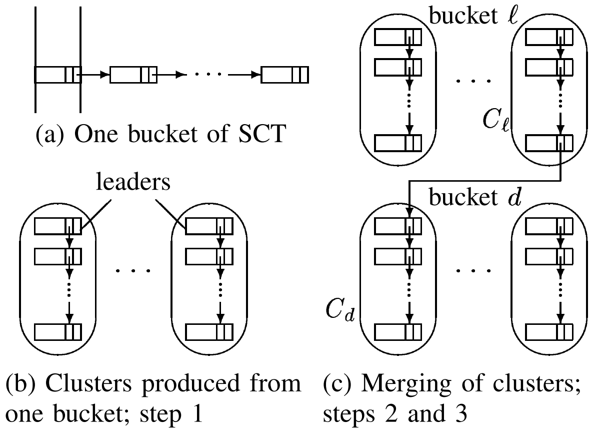

(a) One bucket of SCT
leaders
(b) Clusters produced from one bucket; step 1
bucket ℓ
C_ℓ
bucket d
C_d
(c) Merging of clusters; steps 2 and 3

Fig. 1. Clustering by removing aliases and merging based on dominant spams and spam-specific words.

To resolve the collisions in step 1, we put the spams with the same $\mathrm{XOR}_{i=0}^{31}\mathrm{token}_i$ into one cluster (XOR: eXclusive-OR). Then the spams in a obtained cluster have the same $\sum_{i=0}^{31} \mathrm{token}_i$ (= sFP) and the same $\mathrm{XOR}_{i=0}^{31}\mathrm{token}_i$.

A cluster C_d in bucket SCT[d] obtained in step 1 may consist only of the spams that have the same dominant index d but have the individual origin indices different from d (see Section A). Let ℓ be the original index of the leader of cluster C_d. Then we merge, in step 2, the cluster C_d with a cluster C_ℓ in bucket SCT[ℓ] if cluster C_ℓ includes at least one spam similar to the leader of cluster C_d (see Fig. 1.(c)).

In step 3, we merge two clusters generated in step 2 when their leaders' *Subject* headers have a number of common spam-specific words not less than a threshold $N_{\rm swd}$ and their leaders are similar to each other. Note that after each clustering in steps 1 to 3, we verify (see Strategy S4) using the FP files whether each spam in a cluster is similar to the cluster's leader. Thus the searching space for the clustering is greatly reduced, preserving the correctness of clustering if the threshold $R_{\rm sim}$ for similarity is appropriate.

D. Detection of spamming botnets

Major techniques for concealing botmaster's identity are to use a member bot of his (or her) botnet as an open proxy or an open mail relay [19]. Although restricting the usage of open email relay (since it is a misconfigured relay that exchanges information with any other computer on the Internet) is recommended in RFC2505 [20], the botmaster can instruct his bot to start a new open proxy or relay server so that he can send email via the open server. Then the *Received* headers in the received email contain the open server's network address, revealing nothing about the botmaster's identity. Moreover, the email received from the server (i.e., bot) then appears to come from a legitimate home or office user infected by the bot malware.

The first internal line gives the IP address of the outside machine that directly delivered the email across the Internet to the recipient's organization. Assuming that an incoming email originates from a botnet, then the first internal line includes the IP address of a member bot of the botnet, from the discussion above. Thus we identify the IP addresses of bots, using the first internal lines in SCT.

If two sets of IP addresses of the spamming bots in separate clusters have a common IP address, it is probable that the all bots in the clusters belong to the same botnet but the clusters are sending different spams, assuming a single bot malware infection per host. Then we can merge the clusters to obtain a larger size of botnet. As described above, it is not so easy to identify the botmasters' IP addresses due to the proxy and relay servers.

IV. Experiments

We experiment using the message sources of real-world emails received by the Mozilla mailer on the Linux system from Oct. 3, 2008 to Dec. 27, 2008 and from May 25, 2009 to Oct 16, 2009; the total number of collected emails reaches about 20k (6k and 14k, respectively). The emails are from U.S.A., E.U., Japan, and other countries and are written in English (about 90%), Japanese (about 5%), or other languages. Of the collected 20k emails, the legitimate email and spam are about 9% and 91%. We experiment on our workstation with dual-processor Xeon (2.66 GHz).

We have the three threshold values, N_{dom}, N_{swd}, and R_{sim}, for the clustering and its verification (see Section III). From the results of preliminary experiments, we set the thresholds thereafter so that $N_{\mathrm{dom}} = 6$, $N_{\mathrm{swd}} = 1$, and $R_{\mathrm{sim}} = 0.4$, since these values have produced largest clusters with no misclustering. To reduce the delay of disk I/O, the FPs of spams are stored in 63 files, in ascending order of email number (attached by person) and are accessed by selecting one of the files by the email number and sequentially searching in the selected file.

A. Spam clustering

To figure out the effects of clustering in each step, we evaluate three clustering methods respectively with only step 1 (denoted by S1), with steps 1 and 2 (S2), and with steps 1 to 3 (S3). The results are shown in Table 1. A smaller number of clusters mean a greater average cluster size. The average cluster size with method S1 is increased with S2, and is further increased with S3. The maximum cluster size is almost the same with the three methods. The execution time required for clustering increases in the order of S1 to S3, but S3 needs much larger time, 170,908 seconds (i.e., about 47 hours) than the other methods do.

To investigate the reason of the large execution time of S3, we experiment S3 with N_{swd} of 3; the results are listed in the parentheses of Table 1. When $N_{\mathrm{swd}} = 3$, the number of clusters slightly increases, and hence, the average size decreases, as compared with those when $N_{\mathrm{swd}} = 1$. However, the execution time greatly decreases to 12,659 seconds (about 3.5 hours). This is because the candidate clusters for merging with a cluster have to be verified before really merged, using the FPs in the files, and a smaller N_{swd} produces a larger number of candidates and hence need a greater verification time.

Table 1. Clustering characteristics and performance; results in the parentheses are obtained when $N_{\mathrm{swd}} = 3$.

	S1	S2	S3
Number of clusters	11,240	8,967	8,530 (8,814)
Average cluster size	1.3	1.6	1.7 (1.6)
Maximum cluster size	225	228	228 (228)
Execution time [sec]	2,251	4,095	170,908 (12,659)

Next, we measure the size distribution of the largest 30 (denoted by top-30) clusters; the results are shown in Fig. 2. The difference of size distributions with S3 and S2 is small, though the execution time with S3 is much greater than the time with S2 (see Table 1). Moreover, in the top-30 clusters produced with S2 and S3, no false positive and negative is found by inspection. Thus we can conclude that S2 is better than S3 in terms of performance/cost; we evaluate with method S2 in the rest of this section.

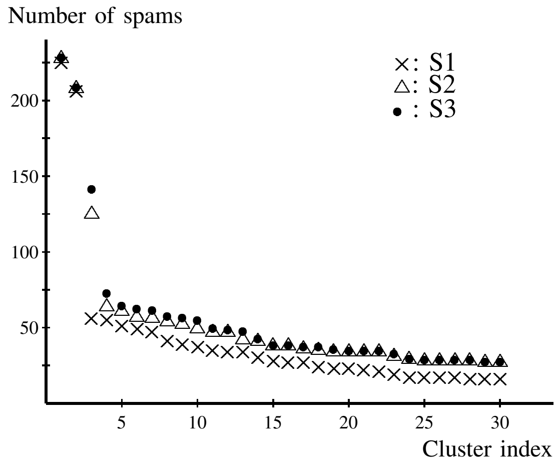


Fig. 2. The distribution of the number of spams in the top-30 clusters.

B. IP space of spamming bots

Next we plot the upper 8 bits of IP addresses (denoted by IPs/8) of spamming bots in the top-20 clusters; the result is shown in Fig. 3, where the dot symbol ($\cdot$) means that at least one spam in the cluster has the IP/8. The most active IPs/8 space consists of 57.*–97.*, 113.*–126.*, and 188.*–222.*; this result almost completely matches a previous result [8]. Surprisingly, 91% of the spams detected by our spam filter are from the active space. Unfortunately, subspaces 188.*–222.* and 113.*–126.* are expanding downward, and new active subspaces may be emerging around 160.* and below 40.*, as you can see in the figure.

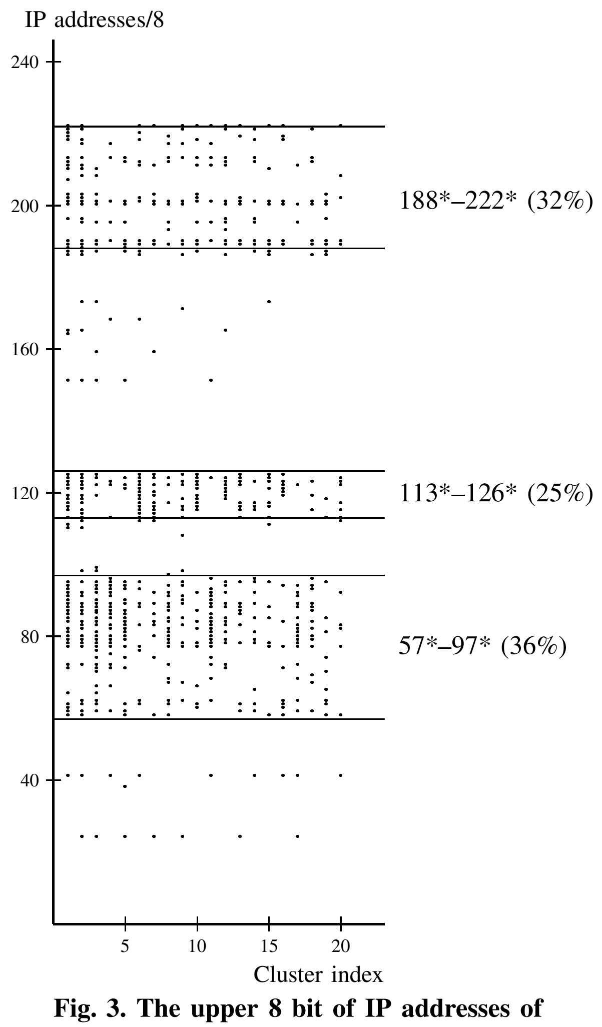


Fig. 3. The upper 8 bit of IP addresses of spamming bots in the top-20 clusters.

C. Botnets and their members

To investigate if the spamming bots in multiple clusters are controlled by the same botmaster, we inspect the IPs/16, IPs/24, and IPs/32 (respectively denoting the upper 16, 24, and 32 bits of IP addresses) that are common to the bots in the top-20 clusters. The result is shown in Fig. 4, where the symbols $\circ$, $\triangle$, and $\times$ respectively mean that at least one bot in a cluster has the same IP/16, the same IP/24, and the same IP/32 as those at least one bot in another cluster has. For instance, at least one bot in cluster 10 (denoted by C10) has the same IP/16 as that at least one bot in C1 has, and has the same IP/32 as that at least one bot in cluster C2 has. Note that we can consider Fig. 4 as an adjacency matrix for a graph, supposing that each cluster is a node of the graph and symbols $\circ$, $\triangle$, and $\times$ represent 1's in the matrix.

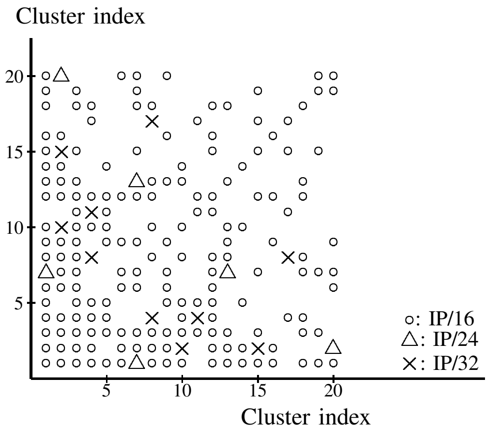


Fig. 4. Partial IP addresses (IPs/n) shared by the top-20 clusters; a symbol at (x,y) means that clusters x and y have at least one common IP/n.

When the bots in some clusters share at least one common IP/16, IP/24, or IP/32 (denoted by cIP/16, cIP/24, or cIP/32), we assume here that the bots in the clusters belong to the same botnet. Then inspecting based on cIP/16, the all bots in the top-20 clusters belong to a single botnet, because a transitive closure of nodes (i.e., clusters) labeled by the $\circ$ (i.e., IP/16) shown in Fig. 4 means that they belong to a single botnet. Likewise, based on cIP/24 ($\triangle$), the bots in clusters {C1,C7,C13}, the bots in {C2,C10,C15,C20}, and the bots in {C4,C8,C11,C17} belong to three different individual botnets. Moreover, based on cIP/32 ($\times$), the bots in {C2,C10,C15} and the bots in {C4,C8,C11,C17} are under the control of two botmasters, respectively.

The decision on botnet membership also depends on the dynamics of IP address. It is reported that more that 102 million dynamic IP addresses are identified in a month-long Hotmail user-login trace, and 97% of mail servers setup on dynamic IP addresses send out solely spam emails [21]. Under this situation, we may be able to figure out much larger sizes of botnets than described above. The detection of dynamic IP addresses is, however, out of scope of the paper, and we are reporting a method of botnets detection taking dynamic IP addresses into account in a future paper.

V. Conclusions

We have presented a lightweight method of detecting spamming bots and botnets, exploiting the information used in our spam filter, which provides us with a hash table of preliminary clusters each of a bucket of detected spams. Beginning with the hash table, we have clustered the spams through three steps, first by partially removing the hash collision in each bucket of the hash table, second by merging the obtained clusters based on the similarity of message bodies of the leaders (i.e., the representative spams in the clusters), and last by further merging the second-step clusters, using spam-specific words in the *Subject* headers of their leaders. A bot was identified by the IP address in the first internal line of the spams' *Received* headers, and a botnet was detected by merging the clusters based on the common (partial or total) IP addresses of their member bots.

By simulation with about 20,000 real-world emails, the clustering without step 3 is desirable in terms of computation cost and clustering potential. Then about 18,000 spams are clustered in about 4,000 seconds. The cluster sizes of the largest 30 clusters distribute from 27 to 228 spams, and the active IPs/8 (i.e., the upper 8 bits of the IP addresses) of the largest 20 clusters consists of 57*–97*, 113*–126*, and 188*–222*; some new subspaces appear to be emerging. The distribution of detected botnet sizes significantly differs, depending on the size (i.e., the upper 8, 16, 24, and 32 bits) of common IP addresses; we have to investigate the effect of dynamic IP address for more accurate sizes of botnets.

Our next work is a botnet detection system distributed worldwide and based on the lightweight detection of spamming bots presented in the paper: Suppose that we can identify spamming bots by spam filters running on home/office computers, and assume that the identifiers and related information for detected bots are sent to a server for bot/botnet analysis. Then we will be able to promptly and accurately figure out botnet structures.

References

[1] Messaging Anti-Abuse Working Group (MAAWG), "Email Metrics Program: The Network Operators' Perspective," *Report #12–Third and Fourth Quarter 2009*, MAAWG, 2010.

[2] L. Zhuang, J. Dunagan, D. R. Simon, H. J. Wang, I. Osipkov, G. Hulten, and J. D. Tygar, "Characterizing Botnets from Email Spam Records," *1st USENIX Workshop on Large-Scale Exploits and Emergent Threats (LEET)*, 2008.

[3] L. MaLaughlin, "Bot Software Spreads, Causes New Worries," *IEEE Distributed Systems Online*, Vol. 5, No. 6, 2004.

[4] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov, "Spamming Botnets: Signatures and Characteristics," *ACM SIGCOMM*, 2008.

[5] J. Goodman, "IP Address in Email Clients," *Conf. on Email and Anti-Spam (CEAS)*, 2004.

[6] M. Takesue, "Cascaded Simple Filters for Accurate and Lightweight Email-Spam Detection," *Intl. Conf. on Emerging Security Information, System and Technologies (SECURWARE)*, 2010.

[7] R. Xu and D. Wunsch II, "Survey of Clustering Algorithms," *IEEE Trans. on Neural Networks*, Vol. 16, No. 3, 2005.

[8] M. Kokkodis and M. Faloutsos, "Spamming Botnets: Are we losing the war?," *Conf. on Email and Anti-Spam (CEAS)*, 2009.

[9] A. Karasaridis, B. Rexroad, and D. Hoeflin, "Wide-scale Botnet Detection and Characterization," *1st Workshop on Hot Topics in Understanding Botnets (HotBot)*, 2007.

[10] G. Gu, J. Zhang, and W. Lee, "BotSniffer: Detecting Botnet Command and Control Channels in Network Traffic," *15th Annu. Conf. on Network and Distributed System Security (NDSS)*, 2008.

[11] Y. Al-Hammadi and U. Aickelin, "Detecting Botnets Through Log Correlation," *Workshop on Monitoring, Attack Detection and Migration (MonAM)*, 2006.

[12] E. Cooke, F. Jahanian, and D. McPherson, "The Zombie Roundup: Understanding, Detecting, and Disrupting Botnets," *Workshop on the Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, 2005.

[13] M. Akiyama, T. Kawamoto, M. Shimamura, T. Yokoyama, Y. Kadobayashi, and S. Yamaguchi, "A Proposal of Metrics for Botnet Detection Based on its Cooperative Behavior," *Int. Symp. on Applications and Internet Workshops (SAINTW)*, 2007.

[14] W. T. Strayer, R. Walsh, C. Livadas, and D. Lapsley, "Detecting Botnets with Tight Command and Control," *31st IEEE Conf. on Local Computer Networks*, 2006.

[15] D. Dagon, C. Zou, and W. Lee, "Modeling Botnet Propagation Using Time Zones," *13th Network and Distributed System Security Symp. (NDSS)*, 2005.

[16] A. Ramachandran, N. Feamster, and D. Dagon, "Revealing Botnet Membership Using DNSBL Counter-Intelligence," *2nd Conf. on the Steps to Reducing Unwanted Traffic on the Internet (SRUTI)*, 2006.

[17] H. Choi, Hanwoo Lee, Heejo Lee, and H. Kim, "Botnet Detection by Monitoring Group Activities in DNS Traffic," *7th IEEE Int. Conf. on Computer and Information Technologies*, 2007.

[18] U. Manber, "Finding Similar Files in a Large File System," *Winter USENIX Technical Conf.*, 1994.

[19] D. Boneh, "The Difficulties of Tracing Spam Email," *www.ftc.gov/reports/rewardsys/expertrpt_boneh.pdf*, 2004.

[20] G. Lingberg, "Anti-Spam Recommendations for SMTP MTAs," *RFC 2505*, 1999.

[21] Y. Xie, F. Yu, K. Achan, E. Gillum, M. Goldszmidt, and T. Wobber, "How Dynamic are IP Addresses?," *ACM SIGCOMM Conf.*, 2007.

Detecting & Defeating Split Personality Malware

Kalpa Vishnani, Alwyn R. Pais, Radhesh Mohandas
National Institute of Technology Karnataka, India
emails:kalpavishnani@gmail.com, alwyn.pais@gmail.com, radhesh@gmail.com

Abstract— **Security analysts extensively use virtual machines to analyse sample programs and study them to determine if they contain any malware. In the process, if the malware destabilizes the guest OS, they simply discard it and load in a fresh image. This approach increases their productivity. Since naive users do not run virtual machines, malware authors have observed that it is a pretty good probability that their malware is being analysed if it is being run in a Virtual Machine (VM). When these analysis aware malware detect the presence of VMs, they behave in a benign manner thus escaping detection. A determined analyst will have to end up running the sample on a native machine that adds to his chase time. In this paper, we briefly discuss the techniques deployed to detect VM by the Analysis Aware Malware also known as the Split Personality Malware. We then introduce our tool that not only detects this category of malware but also fools it into believing that it is running on a native machine even when it is running on a virtualized one, forcing it to exhibit its malicious form. Most security analysts should find this tool really useful.**

Keywords- Detecting Virtual Machines, Vmware, Analysis Aware Malware, Split Personality Malware, guest OS, host OS.

I. Introduction

Security researchers and analysts use a wide variety of tools to carry out malware analysis. Virtualization has emerged as a very useful technology in the field of security research and has gained widespread acceptance in the fraternity. It is very popular amongst malware researchers since they can intrepidly execute suspicious malware samples on the virtual machines without having their systems affected. Since many malware tend to destabilize the host systems, allowing them to run in a virtual environment increases the productivity of the analysts. This decreases the time and cost that the analysts need to study malware behaviours enabling them to build patches against the vulnerabilities that the malware exploit.

However, the malware developers have once again upped the ante by adding analysis awareness functionality into their malware. They detect the presence of malware analysis tools such as Virtual Machines (VM), debuggers and sandboxes and then either terminate execution or hide their malicious nature by executing like a benign application. As a result, they escape detection from a casual malware analyst. This category of malware is known as Analysis Aware malware or Split Personality malware.

The main subject of this paper is to tackle this class of malware. Current efforts mainly focus on flagging the Split Personality malware and once flagged they resort to analyzing them on a native machine to bring out their malicious nature. In this paper, we discuss our novel approach using which we detect the VM detection attempts and further trick the malware into believing that they are running on a host OS and hence make them exhibit their non-benign nature. We have developed a tool, VMDetectGuard for this purpose. We present the effective results obtained by means of this tool.

Our tool is currently built for VMware running the Windows platform. However, it is important to note that the solution we provide here is generic and can be easily tailored to cater to other OS platforms as well as VMs.

The remainder of this paper is organized as follows. Section 2 discusses the different VM detection techniques. In Section 3, we discuss related work and highlight their shortcomings. In Section 4, we present our approach for combating the VM-detecting Split Personality malware. In Section 5, we discuss the implementation details of our solution, VMDetectGuard. In Section 6, we present the analysis results obtained by running various VM detecting malware samples (both proof of concept and live malware) in the presence as well as in the absence of VMDetectGuard and noting down the behavioral changes in the malware. In Section 7, we conclude.

II. VM Detection Techniques

There are various ways of VM detection, all of which can be classified in one of the following categories:

A. Hardware Fingerprinting

Hardware Fingerprinting involves looking for specific virtualized hardware [1]. It can reveal a plethora of information about VM specific components required for reliable detection. In Table I, we have included the results of hardware fingerprinting which we obtained on a host OS and on a guest OS running on VMware. We carried out this fingerprinting using Windows Management Instrumentation (WMI) classes and APIs [2]

B. Registry Check

The registry entries contain hundreds of references to the string "VMware" in the guest OS. Checking the registry values for certain keys clearly reveals the VM presence [1]. The following are a few examples:

```
HKEY_LOCAL_MACHINE\HARDWARE\DEVICEMAP\Scsi\Scsi
Port1\Scsi Bus 0\Target Id 0\Logical Unit Id
0\Identifier
  → VMware, VMware Virtual S1.0
```

```
HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control
\Class\{4D36E968-E325-11CE-BFC1-
08002BE10318}\0000\DriverDesc
  → VMware SCSI Controller

HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control
\Class\{4D36E968-E325-11CE-BFC1-
08002BE10318}\0000\ProviderName
  → VMware, Inc.
```

C. *Memory Check*

This technique involves looking at the values of specific memory locations after the execution of instructions such as SIDT (Store Interrupt Descriptor Table), SLDT (Store Local Descriptor Table), SGDT (Store Global Descriptor Table), and STR (Store Task Register) [1][3]-[7]. It is the most widespread detection technique employed by the present day VM detecting malware.

D. *VM Communication Channel Check*

This check involves detecting the presence of a host-guest communication channel. The IN instruction is a privileged instruction which when executed from ring 3 of a protected mode OS such as Windows, raises the exception "EXCEPTION PRIV INSTRUCTION"' [1]. However, when it is running on VMware, no such exception is generated. Instead, VMware initiates guest to host communication by calling the 'IN' instruction. If the magic number ('VMXh') is returned to the register EBX, then it is certain that the program is running inside VMware.

E. *Timing Analysis*

An obvious yet rare attack against a Virtual Machine is to check a local time source, such as the "Time Stamp Counter" (TSC). We briefly restate the concept behind this attack discussed in a previous work [5].

Translation Lookaside Buffers (TLBs) can be explicitly flushed out and then the time to access a new page is determined by reading the TSC before and after the access. This duration can be averaged out over the number of TLBs to be filled. Next, the TLBs are filled with known data by accessing a set of present pages and the time to access a cashed page is determined as before. This value can also be averaged over the number of pages in the TLBs. Now, the CPUID instruction is executed. CPUID is the only VM sensitive instruction which on execution flushes out at least some of the TLBs as a side effect. Now each of the pages that were present in the VM is accessed again. If any of the page's access time matches that of a new page, the presence of a VM is revealed!

F. *Process & File Check*

There are many VMware specific processes such as VMwareUser.exe, vmacthlp.exe, VMwareService.exe, VMwareTray.exe that constantly run in the background. There also exist some VMware specific files and folders [1]. Hence querying for these objects could also serve as a method for VM detection. Though this method could easily be fooled, when combined with other detection techniques, it could obtain more reliable results.

III. Related Work

We found that the amount work done for the containment of Split Personality malware is not substantial. Very few researchers have provided solutions to counter the same. Moreover, most of them have focussed only on the detection of this class of malware [8][9][10]. Once this class of malware is detected, they propose to further analyse these malware on a host OS. The only approaches we found that aim at tricking the malware are proposed by Carpenter et al. [11] and Guizani et al [12].

Zhu & Chin [9] discuss two approaches to counter VM-aware malware. One approach professes the use of dynamic analysis to identify known virtual machine detection techniques. The authors have built an implementation for it called "Malaware". This is a mere detection approach. Once the malware is detected it is to be further analysed on a native machine. In the second method they propose the use of dynamic taint tracking to detect any impact caused by the input that changes the execution path of the malware. Although the authors claim that this approach will help to detect the already unknown VM detection techniques, we beg to differ. In this second approach they have addressed only two of the various VM detection techniques, Memory Check and Registry Check. Out of these, for countering Registry Check detection method they propose that, a check should be made to determine if the sample contains any conditional jump statement following a registry query. If so, they conclude that the sample is probably taking another execution path because it detected the presence of virtual machine. We further argue that this is not a good heuristic as a sample will not always be a split personality malware if it has a conditional statement after a registry access. Even a legitimate application could do that for other genuine reasons. For instance, the commercial software with trial periods have to extensively make use of this logic in order to check the registry values to see if the software has been registered by the user or not. If not then it must run in the trial mode. Moreover, this method does not give any solutions for other types of VM detection techniques such as Hardware Fingerprinting and Timing Analysis, both of which are gradually being adopted by the advanced Split Personality malware.

Carpenter et al. propose [11] two mitigation techniques. They aim at tricking the malware by, 1) changing the configuration settings of the .vmx file present on the host system and, 2) altering the magic value to break the guest-host communication channel. Out of these two techniques the first one has the following setbacks:

- The configuration options break the communication channel between guest and host not just for the program trying to detect the VM, but for all the programs.

- Moreover the authors claim that these are undocumented features and that they are not aware of any side effects.

Their second technique is targeted only against VM Communication Channel Check method.

The work by Guizani et al. [12] provides an effective solution for Server-Side Dynamic Code Analysis. A small part of their solution deals with tricking the Split Personality malware employing Memory Check and VM Communication Channel detection techniques. However they do not address other detection techniques. It was their work that inspired us to build a complete solution for the containment of the all the VM detection methods and provide a more complete and robust solution.

The approach mentioned in the work by Balzarotti et al. [10] involves first running the sample on the reference system (physical system), logging its input and output values exchanged with the system and then running the same sample on the analysis system which runs a virtual environment where the output values, that were obtained on the reference system are simply replayed. Then, the differences in the sample's behaviour are observed. Thus, in this work too, the entire analysis of the detected Split Personality Malware is not carried out in the virtualized environment.

Hence we conclude that there does not exist any complete solution that effectively counters Split Personality malware.

IV. OUR APPROACH

The main objective of this paper is to carry out the analysis, detection and containment of the Split Personality malware entirely on the virtualized system. We perform dynamic binary instrumentation of the sample under test in order to obtain its low level information as well as to intercept all the API calls made by it. We then check to see if the sample is trying to access any information which would help it in determining the VM presence. If a match is found with any of our monitored set of API calls or low level instructions, our tool logs the activity and provides fake values to the sample so as to make it feel that it is running on the native system. Fig. 1 illustrates the approach step by step.

Step 1: *Maintain a list of all the hardware as well as registry querying API calls. Also maintain a list of all the VM specific instructions such as SIDT, SLDT, SGDT, STR, IN.*

Following is a partial list of API calls to be monitored.

a) Hardware Querying APIs
   i) SetupDiEnumDeviceInfo()
   ii) SetupDiGetDeviceInstanceId()
   iii) SetupDiGetDeviceRegistryProperty()
   iv) WMI APIs

b) Registry Querying APIs
   i) RegEnumKey()
   ii) RegEnumValue()
   iii) RegOpenKey()
   iv) RegQueryInfoKeyValue()
   v) RegQueryMultipleValues()
   vi) RegQueryValue()

Step 2: *Perform dynamic binary instrumentation of the sample under test in order to obtain its low level information as well as to intercept all the API calls made by it.*

We perform dynamic binary instrumentation of the sample using the Pin framework [13]. It allows for monitoring all the API calls and low level instructions being executed by the sample.

Step 3-12: *Check to see if the sample under test makes a call or executes any of the monitored API calls or instructions respectively. If a match is found, set the OUTPUT to "Split Personality Malware Detected". Also, log the activity and provide fake values to the sample so as to make it feel that it is running on a host system.*

Let us consider the example of a sample that makes the following API calls with the given arguments:

```
RegOpenKeyEx(
       HKEY_LOCAL_MACHINE,
       TEXT("HARDWARE\\DEVICEMAP\\Scsi\\Sc
       si Port 0\\Scsi Bus 0\\Target Id
       0\\Logical Unit Id 0"),
       0,
       KEY_QUERY_VALUE,
        &hKey);

RegQueryValueEx(
       hKey,
        TEXT("Identifier"),
        NULL,
        NULL,
        (LPBYTE) PerfData,
        &cbData );
```

In the above case, the key value returned in a VMware machine will contain the string "VMware". Thus, we monitor the values returned by the OS in response to the API calls made by the sample. If it contains the string "VMware", the control passes to our replacement routine where we change the value to a more appropriate value such as "Miscrosoft" or to a value that would have been returned on a host Windows OS.

Similarly when VM specific instructions such as SIDT are at the verge of being executed by the sample, the control passes to our replacement routine where we set the value of the destination operand to a value that would be obtained on the host Windows OS.

```
Algorithm 1 VMDetectGuard to detect and trick Split Personality Malware
Input: Malware sample to be tested.
Output: Boolean Result: OUTPUT indicating Split Personality malware detected/not detected.

 1: Maintain a list of API calls and low level instructions that help in VM Detection.
 2: Run the sample under test.
 3: Hook into the sample.
 4: Set OUTPUT to NULL.
 5: while the sample executes do
 6:   Intercept the API calls and low level instructions being executed by the sample.
 7:   if match is found with the monitored set of API calls or low level instructions. then
 8:     Log the activity.
 9:     Set the OUTPUT to Split Personality malware detected.
10:     Provide fake values to the running sample.
11:   else
12:     Do nothing.
13:   end if
14: end while
```

Figure 1. Our Approach for Countering Split Personality Malware

V. Implementation

We have designed and implemented a solution to counter Split Personality malware that employ the various VM detecting techniques. In this section we present a detailed discussion of our implementation, VMDetectGuard.

We implemented our solution in the framework provided by the Pin tool [13] released by Intel Corporation. Pin is a tool for the instrumentation of programs. Pin allows a tool (such as ours) to insert arbitrary code in arbitrary places in the executable. The code is added dynamically while the executable is running.

A. Methodology

As we stated earlier, all the VM detection methods fall under one or more categories of VM detection discussed in Section 2, we present our implementation methodology with respect to each VM detection category.

i) Countering Hardware Fingerprinting

We propose hardware emulation. The idea is to maintain a list of all the API calls that provide hardware information such as BIOS, Motherboard, Processor, Network Adapter etc. such that even if false values are supplied about them to a ring 3 application querying such information, the application would not crash. We created a proof of concept program to carry out hardware fingerprinting of a native as well as a virtual machine. Table I summarizes the results.

VMDetectGuard hooks into the sample under analysis and monitors the API calls it makes. Whenever a match is found with our set of monitored API calls, it logs this activity and provides fake values to the sample. In Table I we see how VM returns a value “none” for motherboard serial number. VMDetectGuard returns a more appropriate string such as “.16LV3BS.CN70166983G1XF” instead. Each time a match with the monitored set of APIs is found, our tool empowers the analyst who can choose either to modify or not to modify the values being returned to the sample, thus enabling him to notice any changes in the sample's behaviour.

Caveat: There are certain hardware components that cannot be emulated. For instance, the MAC address cannot be faked because the program requesting this value would be unable to carry out the desired networking tasks. In this case, we urge the malware analysts to change their MAC address in their VMware machine so that it does not match the VMware MAC address pattern. The guidelines for this are provided on the VMware forums [14].

ii) Countering Registry Check

VMDetectGuard also monitors registry querying APIs such as RegQueryInfoKeyValue, RegOpenKey, etc. It intercepts these API calls whenever they are executed by the sample. It then looks at the output values returned by the system. If the output contains the string "VMware", our tool replaces this string with a value that would have been returned on a non virtual system running the same OS.

iii) Countering Memory Check

For countering memory check we detect the presence SIDT, SLDT, and SGDT and STR instructions.

VMDetectGuard logs the activity whenever any of the above instructions is at the verge of being executed by the malware sample. It also appropriately modifies the values of the registers that are affected by these instructions after their execution making the sample feel that it is running on a native system.

Table II shows the different values obtained on a VMware and a host machine respectively on executing the above mentioned instructions.

TABLE II. Values obtained on executing Memory Check instructions on VMware and Host Machine (Windows)

Instruction	VMware	Host machine
SIDT	IDT is located typically at 0xffXXXXXX	IDT is located at a location lower than that. Around 0x80ffffff.
SLDT	Not located at 0xdead0000	Located at 0xdead0000
SGDT	GDT is located typically at 0xffXXXXXX	GDT is located at a location lower than that around 0x80ffffff.
STR	Selector segment value of TR register is value other than 0x40000000	Selector segment value of TR register is 0x40000000

iv) Countering VM Communication Channel Check

We address this check in a way similar to countering Memory Check. We monitor execution of the IN instruction, and change the value of the magic number

('VMXh') that was supplied as an input parameter by the sample under test to some other value.

v) Countering Timing Analysis

Our tool monitors the sample for instructions such as CPUID and RDTSC (Read Time Stamp Counter). Moreover it maintains the count of each type of instruction executed. So if a particular instruction is executed a large number of times which is above the threshold value for that type of instruction, it logs this activity too. This is because timing attacks are known to execute a single or a couple of instructions for a very large number of times as certain instructions when run for a large number of times on a virtualized system take considerably longer than on a native machine to execute. Such attacks also make use of the CPUID instruction. We counter this detection method by deleting the CPUID instruction just before its execution and then modifying the values of the general purpose registers that are affected by the CPUID instruction (ebx, ecx, edx).

vi) Countering File & Process Check

These checks are countered in the similar way as the Registry Check. APIs for File/Folder/Process queries are monitored. If the sample makes querying request for VMware files, folders or processes, the tool sends out the 'file/process not found' error.

Thus our tool takes complete control of the sample and governs the output values to be fed to it.

B. VMDetectGuard Output

VMDetectGuard produces various log files along with the Boolean Result: Split Personality malware detected/ not detected.

It generates instruction trace, system call trace, instruction count log, opcode mix log as well as a VM specific log. This VM specific log contains all the API calls as well as the low level instructions that were executed by the sample under test. In case the sample is not a Split Personality malware, the VM specific log remains empty. All these logs can be used for further analysis of the sample.

VI. Results & Analysis

In order to test the effectiveness of our tool VMdetectGuard, we ran various VM detecting malware samples (both, proof of concept samples and live malware captured from the internet) on VMware in the presence as well as absence of VMDetectGuard; to observe if there were any notable changes in their behaviour. Table III summarizes the results of our analysis.

Fig. 2 and Fig. 3 illustrate the changes in the behaviours of redpill.exe [6] and scoopyNG.exe [7] respectively when ran on VMware in the presence and absence of VMDetectGuard respectively. It can be seen how VMDetectGuard fools both the binaries into believing that they are not running on a Virtual Machine.

We also analysed some samples of live malware captured from the internet. Amongst these, Backdoor.Win32.SdBot.fmn was found to employ both, Timing Analysis as well as Memory Check. When run in the absence of VMDetectGuard, the application displays a message, "Sorry, this application cannot run in a Virtual Machine". However on running it in the presence of VMDetectGuard, it runs and ultimately shuts the instance of OS running on VMware! While analyzing the logs generated by this malware sample we noted that it executed RDTSC 487 times, CPUID once. It also executed SIDT and SLDT instructions. But since our tool provided it with fake values it continued to act malicious and ultimately shut the OS. By means of VMDetectGuard, we also obtained its low level trace as well as system call trace for further analysis. Fig. 4 shows how Backdoor.Win32.SdBot.fmn refuses to run in a virtual machine when run in the absence of VMDetectGuard. Fig. 5 is a snapshot of the low level information of Backdoor.Win32.SdBot.fmn obtained while tricking it using VMDetectGuard. It shows the use of the Memory Check method (SLDT instruction) made by the malware sample.

VII. Conclusion

Split Personality malware is on a gradual rise and proactive measures are necessary to curb them before they become uncontrollable.

We found lack of research in this field. Moreover there does not exist any full-fledged tool to counter Split Personality malware.

We have designed and implemented VMDetectGuard, a tool that detects as well as tricks Split Personality malware. Our experimental results demonstrate that the tool effectively detects as well as tricks the split personality binaries leading to their effective analysis in the virtualized environment.

Although we have tested VMDetectGuard for several VM Detecting malware, we are still in the testing phase to ensure the completeness of our solution. Moreover, we are yet to carry out its performance evaluation to make it more efficient. We are working on it.

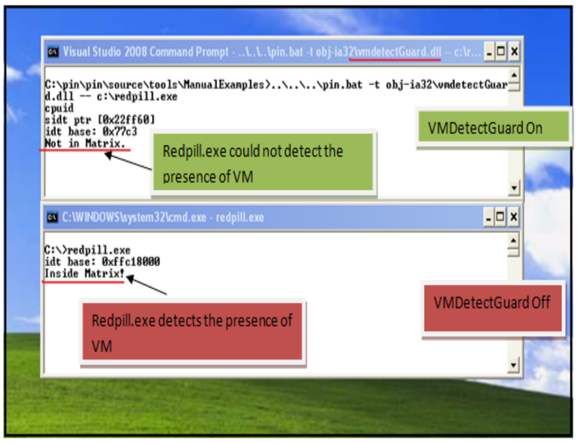
Our solution is currently built for VMware and Windows OS. We now seek to extend the support to other Operating Systems as well as Virtual Machines such as VirtualBox, Virtual PC, Xen, Hydra, Qemu etc. Similar techniques can also be used to counter anti-debugging tricks.

TABLE I COMPARISON RESULTS OF HARDWARE FINGERPRINTING OBTAINED ON A WINDOWS VIRTUAL AND A NATIVE MACHINE RESPECTIVELY

Hardware component	**Attribute queried**	**VMware**	**Native Machine**
Motherboard	Serial No.	None	.2GTP3BS.CN7016697MG1DN.
Processor	SocketDesignation	CPU Socket #0	Microprocessor
SCSI Controller	Caption	VMware SCSI Controller	Microsoft iSCSI Initiator
BIOS	Serial Number	VMware-56 4d 68 4c f9 e5 62 f4-fb 4d f0 5b 88 28 29 d9	2GTP3BS
USB Controller	Caption	1. Intel(R) 82371AB/EB PCI to USB Universal Host Controller 2. Standard Enhanced PCI to USB Host Controller	1. Intel(R) ICH9 Family USB Universal Host Controller – 2936 2. Intel(R) ICH9 Family USB Universal Host Controller – 2938 3. Intel(R) ICH9 Family USB Universal Host Controller – 2937
Network Adapter	Caption	1. VMware Accelerated AMD PCNet Adapter	1. WAN Miniport (SSTP) 2. WAN Miniport (IKEv2) 3. WAN Miniport (L2TP)
Network Adapter	Mac Address	00:0C:29:28:29:D9 (This MAC address falls in VMWare Mac Address Range)	50:50:54:50:30:30

TABLE III SPLIT PERSONALITY MALWARE ANALYSIS RESULTS OBTAINED USING VMDETECTGUARD

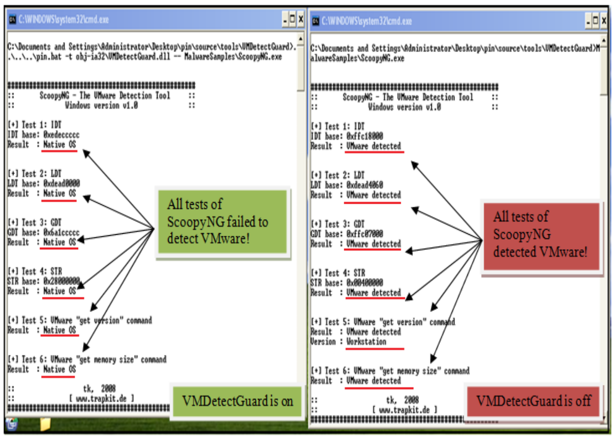
No.	**VM detecting program sample**	**VM detection method employed**	**VMware run with VMDetectGuard turned off**	**VMware run with VMDetectGuard turned on**
1	RedPill [6]	Memory	Detected VMware	Could not Detect VMware
2	ScoopyNG [7]	Memory	Detected VMware	Could not Detect VMware
3	VmDetect [15]	Memory	Detected VMware	Could not detect VMware
4	Worm.win32.autorun.pga	Timing Analysis	Displayed message saying "not a valid win32 application"	Ran maliciously
5	Trojan-Spy.Banker.pcu	Memory	Immediately terminated execution	Ran maliciously
6	Trojan-Spy.Win32.Bancos.zm	Memory	Ran benignly	Ran maliciously
7	Backdoor.Win32.SdBot.fmf	Memory	Ran benignly	Ran maliciously
8	Backdoor.Win32.SdBot.fmn	Memory, Timing Analysis	Displays a message, "This application cannot run under a Virtual Machine"	Ran maliciously


Visual Studio 2008 Command Prompt - ..\..\..\pin.bat -t obj-ia32\vmdetectGuard.dll -- c:\r...
C:\WINDOWS\system32\cmd.exe - redpill.exe
VMDetectGuard On
Redpill.exe could not detect the presence of VM
VMDetectGuard Off
Redpill.exe detects the presence of VM

```
C:\pin\pin\source\tools\ManualExamples>..\..\..\pin.bat -t obj-ia32\vmdetectGuar
d.dll -- c:\redpill.exe
cpuid
sidt ptr [0x22ff60]
idt base: 0x77c3
Not in Matrix.
```

```
C:\>redpill.exe
idt base: 0xffc18000
Inside Matrix!
```

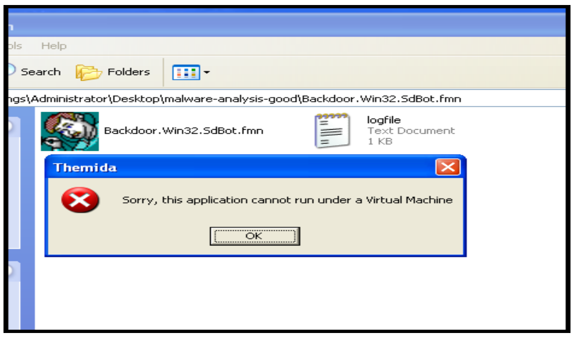
Figure 2. Redpill.exe executed in the presence and absence of VMDetectGuard resp.


C:\WINDOWS\system32\cmd.exe
All tests of ScoopyNG failed to detect VMware!
VMDetectGuard is on
C:\WINDOWS\system32\cmd.exe
All tests of ScoopyNG detected VMware!
VMDetectGuard is off

```
C:\Documents and Settings\Administrator\Desktop\pin\source\tools\VMDetectGuard>.
.\..\..\pin.bat -t obj-ia32\VMDetectGuard.dll -- MalwareSamples\ScoopyNG.exe


####################################################
::       ScoopyNG - The VMware Detection Tool       ::
::              Windows version v1.0                ::

[+] Test 1: IDT
IDT base: 0xedeccccc
Result  : Native OS

[+] Test 2: LDT
LDT base: 0xdead0000
Result  : Native OS

[+] Test 3: GDT
GDT base: 0x6a1ccccc
Result  : Native OS

[+] Test 4: STR
STR base: 0x28000000
Result  : Native OS

[+] Test 5: VMware "get version" command
Result  : Native OS

[+] Test 6: VMware "get memory size" command
Result  : Native OS

::                   tk,  2008
::               [ www.trapkit.de ]
####################################################
```

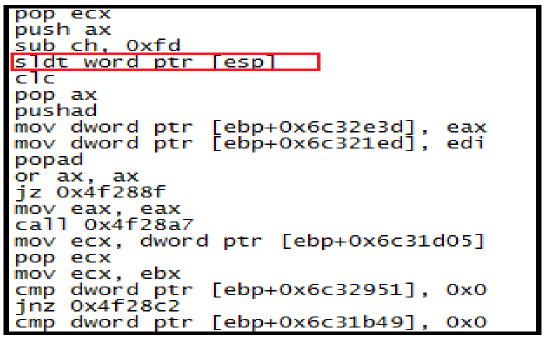
```
C:\Documents and Settings\Administrator\Desktop\pin\source\tools\VMDetectGuard>M
alwareSamples\ScoopyNG.exe


####################################################
::       ScoopyNG - The VMware Detection Tool       ::
::              Windows version v1.0                ::

[+] Test 1: IDT
IDT base: 0xffc18000
Result  : VMware detected

[+] Test 2: LDT
LDT base: 0xdead4060
Result  : VMware detected

[+] Test 3: GDT
GDT base: 0xffc07000
Result  : VMware detected

[+] Test 4: STR
STR base: 0x00400000
Result  : VMware detected

[+] Test 5: VMware "get version" command
Result  : VMware detected
Version : Workstation

[+] Test 6: VMware "get memory size" command
Result  : VMware detected

::                   tk,  2008
::               [ www.trapkit.de ]
####################################################
```

Figure 3. ScoopyNG.exe executed in the presence and absence of VMDetectGuard resp.



Sorry, this application cannot run under a Virtual Machine

Figure 4. Backdoor.Win32.SdBot.fmn run in the absence of VMDetectGuard



```
pop ecx
push ax
sub ch, 0xfd
sldt word ptr [esp]
clc
pop ax
pushad
mov dword ptr [ebp+0x6c32e3d], eax
mov dword ptr [ebp+0x6c321ed], edi
popad
or ax, ax
jz 0x4f288f
mov eax, eax
call 0x4f28a7
mov ecx, dword ptr [ebp+0x6c31d05]
pop ecx
mov ecx, ebx
cmp dword ptr [ebp+0x6c32951], 0x0
jnz 0x4f28c2
cmp dword ptr [ebp+0x6c31b49], 0x0
```

Figure 5. Low level information of ackdoor.Win32.SdBot.fmn obtained while tricking it using VMDetectGuard

REFERENCES

[1] Liston T. and Skoudis E. (2006). “On the Cutting Edge: Thwarting Virtual Machine Detec-tion” [Online]. Available: http://handlers.sans.org/tliston/ThwartingVMDetection_Liston_Skoudis.pdf (Nov 1, 2010)

[2] WMI Classes (2008). “WMI Classes Windows” [Online]. Available: http://msdn.microsoft.com/en-us/library/aa394554%28v=vs.85%29.aspx (Dec 30, 2010)

[3] Quist D. and Smith V. (2005). “Detecting the Presence of Virtual Machines Using the Local Data Table” [Online] Available: http://www.offensivecomputing.net/files/active/0/vm.pdf (Nov 14, 2010)

[4] Omella A. (2006). “Methods for Virtual Machine Detection” [Online]. Available: http://www.s21sec.com (Nov 24, 2010)

[5] Ferrie P. “Attacks on Virtual Machines”. In the Proceedings of the Association of Anti-Virus Asia Researcher Conference, 2007.

[6] Rutkowska J. (2004). “Red Pill” [Online]. Available: http://invisiblethings.org/papers/redpill.html (Nov 4, 2010)

[7] Klein T. (2005). “Scooby Doo - VMware Fingerprint suite” [Online]. Available: http://www.trapkit.de/research/vmm/scoopydoo/index.html (Nov 20, 2010)

[8] Lau B. and Svajcer V. “Measuring virtual machine detection in malware using DSD tracer”. In the Proceedings of Virus Bulletin, 2008, pp. 181-195.

[9] Zhu D. and Chin E. (2007). “Detection of VM-Aware Malware” [Online]. Available: http://radlab.cs.berkeley.edu/w/uploads/3/3d/Detecting_VM_Aware_Malware.pdf (Dec 10, 2010)

[10] Balzarotti D., Cova M., Karlberger C., Kruegel C., Kirda E., and Vigna G. “Efficient Detection of Split Personalities in Malware”. In the Proceedings of 17th Annual Network and Distributed System Security Symposium (NDSS 2010), Feb 2010

[11] Carpenter M., Liston T., and Skoudis E. "Hiding Virtualization from Attackers and Malware". IEEE Security and Privacy, June 2007, pp. 62-65.

[12] Guizani, W., Marion J.-Y., and Reynaud-Plantey D. “Server-Side Dynamic Code Analysis”. Analysis, 2009

[13] Pin (2004). “Pin - A Dynamic Binary Instrumentation Tool” [Online]. Available: http://www.pintool.org/ (Jan 10, 2010)

[14] VMware (2010), “VMware KB: Changing a MAC address in a Windows virtual machine” [Online]. Available:

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1008473

(Jan 15, 2010)

[15] VmDetect (2005), “Detect if your program is running inside a Virtual Machine - CodeProject” [Online]. Available: http://www.codeproject.com/KB/system/VmDetect.aspx

(Jan 4, 2010)

Proposal of n-gram Based Algorithm for Malware Classification

Abdurrahman Pektaş
The Scientific and Technological Research Council of Turkey
National Research Institute of Electronics and Cryptology
Gebze, Turkey
e-mail:apektas@uekae.tubitak.gov.tr

Mehmet Eriş
The Scientific and Technological Research Council of Turkey
National Research Institute of Electronics and Cryptology
Gebze, Turkey
e-mail:eris@uekae.tubitak.gov.tr

Tankut Acarman
Computer Engineering Dept.
Galatasaray University
Istanbul, Turkey
e-mail:tacarman@gsu.edu.tr

***Abstract*— Obfuscation techniques degrade the n-gram features of binary form of the malware. In this study, methodology to classify malware instances by using n-gram features of its disassembled code is presented. The presented statistical method uses the n-gram features of the malware to classify its instance with respect to their families. n-gram is a fixed size sliding window of byte array, where n is the size of the window. The contribution of the presented method is capability of using only one vector to represent malware subfamily which is called subfamily centroid. Using only one vector for classification simply reduces the dimension of the n-gram space. Experimental results are performed over a fairly large data set, which is being collected through Computer Emergency Response Team (CERT) activities in the National Research Institute of Electronics and Cryptology, to illustrate the effectiveness of the proposed malware classification methodology.**

Keywords-* *malware; n-gram based; classification

I. INTRODUCTION

The basic definition of malware (malicious software) may be presented as follows: piece of software code that works for the attacker. Malware has great popularity amongst cyber criminals since it offers attractive income opportunities. This popularity makes the malware an important threat for the computing society.

The presented classification approach uses the centroid of the subfamily which is constructed from its samples. Therefore unknown malware classification can be achieved by using low dimension centroid vector which requires less computational work. Experimental study is performed to validate the accuracy of the presented centroid-based approach. Used data set is constituted by the national activities of CERT Coordination Center, which is the national consultation center for computer security incidents [1].

The representation of malware by using n-gram profiles has been presented in the open literature, see for example [2], [3] and [4]. In these studies some promising results towards malware detection are presented. However malware domain has been evolving due to survivability requirements. Malware has to evade anti-virus scanners to perform its functions. Obfuscation techniques have been developed in order to avoid detection by anti-virus scanner. And these techniques disturb n-gram features of binary form of the malware used by the previous work. Similar methodologies have been used in source authorship, information retrieval and natural language processing [5], [6].

The first known use of machine learning in malware detection is presented by the work of Tesauro *et al.* in [7]. This detection algorithm was successfully implemented in IBM's antivirus scanner. They used 3-grams as a feature set and neural networks as a classification model. When the 3-grams parameter is selected, the number of all n-gram features becomes 256^3, which leads to some spacing complexities. Features are eliminated in three steps: first 3-grams in seen viral boot sectors are sampled, then the features found in legitimate boot sectors are eliminated, and finally features are eliminated such that each viral boot sectors contained at least four features. Size of feature vectors in n-grams based detection models becomes very large so feature elimination is very important in these models. The presented work has been limited by the boot sector viruses' detection because boot sectors are only 512 bytes and performance of technique is degraded significantly for larger size files.

As a historical track, IBM T.J. Watson lab extended boot virus sector study to win32 viruses in 2000 [8]. At this stage, 3 and 4 grams were selected and encrypted data portions within both clean files and viral parts were excluded due to the fact that encryption may lead to random byte sequences. At the first instance, n-grams existed in constant viral parts were selected as features and then, the ones existed in clean files more than a given threshold value were removed from the feature list. In this study, along the use of neural networks boosting was also performed. Results of this study shown that the developed method performance was not sufficient. Schultz *et al.* has used machine learning methods in [9]. Function calls, strings and byte sequence were used as the feature sets. Several machine learning methods such as RIPPER, Naive Bayes and Multi Naive Bayes were applied, the highest accuracy of 97.6% with Multi Naive Bayes was achieved.

Abou-Assaleh *et al.* [3] contributed to the ongoing research while using common n-gram profiles. k nearest neighbor algorithm with k=1 instead of the other learners was used. Feature set was constituted by using the n-grams and the occurrence frequency, where the occurrence frequency is denoted by L. Tests have been done with different *n* (ranging from 1 to 10) and *L* (ranging from 20 to 5000) values. Data set used in these experiments was kept fairly conservative of 25 malware and 40 benign files. With this set, test results shown 98% of success. Using the data in [3], the accuracy slightly dropped to the 94% level.

Kolter *et al.* [2] used 4-grams as features and selected top 500 n-grams through information gain measure. They used instance based learners, TFIDF, naive bayes, support vector machines, and decision trees and also boosted last three learners. Boosted decision tree outperformed all others and gave promising results such as ROC curve of 0.996.

While the battle between malware authors and anti-virus producers are continuing, our motivation is to find the statistical method to classify the malware instance by using n-gram features (profiles) of disassembled malware. In our methodology, we use n-gram feature of the malware to classify the malware instance with respect to their family. n-gram is a fixed size sliding window of byte array, where n is the size of the window. For example the “81EDD871” sequence is segmented (represented) into 5-gram as “81EDD”, “1EDD8”, “EDD87” and “DD871”.

This paper is organized as follows: Section 2 proposes the methodology. Section 3 elaborates and computes the accuracy of the proposed methodology. Finally, concluding remarks and future works are presented in Section 4.

II. System Design

As stated in the introduction, current malware samples cannot be analyzed easily based on their statistical features’ as in the previous decade because of the increasing use of the obfuscation techniques by the malware authors.

The proposed algorithm consists of preprocessing, training and testing phase. Malware samples are collected through TR-CERT [1] activities in The National Research Institute of Electronics and Cryptology. We classified our dataset by using Microsoft Security Essential (MSE) antivirus tool [17]. In other words, naming of the malware instance is performed by the MSE tool. Malware naming is not a well standardized area where all vendors, players can name and classify malware according to their intentions, and common sense in naming cannot be achieved among the stakeholders [16]. After that preprocessing step, PEid as a useful tool to inspect PE files, is used to dissemble malware instances [18]. We extract a malware instance’s n-gram profile through opcode sequences obtained from PEid. We are using opcode sequences instead of byte sequences of the malware.

In our study, machine codes to extract malwares’ n-gram profile instead of byte sequences are considered and the n-gram feature space is considerably reduced. In this manner calculations are performed faster and efficiently. Each malware sample is used to determine its subfamily vector which is named as the centroid of the subfamily.

Family of the malware is a descriptor of the malware used to classify malware samples according to their features

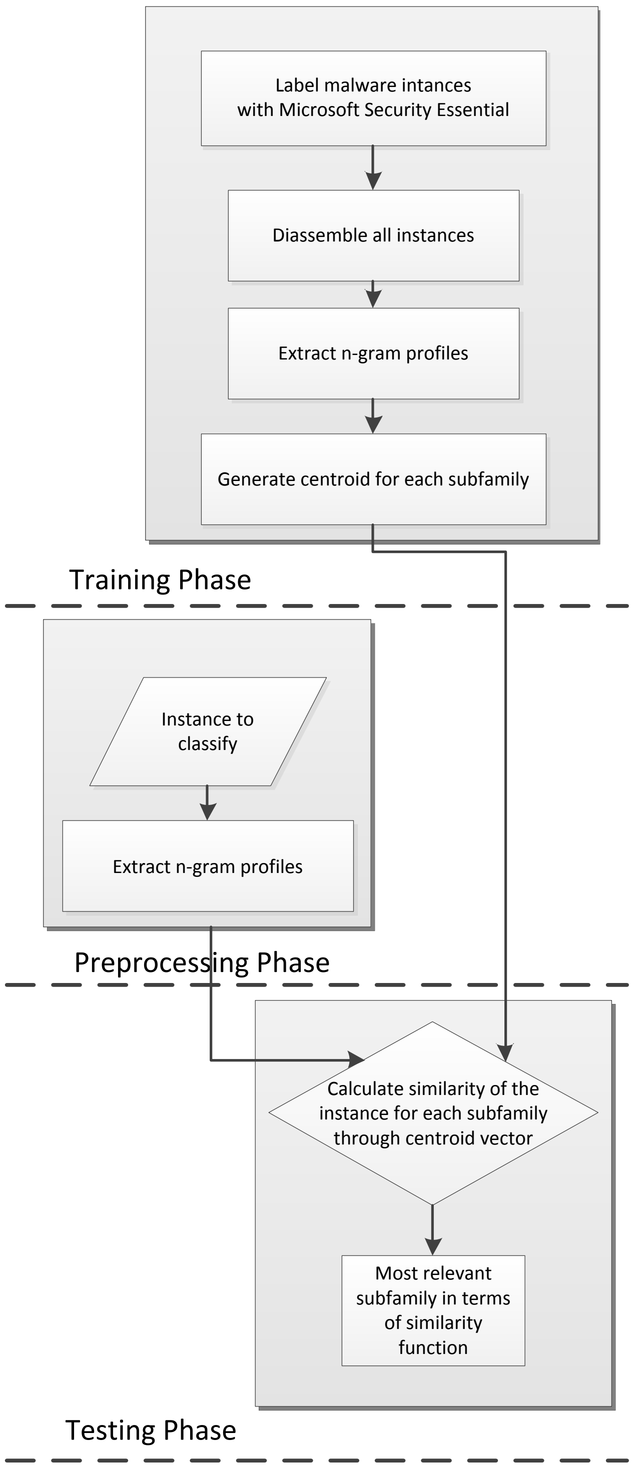

Label malware intances with Microsoft Security Essential
Diassemble all instances
Extract n-gram profiles
Generate centroid for each subfamily
Training Phase
Instance to classify
Extract n-gram profiles
Preprocessing Phase
Calculate similarity of the instance for each subfamily through centroid vector
Most relevant subfamily in terms of similarity function
Testing Phase

Figure 1. Architecture of the malware classification system

especially in terms of the tasks performed and the purpose of the creation. Subfamily is the specialized version of the family that describes malware samples definitely. For instance if a malware labeled as Win32-Ramnit.F by an anti-virus scanner, this means the malware belongs Win32-Ramnit family and Win32-Ramnit.F subfamily.

Centroid of the subfamily comprises the most frequent n-gram of the subfamily instances. In other words, n-grams (words or terms), which occur with higher document frequency in the subfamily instances, are used to construct the centroid vector. So the subfamily is represented by its centroid vector. For instance, centroid of the subfamily is presented by $\overrightarrow{C_s}$ as follows:

$$\overrightarrow{C_s} = \begin{pmatrix} n - gram\ with\ highest\ df\ value \\ n - gram\ with\ second\ highest\ df\ value \\ \vdots \\ n - gram\ with\ L^{th}\ highest\ df\ value \\ \end{pmatrix}$$

where *df* is the document frequency.

To classify an instance, similarity function is calculated by counting the number of matching n-gram (term) for each centroid of the subfamily.

$$Common(C_{s_i}, \vec{m}) = \begin{cases} 1, & (C_{s_i} \epsilon\ \vec{m}) \\ 0, & otherwise \end{cases} \qquad (1)$$

$$Sim(\overrightarrow{C_s}, \vec{m}) = \sum_{i=0}^{L} Common(C_{s_i}, \vec{m}) \qquad (2)$$

$$Class(\vec{m}) = \max(\bigcup_{i=0}^{sub\ number} Sim(\overrightarrow{C_s}, \vec{m}\ )) \qquad (3)$$

where m denotes malware whose family is unknown and it will be determined via presented method. $\vec{m}$ is the n-gram feature vector extracted from unknown malware instance denoted by m. Subindice is the subfamily indexing for s=1,2…15. The function, denoted by $Common$, returns 1 if malware n-gram profile ($\vec{m}$) consists *i-th* n-gram of the centroid of taken subfamily($\overrightarrow{C_s}$) denoted by C_{s_i} otherwise return 0. Equation (2) gives similarity measure between the unknown instance and the subfamily centroid. Similarity measure is the sum of the common n-grams. In Equation (3), after all similarity measures are calculated, the unknown instance is classified as the closest centroid's subfamily.

Process flow is illustrated in Figure 1. When an instance has two or more equal similarity value for two different subfamilies, an error occurs. However this error will be named as the small error because these two or more equal similarity values for subfamily may belong to the same family. As we know, the subfamilies sustain their common family feature. Other types of error are named as big error.

III. EXPERIMENTAL STUDY

In order to perform our experiments, we collect significantly large malware database as stated in the system design section. To obtain more accurate results we count in the subfamilies that contain maximum number of samples in our dataset. In this manner, experiments are carried out 1056 samples belonging to ten families, five of them have two subfamilies, and therefore there exists 15 subfamilies in our dataset. TABLE I indicates how many samples were taken from which subfamily in our dataset. This data set consists only a 2% of the original database. The amount of the sample is sufficient to demonstrate whether n-gram centroid of the subfamily may be used to classify malware instance or may not.

TABLE I. NUMBER OF THE INSTANCES FOR EACH SUBFAMILY

Subfamily Name	Instance Number	Subfamily Name	Instance Number
Win32-Vobfus.Y	13	Win32-Sality.AT	64
Win32-Alureon.H	19	Win32-Small.AHY	69
Win32-Ramnit.F	19	Win32-Renos.NS	95
Win32-Virut.BG	19	Win32-Sality.AM	100
Win32-Alureon.CT	22	Win32-Renos.LT	137
Win32-Agent.ACF	23	Win32-Vobfus.gen!D	183
Win32-Viking.CR	30	Win32-Ramnit.B	200
Win32-Vobfus.AH	42		

To evaluate our methodology, five-fold cross-validation is used: the selected malwares' subfamilies are randomly partitioned into five disjoint sets of approximately equal size, named as "folds". Training and testing phases are performed five times. At each iteration step, one fold is selected as a testing set, and other four folds are combined to form a training set. Therefore, each sample is used five times for training and once for testing. And the estimated error is computed as the total error generated from the five iterations, divided by the total number of the initial tuples.

There are two main parameters in the experimental setup: the first parameter is the size of the n-grams and the second parameter is the number of the list size which is constituted by ranking the n-grams according to their df values in the subfamilies. The size of the n-grams, denoted by n, allows us to decide how long in bytes the n-gram will be. In the experiments, tests are run with n=3, n=4, n=5 and n=6. The second parameter, denoted by L, is chosen to express a subfamily in a simple way. Tests are run with L=40, L=50 and L=60.

TABLE II shows the obtained training error over the parameters n and L as well as TABLE III shows the resulting testing error. As can be seen from the TABLE II and TABLE III, to increase the size of the n-gram does not produce accurate results every time. Because if the parameter n increases, n-grams cannot capture the subfamily features, in contrary the selected n-grams can only represent a feature specific of the sample. However, the opposite case, namely if the n is chosen very small, n-grams can mostly become the common feature of the all subfamilies as well as all samples.

We achieved the highest success rate when n=4 as confirmed by the results in [2] also. Elaborating the parameter choice effects, if the parameter L is increased, the error rate decreases. Since the more common n-gram makes it easy to classify instance appropriately. As maintained in the previous section, the n-gram profiles are captured from the disassembled malware, therefore the space of the n-gram decreases dramatically. For all that, L could not be taken more than 60, due to having very small sized n-gram space (*i.e.*, for Win32-Agent.ACF n-gram feature space is 74)

As a result of the experiment, the most appropriate parameter pair is obtained when n=4 and L=60. The obtained training and testing errors rate for n and L pairs from our experiment are listed in the following TABLE II and TABLE III, respectively.

TABLE II. TRAINING ERROR

N-gram Length	Top L N-gram in the Subfamily Malwares					
	L=40		L=50		L=60	
	Total Error	Without Subfamily Error	Total Error	Without subfamily Error	Total Error	Without subfamily Error
n=3	0.231	0.101	0.150	0.058	0.090	0.024
n=4	0.143	0.056	0.106	0.021	0.053	0.014
n=5	0.124	0.041	0.109	0.024	0.058	0.015
n=6	0.123	0.038	0.115	0.024	0.108	0.019
n=7	0.151	0.031	0.115	0.031	0.098	0.019
n=8	0.125	0.041	0.124	0.037	0.111	0.028

TABLE III. TESTING ERROR

N-gram Length	Top L N-gram in the Subfamily Malwares					
	L=40		L=50		L=60	
	Total Error	Without Subfamily Error	Total Error	Without subfamily Error	Total Error	Without subfamily Error
n=3	0.262	0.109	0.184	0.066	0.131	0.038
n=4	0.169	0.069	0.141	0.037	0.082	0.023
n=5	0.150	0.056	0.128	0.038	0.082	0.026
n=6	0.143	0.043	0.140	0.027	0.134	0.023
n=7	0.170	0.039	0.140	0.036	0.125	0.025
n=8	0.139	0.042	0.148	0.040	0.138	0.034

IV. CONCLUSION

In this paper, a methodology for classifying malware instances from disassembled code by using n-gram feature is presented and it is implemented on a fairly large set. Empirical results demonstrate that the proposed methodology may show acceptable performance in practice. Experimental results show that the classification accuracy for training and testing when n and L are chosen 4 and 60, is achieved at their highest success percentage of %99 and %98, respectively, which seem to be very promising versus the other methodologies.

To improve the accuracy of detection, experiments by using large dataset while using variable length n-gram feature vector of the malware is underway.

REFERENCES

[1] Turkey Computer Emergency Response Team, available at site: http://www.bilgiguvenligi.gov.tr/certen/index.php, last access on-line May 31, 2011.

[2] J. Z. Kolter and M. A. Maloof, “Learning to Detect Malicious Executable in the Wild”, The Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, pp. 470-478, 2004.

[3] T. Abou-Assaleh, N. Cercone, V. Keselj, and R. Sweidan, “n-gram-based Detection of New Malicious Code”, The Proceedings of the 28th Annual International Computer Software and Applications Conference, IEEE Computer Society Washington, DC, USA, pp. 41-42, 2004.

[4] I. Santos, Y. K. Penya, J. Devesa, and P.G. Bringas “n-Grams-Based File Signatures For Malware Detection” The Proceedings of the 11th International Conference on Enterprise Information Systems, Volume AIDSS, pp. 317-320, 2009.

[5] S. Burrows and S. M. M Tahaghoghi, “Source code authorship attribution using n-grams”, In Proceedings of the Twelfth Australasian Document Computing Symposium, A. Spink, A. Turpin, and M. Wu, Eds. RMIT University, Melbourne, Australia, pp. 32–39, 2007.

[6] G. Frantzeskou, E. Stamatatos, S. Gritzalis and S. Katsikas, “Effective identification of source code authors using byte-level information”, In Proceedings of the Twenty-Eighth International Conference on Software Engineering, L. J. Osterweil, D. Rombach, and M. L. Soffa, Eds. ACM Special Interest Group on Software Engineering, ACM Press, Shanghai, China, pp. 893–896, 2006.

[7] G. J. Tesauro, O. J. Kephart, and B. G. Sorkin,“Neural networks for computer virus recognition” , IEEE EXPERT Magazine, pp. 5-6, 1996.

[8] W. Arnold and G. Tesauro, “Automatically Generated Win32 Heuristic Virus Detection”, Virus Bulletin Conference, pp. 51-60, September 2000.

[9] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, “Data Mining Methods for Detection of New Malicious Executables”, Proceedings of the 2001 IEEE Symposium on Security and Privacy, pp. 38-49, 2001.

[10] Z. Liu, N. Nakaya, and Y. Koui, “The Unknown Computer Viruses Detection Based on Similarity” IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences vol.E92-A no.1 pp.190-196, 2009.

[11] A. Walenstein and A. Lakhotia, “The Software Similarity Problem in Malware Analysis”, Dagstuhl Seminar Proceedings 06301 Duplication, Redundancy and Similarity in Software, pp.1-10, 2007.

[12] S. B. Mehdi, A. K. Tanwani, and M. Farroq, “IMAD: In-Execution Malware Analysis and Detection”, the Proceedings of the 11th Annual conference on Genetic and Evolutionary Computation, pp. 1553-1560, 2009.

[13] S.S Anju, P. Harmya, N. Jagadeesh, and R. Darsana, “Malware Detection using Assembly Code and Control Flow Graph Optimization”, Proceedings of the 1st Amrita ACM-W Celebration on Women in Computing in India, article no: 65, 2010.

[14] J.H. Wang, P. S.Deng, Y.S. Fan, L.J Jaw, and Y.C Liu, “Virus Detection Using Data Mining Techniques”, In Proceedings of the IEEE 37th Annual International Conference on Security Technology, pp.71-76, 2003.

[15] M. Siddiqui, M. C. Wang, and J. Lee, “A survey of Data Mining Techniques for Malware Detection using File Features”, In proceedings of the 46th Annual Southeast Regional Conference, pp. 509-510, 2008.

[16] M. Bailey, J. Oberheide , J. Andersen ,Z. M. Mao, F. Jahanian, and J. Nazario, “Automated Classification and Analysis of Internet Malware”, Proceedings of the 10th international conference on Recent advances in intrusion detection, pp. 178-197, 2007.

[17] Microsoft Security Essential, available at site: http://www.microsoft.com/security_essentials/, last access on-line May 31, 2011.

[18] PEiD tool, available at site: http://www.peid.info/, last access on-line May 31, 2011.

Protecting Remote Component Authentication

Rainer Falk, Steffen Fries
Corporate Technology
Siemens AG
D-81739 Munich, Germany
e-mail: {rainer.falk|steffen.fries}@siemens.com

***Abstract*—Component authentication allows verifying the genuineness of various components being part of a machine or a system or connected to control equipment. Various technologies are used, ranging from holograms, hidden marks, special inks to cryptography-based component authentication. Typical cryptography-based mechanisms employ a challenge-response-based component authentication mechanism. These mechanisms have been designed originally for a local verification, i.e., for an authentication performed in direct vicinity of the product to be verified. This paper describes an attack on challenge-response component authentication when supporting a remote component authentication and describes a new security measure to prevent this attack.**

Keywords-device authentication, counterfeiting, tunneled authentication

I. INTRODUCTION

Authentication is an elementary security service proving that an entity in fact possesses a claimed identity. Often natural persons are authenticated. The basic approaches a person can use to prove a claimed identity are by something the person knows (e.g., a password), by showing something the person has (e.g., passport, authentication token, smart card), or by exposing a physical property the person has (biometric property, e.g., a fingerprint, voice, iris, or behavior). Considering the threat of counterfeited products (e.g., consumables, replacement parts) and the increasing importance of ubiquitous machine-based communication, also physical objects need to be authenticated in a secure way. Various different technologies are used to verify the authenticity of products, e.g., applying visible and hidden markers, using security labels (using e.g., security ink or holograms), and by integrating cryptographic authentication functionality (wired product authentication token or RFID (Radio Frequency IDentification) authentication tag).

One important driver for verifying the authenticity of products is safety. For example, counterfeited electrical components as switches or fuses can cause physical damage when they do not fulfill electrical safety norms (e.g., by causing electric shock to humans or a fire). Other examples are provided through electric safety devices as e.g., overvoltage protecting devices or an earth leakage circuit breaker which do not fulfill the expected functionality.

Focus of this paper is a challenge-response authentication of a physical object, e.g., an electric component. This authentication technology has the advantage that a control or supervisory equipment can automatically verify the authenticity of installed components. Local object authentication is used widely e.g., for authenticating battery packs or printer consumables (toner / ink cartridges). Here, cryptographic challenge-response based authentication is applied. Highly cost-optimized solutions are available commercially that allow to use these technologies also in low-cost devices. The authentication protocol is, however, not being designed to protect against man-in-the-middle attacks as these are not relevant in the targeted use case.

Section II gives an overview on challenge-response based object authentication. The scenario for remote object authentication is described in Section III as well as typical technical solutions, and their susceptibility to man-in-the-middle attacks when used for remote component authentication by different verifiers. A new, simple to implement countermeasure protecting against man-in-the-middle attacks is described in Section IV. It enables the re-using of highly cost-optimized object authentication also for remote object authentication, i.e., for a usage scenario not being designed for. This enables to use extremely simple and therefore cost-efficient hardware-based device security mechanisms for purposes not being intended for originally. It can be applied in particular even in those cases when the verifier needs access to the unmodified response value. The application to IP-based smart objects is described in Section VI, providing a highly optimized basis for a secure device identity within the Internet of Things. Related work is summarized in Section VII, before giving a summary and outlook in Section VIII.

II. COMPONENT AUTHENTICATION

A. Overview

Components of a machine (internal or attached) shall be identified securely. This requirements is known for components like ink cartridges, batteries. In industrial machines it applies to replacement parts, sensors, actor devices. Authentication of a device allows a reliable identification of original products.

For authentication a challenge value is sent to the object to be authenticated. A corresponding response value is sent back and verified. The response can be calculated using a cryptographic authentication mechanism or by using a physically unclonable function (PUF). As only an original product can determine the correct response value corresponding to a challenge, the product entity or a dedicated part of the product is thereby authenticated.

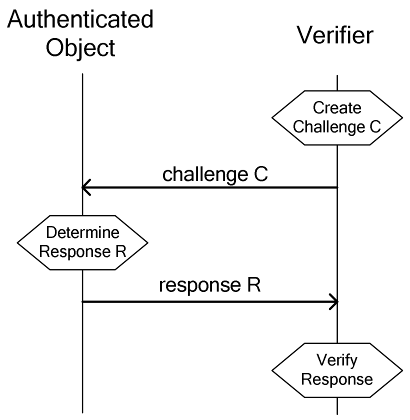


Figure 1. Challenge-Response Object Authentication

Figure 1 shows the schematic data flow between a verifier and an object to be authenticated. The verifier sends a random challenge to the product that determines and sends back the corresponding response. The verifier checks the response. Depending on the result, the product is accepted as authenticated or it is rejected.

B. *Implementation Examples*

Various cryptographic mechanisms can be used to realize a challenge-response product authentication. Basically, symmetric cryptography, asymmetric cryptography, or physically unclonable functions can be used. While in the case of symmetric cryptography also the verifier is in possession of the cryptographic device key and can therefore calculate the expected response value, in case of asymmetric cryptography or PUFs the verifier might not be in a position to calculate the correct response. He has only the option to verify the received response.

This has consequences on whether it is possible to include binding parameters in the response, i.e. to calculate a derived response value. In the symmetric case, the verifier can calculate the expected response and perform the expected response modifications and compare this obtained expected result with the actually received result. However, in the PUF and asymmetric case, this is not possible as the verifier can perform only a verification operation on the received result, but cannot determine a valid response on its own. The verifier needs therefore access to the original, unmodified response value to perform the verification operation. It is not possible to use a derived response value, e.g., by using a keyed hash function or a key derivation function that uses freely definable binding parameters during the response derivation.

Note that the application of symmetric methods may also imply a higher administrative overhead, as the necessary symmetric shared key needs to be securely available on both sides, the product and the verifier.

Implementation examples of product authentication are summarized in the following, describing one example of each category.

- Atmel CryptoAuthentication [1], [2]: A symmetric-key based authentication is performed, intended for example for authenticating battery packs. A challenge-response authentication based on the SHA256 hash algorithm is implemented to compute a keyed digest for the provided challenge value. The input parameter to the SHA256 algorithm is the concatenation of the secret key, the challenge value, and optionally other chip-specific data (serial number, fuses). The challenge is an arbitrary 256 bit value selected by the verifier.
- Infineon ORIGA [3]: An asymmetric authentication based on elliptic curves is performed. A cryptographic operation is performed by the product to be authenticated using the product's private key. The verifier checks the received response using the corresponding public key by performing a cryptographic verification operation on the received value. The verifier does not need access to the products private key.
- Verayo RFID Tag “Vera M4H” [4]: An integrated circuit comprising a physically unclonable function is used to determine a response value depending on the challenge value and hardly to reproduce physical characteristics of the product to be authenticated. Therefore, no cryptographic key has to be stored on the RFID tag as a physical fingerprint of the RFID tag is employed.

C. *Applications / Use Cases*

A reliably identification of products is needed in various use cases. For safety reasons, components can be verified to ensure that no counterfeited products are installed. Also an unverifiable product may be used with conservative operating conditions (e.g., maximum charging current of battery pack) to prevent damage. Another example is authenticated setup of a protected communication session for field level device communication.

III. REMOTE COMPONENT AUTHENTICATION

One important class of use cases is remote component authentication. A machine equipped with or connected to several field devices (sensors, actuators) performs not only a local authentication, but supports a remote authentication of components by a supervisory system.

A. *Use Case Description*

In a remote object authentication, the verifier is remote to the object to be authenticated. The challenge and response values are encoded in messages that are transported over a communication network, e.g., an IP-based network, see Figure 2.

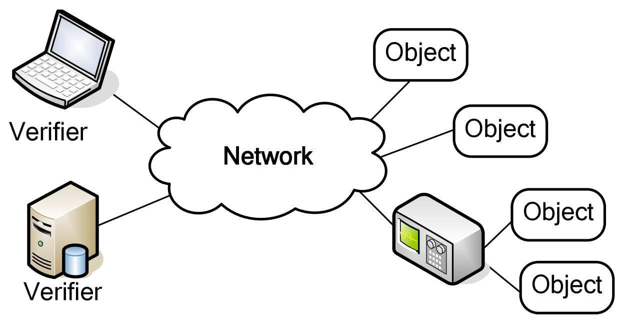


Figure 2. Remote Object Authenticationn

A verifier may be a service technician performing remote maintenance, or an automatic device tracking server or an inventory management server, see Figure 2. The objects to be authenticated may be connected directly to the network, or they may be attached to an intermediary device, e.g. a programmable logic controller.

The challenge response authentication operation is performed by the authenticated object as described above. However, the verifier is not in close vicinity to the object to be authenticated. In some cases, a protected communication channel may be used between the verifier and the intermediary, e.g., IPsec or SSL/TLS.

B. Man-in-the-Middle Suceptibility

In the case of remote object authentication, a (malicious) remote verifier may act as a man-in-the-middle attacker, see Figure 3.

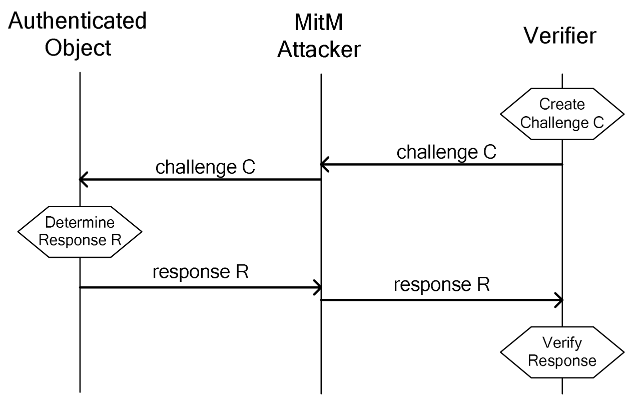


Figure 3. MitM Attack on Object Authentication

An attacking node as "man-in-the-middle" forwards unchanged messages towards and from the object. The verifier having authenticated the object as genuine assumes that it is communicating in fact with the device in the following data exchange. An attacker can use an arbitrary object as oracle that provides valid responses for freely chosen challenges. In consequence, any remote entity that has access to the object authentication functionality can act as a man-in-the-middle and may authenticate itself as the authenticated object if it has a sufficient number of challenge and response pairs. Hence, the object authentication can be manipulated.

When furthermore the authentication response is used for deriving cryptographic session keys, these keys can be derived by an attacker as well.

The fact that such a simple challenge-response authentication is prone to man-in-the-middle attacks is well known and documented also in the corresponding product documentation. For example, the man-in-the-middle attack is mentioned in [5]. In the considered usage environment where authenticated object and verifier are in direct physical connection, the attacker needs both a direct physical access to the attacked object and measurement equipment like e.g., a logic analyzer to analyze the information exchanged between the components. This increases the overall effort of the attack.

IV. EXAMPLE FOR CRYPTOGRAPHIC BINDING REQUIREMENTS

This section motivates the need for cryptographic binding by describing a known weakness. Transport Layer Security (TLS) is a very popular security protocol, which is used to protect web transactions in applications like online banking, to protect the mail communication via IMAP, to realize VPNs or for remote administration. Meanwhile the protocol is available as standard in version 1.2 as RFC 5246 [6].

Early November 2009, a vulnerability has been discovered allowing an attacker to inject data into a TLS connection without being noticed by the client. Such attacks were facilitated by a protocol weakness concerning renegotiation of security parameters. Renegotiation is a TLS feature to exchange fresh security parameter for an existing session. The problem arose due to the missing cryptographic binding between the initially negotiated security parameters and the new parameter set resulting from the renegotiation process. This can be exploited by an attacker as man-in-the-middle attack. A potential attack – a request to a web server – is described in the following, see Figure 4. :

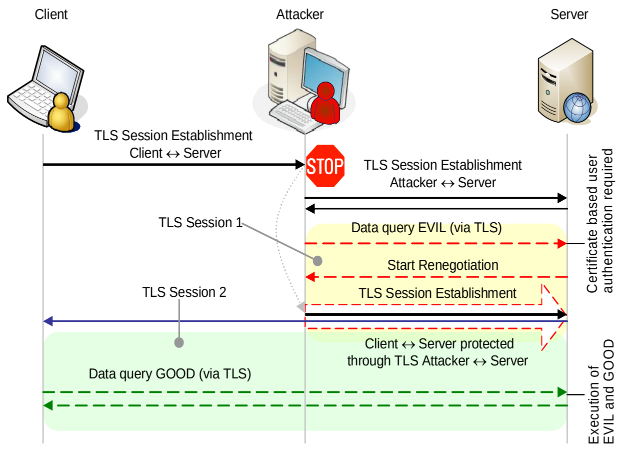

Client
Attacker
Server
TLS Session Establishment Client ↔ Server
STOP
TLS Session Establishment Attacker ↔ Server
TLS Session 1
Data query EVIL (via TLS)
Start Renegotiation
TLS Session 2
TLS Session Establishment
Client ↔ Server protected through TLS Attacker ↔ Server
Data query GOOD (via TLS)
Certificate based user authentication required
Execution of EVIL and GOOD

Figure 4. MitM Attack on Object Authentication

A potential attacker controlling the data path is waiting for a connection attempt by a client. As soon as the client establishes a TLS connection to the server, the attacker delays the client request. Now, the attacker establishes his own TLS connection to the server which is most often server-side authenticated. As soon as the TLS connection is established, the attacker sends a request *EVIL*, requiring the certificate based authentication of the client. This authentication is now being invoked by the server through starting the renegotiation. The *EVIL*-request, which is not authenticated at this time, is stored and executed after successful client authentication. Web servers are typically configured to request client authentication, e.g., when data from an access protected directory shall be read.

The attacker now sends the delayed *Client request* to the server which interprets this message as part of the renegotiation phase, over the TLS protected link between the attacker and the server. This enables an end-to-end key negotiation between the client and the server. All subsequent messages are now secured and the attacker is not able to access them. But, the stored *EVIL*-request was submitted and is executed by the server.

This attack showed on the one hand a potential weakness of TLS due to the missing cryptographic session binding, on the other hand it shows the insufficient integration of TLS into the application, as the web server in this example should have requested an affirmation of the *EVIL* request over the renegotiated TLS session before executing it. This weakness could be exploited for instance for stealing passwords or cookies from Web applications. The weakness has been addressed as part of an update of the TLS protocol using a binding of the initial session to the renegotiated session [7].

V. CHALLENGE BINDING (PRE-CHALLENGE)

The problem originates from the fact that the same authentication mechanism resp. the same authentication key is used in different contexts. Following common security design different keys would be used for different purposes, and to bind the cryptographic material to the intended context (i.e., to derive context-bound session keys from the response).

As in important implementations of component authentication the verifier needs access to the unmodified response, the response value cannot be modified. Therefore, challenge binding is proposed as countermeasure: When a remote verifier cannot select the challenge value, it cannot use the authenticating object as oracle to determine responses for arbitrary challenge values.

A. Challende Binding

The challenge selected by a verifier is bound to the verifier context. This binding operation can be performed by the authenticated object itself or by a (trusted) intermediary node, see Figure 5.

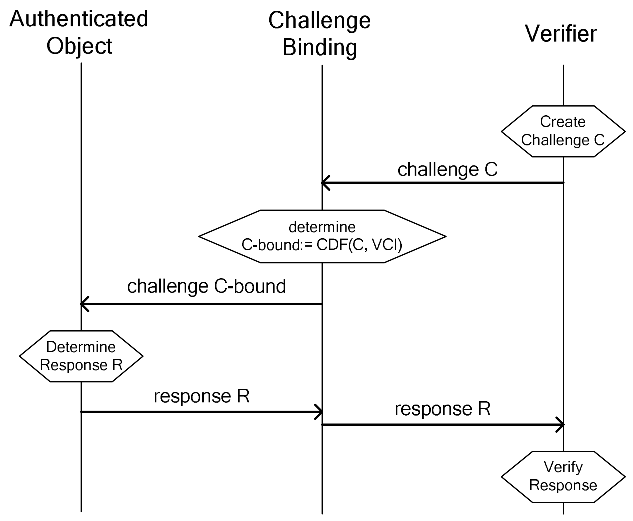


Figure 5. Challenge Binding

The challenge C selected by the verifier is sent to the object directly or to an intermediary node in close vicinity of the object to be authenticated (e.g., a control unit to which a sensor or actuator is directly connected), see Figure 5. This challenge is modified by deriving a bound challenge value C-bound using a non-invertible function (challenge derivation function, CDF). Verifier-dependent context information (VCI) is used as derivation parameter to bind the challenge to the respective verifier. In particular, the verifier’s network address, node identifier, or a session key established between the verifier and the intermediary can be used. The challenge derivation can be performed by both, the object to be authenticated itself, or by a trusted intermediary node.

This modified, verifier-context bound challenge C-bound is forwarded to the object to be authenticated. The object determines the corresponding response and sends it back to the intermediary that forwards the response to the (remote) verifier. The verifier determines the bound challenge C-bound as well, using the selected challenge C and the verifier context information VCI. Note that the VCI can be determined either by the verifier and the intermediary, if both are configured in a way to determine the VCI on available information (like certain address information of the verifier, see also section B below). Alternatively, the VCI may be sent as part of the communication from the intermediary or the verifier.

The remote verifying party can therefore not freely select the challenge for which a response is computed. Anyhow, it can be sure about the freshness of the challenge C-bound for which it received the response as it depends on the pre-challenge C selected by the verifier.

B. Verifier Context Information

Verifier dependent context information is used as derivation parameter to bind the challenge to the respective verifier. There is a variety of parameters that can be used to specify a verifier context. In particular, the verifier identity, e.g., IP or MAC address, DNS name or URL, an (unpredictable) session ID, or a digital certificate or security assertion may be used. This information can be determined by the intermediary, without direct involvement of the verifier. This verifier context is used as parameter to separate two different verifiers. So it must not be possible in practice for a verifier to act successfully within a verification context belonging to a different verifier.

C. Challenge Derivation Function

Requirements on a challenge derivation function are similar as for a key derivation function, i.e. being non-invertible and pre-image resistant (see [7] and [8] for more specific information on key derivation functions). Therefore, the functions that are typically used for key derivation can be used as challenge derivation function as well. For example, the bound challenge C-bound could be derived as HMAC-SHA1(C, VCI), using the challenge instead of a key, and using VCI as textual string determining the verifier context. Alternative key derivation functions may be the higher SHA methods like SHA256 or SHA512 in combination with the HMAC or symmetric algorithms like the AES in CBC-MAC mode [9].

VI. Application to IP-based Smart Objects

One possible application of protected remote component identification is IP-based communication within the Internet of Things. A node communicating with a smart object ("thing") over IP-based communication wants to verify the identity of the smart object resp. of a component being part of or being integrated into the smart object. Communication can be realized e.g., using HTTP-based Web Service protected by TLS or by IP-based communication protected by IPsec. A challenge-response based smart object authentication can be integrated in well-know protected communication protocols, as HTTP Digest over unilaterally authenticated SSL/TLS, or EAP, or within IKEv2 for IPsec.

However, the challenge is modified using verifier context information as derivation parameter. Here, besides the nodes identifier (server name resp. IP address) also the used communication protocol can be used as challenge derivation parameter (e.g., "HTTP-DIGEST/TLS" || Server-IP).

VII. Related Work

Most similar to our proposal is the binding of an authentication challenge for a PUF authentication to the hash of the requesting program, see [10]: The verifier selects a pre-challenge, from which a bound challenge is derived using the hash of the verifier program as input to the challenge derivation. Note that the binding to the hash of the verifier program alone, without address information is weaker, as the hash is supposed to be the same on different hosts. Thus, an attacker possessing the verifier program may still perform the attacks described in section II.

The insecurity of tunneled authentication protocols has been analyzed [11]. In real-world environments, often an existing security deployment and authentication shall be re-used for a different purpose. In particular tunneled EAP authentication was considered, e.g., based on PEAP. The described countermeasure was binding cryptographically the results (session keys) of the two authentication runs, i.e., the inner and the outer authentication, or by binding the session key to an endpoint identifier.

Performing a key derivation is a basic building block for designing secure communication. Various required session keys can be derived from a common master session key. NIST recommended a key derivation function, using a usage-describing textual string as derivation parameter [7]. Another example is the pseudorandom function used within TLS [6], which uses secret keys, seeds and textual strings (identifying label) as input and produces an output of arbitrary length. The same approach is taken in the Multimedia Internet Keying MIKEY [12].

It is also known to bind an authentication to properties of the used communication channel [13]. Two end-points authenticate at one network layer and bind the result to channel properties to prevent against man-in-the-middle attacks where the attack would result in different channel binding properties from the viewpoint of the authenticating nodes.

Furthermore, non-interactive key agreement schemes allow to derive a common, shared key material between nodes that have received a key bound to the own identity [14]. No protocol exchange is required to derive this shared key, but the key is derived similar as with a key derivation function. However, the two derivation steps for binding a root key to two node identifiers can be performed commutatively.

VIII. Summary and Outlook

An attack on component authentication has been described where a single genuine component is used as oracle to compute valid authentication responses. A single malicious verifier may use an obtained valid response value to authenticate as the genuine component towards other verifiers. The described attack is made possible by the fact that the cryptographic solution for component authentication is used within a different usage environment than it has been designed for: The attack is relevant when the component authentication for verifying the genuineness of a component is performed not only locally, but also remotely. This attack is also an example that a small functional enhancement – here making an existing functionality accessible remotely – can have severe implications on security.

This paper proposed a challenge binding mechanism as countermeasure for the described attack. The available, extremely cost-efficient object authentication technology can thereby been used securely also for a different purpose than the one it has been designed for originally. An authentication challenge is bound to the verifier so that a remote verifier can neither simulate a local, unbound authentication nor can it simulate an authentication towards a different remote verifier having a different associated verification context. A possible application of this general challenge-binding mechanism is the cost-efficient authentication of components within the Internet of Things.

REFERENCES

[1] Atmel CryptoAuthentication, http://www.atmel.com/products/cryptoauthentication/, last access March 2011

[2] Atmel CryptoAuthentication Product Uses, Application Note, 2009. http://www.atmel.com/dyn/resources/prod_documents/doc8663.pdf, last access March 2011

[3] Infineon ORIGA, http://www.infineon.com/ORIGA, last access March 2011

[4] Verayo http://www.verayo.com/product/pufrfid.html, last access March 2011

[5] Atmel CryptoAuthentication High level Security Models, Application note, 2009. http://www.atmel.com/dyn/resources/prod_documents/doc8666.pdf, last access March 2011

[6] T. Dierks and E. Rescorla: The Transport Layer Security (TLS) Protocol Version 1.2, RFC 5246, August 2008, http://tools.ietf.org/html/rfc5246, last access March 2011

[7] E. Rescorla, M. Ray, S. Dispensa, and N. Oskov: Transport Layer Security (TLS) Renegotiation Indication Extension, RFC 5746, February 2010, http://tools.ietf.org/html/rfc5746, last access March 2011

[8] Lily Chen: Recommendation for Key Derivation Using Pseudorandom Functions, NIST Special Publication 800-108, 2009. http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf, last access March 2011

[9] John Black and Philipp Rogaway: A Suggestion for Handling Arbitrary-Length Messages with the CBC MAC, http://www.cs.ucdavis.edu/~rogaway/papers/xcbc.pdf, last access March 2011

[10] Srini Devadas: Physical Unclonable Functions and Applications, Presentation Slides (online), http://people.csail.mit.edu/rudolph/Teaching/Lectures/Security/Lecture-Security-PUFs-2.pdf, last access March 2011

[11] N. Asokan, Valtteri Niemi, and Kaisa Nyberg: Man-in-the-Middle in Tunnelled Authentication Protocols, LNCS3364, Springer, 2005. http://www.saunalahti.fi/~asokan/research/tunnel_extab_final.pdf, last access March 2011

[12] J. Arkkom, E. Carrara, F. Lindholm, M. Naslund, and K. Norrman: MIKEY: Multimedia Internet KEYing, RFC3830, August 2004, http://tools.ietf.org/html/rfc3830, last access March 2011

[13] N. Williams: On the Use of Channel Bindings to Secure Channels, Internet RFC5056, 2007. http://tools.ietf.org/rfc/rfc5056.txt, last access March 2011

[14] Rosario Gennaro, Shai Halevi, Hugo Krawczyk, Tal Rabin, Steffen Reidt, and Stephen D. Wolthusen: Strongly-Resilient and Non-Interactive Hierarchical Key-Agreement in MANETs, Cryptology ePrint Archive, 2008. http://eprint.iacr.org/2008/308.pdf, last access March 2011

Extended Fault Based Attack against Discrete Logarithm Based Public Key Cryptosystems

Sung-Ming Yen
Dept of Computer Science and Information Engineering
National Central University
Chung-Li, Taiwan 320, R.O.C.
Email: yensm@csie.ncu.edu.tw

Chi-Dian Wu
Dept of Computer Science and Information Engineering
National Central University
Chung-Li, Taiwan 320, R.O.C.
Email: cs122016@csie.ncu.edu.tw

***Abstract*—Since Bellcore's researchers proposed fault based attacks, these attacks have become serious threats to the implementation of cryptosystems. Boneh *et al.* first proposed a fault based attack against the exponentiation algorithm for RSA, and some variants of attack were proposed later. However, the previous variants of similar attack are applicable only to the right-to-left exponentiation algorithm and none of these attacks can be successfully applied to the left-to-right alternative algorithm since 1997. In this paper, we focus on cryptosystems operated under prime-order groups and emphasize that an extended fault based attack against implementations using the left-to-right exponentiation algorithm is possible. Our attack can also be applied to the Montgomery ladder algorithm which is a well-known countermeasure against some critical physical attacks.**

***Keywords*-exponentiation algorithm; hardware fault attack; physical attack; public key cryptosystem.**

I. Introduction

In the past, cryptographers only analyzed the security of cryptosystems by mathematics. However, when cryptosystems are implemented on physical devices, it brings new threats which had never been considered carefully. These new threats are called the physical attacks, such as the side-channel attack [14] and the fault based attacks [1], [4], [8]. Physical attacks utilize the power consumption and program execution time, or disturb the program execution to infer the secret information stored inside the devices, even though these cryptosystems have been proved secure with mathematical approach. So, when implementing cryptosystems, it is usually essential to prevent such kinds of attack.

Both exponentiation and scalar multiplication are the most central computations for many public key cryptosystems. To evaluate exponentiation or scalar multiplication, the left-to-right and the right-to-left algorithms are the two most widely employed methods. Fault based attack was first introduced in 1997, and afterwards many kinds of fault based attack have been proposed to break a variety of cryptosystems. For example, Boneh *et al.* [8] proposed a fault based attack against the right-to-left exponentiation algorithm for RSA by injecting random faults during the computation to reveal the private exponent. Biehl *et al.* presented a similar attack on elliptic curve cryptosystems (ECC) in 2000 and showed that the secret scalar of a scalar multiplication can be revealed by providing illegal input parameters [4]. Berzati *et al.* modified Boneh *et al.*'s attack in 2008 [2]. Biham and Shamir proposed a differential fault attack (DFA) [5] against symmetric key cryptosystems, e.g., DES. All these researches show that fault based attacks are powerful and dangerous to cryptosystem implementations, especially for those on smart cards.

The aforementioned fault based attacks against public key cryptosystems target at the right-to-left exponentiation algorithm (or the right-to-left scalar multiplication for ECC), while in this paper we extend this kind of fault based attack to the left-to-right exponentiation algorithm. The proposed attack assumes the knowledge of the order of a group and the order must be a prime. For performance reasons or security reasons, many important public key cryptosystems, such as Schnorr scheme [19] and ECC [13] (e.g., elliptic curve Diffie-Hellman key exchange [17]), the group order is a prime integer and it is a public information. So, this attack assumption is reasonable and the proposed attack can be applied to these widely employed cryptosystems. Moreover, the proposed attack can also be extended easily to the Montgomery ladder algorithm.

This paper is organized as follows. In Section II, we first introduce RSA and Diffie-Hellman cryptosystems. We also show the algorithms to compute exponentiation. In Section III, the previous fault based attacks against the exponentiation algorithm are reviewed. In Section IV, we propose an extended attack against the left-to-right exponentiation algorithm and show how to apply this attack to the Montgomery ladder algorithm. Section V concludes the paper.

II. Preliminary Background

A. The RSA Cryptosystem

In the RSA [18] cryptosystem, let p and q be two large primes kept secret to the public and $N = p \cdot q$ is the RSA public modulus. The public key e should be relatively prime to $\phi(N) = (p-1) \cdot (q-1)$, and d is the corresponding private key satisfying $e \cdot d \equiv 1 \pmod{\phi(N)}$. To encrypt a message m with the public key e, we compute $C = m^e \bmod N$ and to decrypt a cipher C with the private key d, we compute $m = C^d \bmod N$. Signing a message m, the private computation $S = m^d \bmod N$ is performed and verification of a signature S is to check whether $m = S^e \bmod N$.

B. Public Key Cryptosystems Based on Discrete Logarithm

Many important public key cryptosystems have been designed with their security based on solving the discrete logarithm problem, e.g., Diffie-Hellman key exchange [9], ElGamal scheme [10], Schnorr scheme [19], and ECC [13] (e.g., elliptic curve Diffie-Hellman key exchange [17]). These cryptosystems are constructed over a finite cyclic group and solving the discrete logarithm problem over this group is believed to be hard. The multiplicative group and the additive group on an elliptic curve are two widely used groups for constructing this kind of cryptosystems. The Diffie-Hellman key exchange scheme is reviewed in the following.

Key generation: Let $\mathbb{G}$ be a cyclic group of order p and g is a generator. Each user selects a random integer $x_i \in \mathbb{Z}_p$ as the private key and the public key is $y_i = g^{x_i}$.

Key exchange: To exchange a shared key k_{ab} with another party, user a receives the public key y_b from user b and computes the shared key $k_{ab} = y_b^{x_a}$.

C. Exponentiation Algorithms

Let $d = \sum_{i=0}^{n-1} d_i\, 2^i$ be the binary expression of the exponent d. An exponentiation algorithm computes the value of m^d given the base number m and the exponent d. A variety of efficient exponentiation algorithms have been proposed so far to compute m^d while the binary left-to-right square-and-multiply algorithm (refer to Figure 1) and the right-to-left square-and-multiply algorithm (refer to Figure 2) are the two most widely employed methods [15].

In this paper, the iteration number of the left-to-right exponentiation algorithm is denoted decreasingly from $(n-1)$ downward towards zero and that of the right-to-left version is denoted increasingly from zero upward towards $(n-1)$.

```
    Input: $m, d = (d_{n-1} \cdots d_0)_2$
    Output: $m^d$
01  $R[0] \leftarrow 1$
02  for $i$ from $n-1$ downto 0 do
03    $R[0] \leftarrow R[0]^2$
04    if ($d_i = 1$) then
        $R[0] \leftarrow R[0] \cdot m$
05  return $R[0]$
```

Figure 1. Left-to-right exponentiation.

```
    Input: $m, d = (d_{n-1} \cdots d_0)_2$
    Output: $m^d$
01  $R[0] \leftarrow 1$; $R[1] \leftarrow m$
02  for $i$ from 0 to $n-1$ do
03    if ($d_i = 1$) then
        $R[0] \leftarrow R[0] \cdot R[1]$
04    $R[1] \leftarrow R[1]^2$
05  return $R[0]$
```

Figure 2. Right-to-left exponentiation.

III. Review of Fault Based Attacks against Exponentiation Algorithm

Some fault based attacks against the exponentiation or the scalar multiplication have been proposed [1], [2], [4], [7], [8], [11] so far and can be classified into two categories. The first category of attacks modify the value of the exponent and the second category of attacks disturb the intermediate value of the exponentiation computation, e.g., the value $R[0]$ in the right-to-left exponentiation algorithm.

A. Fault Based Attack on the Exponent

Bao *et al.* [1] proposed a fault based attack to threaten some cryptosystems, e.g., the RSA system. The fault model of this attack is to induce a one-bit fault into the exponent d such that the binary value of certain bit, say d_j, will be inverted. Let d' be the faulty exponent and the faulty output of the exponentiation is $S' = m^{d'} = m^{(\sum_{i=0, i\neq j}^{n-1} d_i 2^i) + \overline{d_j} 2^j}$ where $\overline{d_j}$ is the one's complement of d_j. Given the aforementioned faulty output S' and the corresponding correct one S, the adversary can identify the value of the bit d_j by analyzing the value of $\frac{S'}{S} = m^{(\overline{d_j} - d_j)2^j}$. We have $\frac{S'}{S} = \frac{1}{m^{2^j}}$ if $d_j = 1$, and $\frac{S'}{S} = m^{2^j}$ if $d_j = 0$.

The attack is also applicable to the multi-bit-fault model. Assume d_j and d_k are inverted, the adversary can derive the values of both bits by analyzing $\frac{S'}{S} = m^{(\overline{d_j} - d_j)2^j} \cdot m^{(\overline{d_k} - d_k)2^k}$. In [11], Joye *et al.* extended the attack such that only the faulty result S' is needed with the knowledge of the plaintext m and additionally its order.

B. Bellcore's Fault Based Attack against the Right-to-left Exponentiation Algorithm

Bellcore's researchers proposed the fault based attack [8] to defeat the RSA private computation with the right-to-left exponentiation algorithm (refer to Figure 2). The fault model of Bellcore's attack is a random one-bit fault injected into the intermediate value of $R[0]$ at the end of the iteration $(j-1)$ or at the end of the Step (03) of that iteration. The faulty result of $R[0]$ at the end of the iteration $(j-1)$ can be expressed as

$$R[0] = (m_j^{\sum_{i=0}^{j-1} d_i 2^i}) \pm 2^b \bmod N$$

where 2^b is the injected error in which $0 \leq b \leq n-1$ (n is the bit length of N) and m_j is the base number, e.g., the plaintext in a signature.

The principle of Bellcore's attack. Bellcore's attack consists of the following steps.

1) The adversary collects sufficient faulty signatures S'_j with the corresponding plaintexts m_j by injecting a random one-bit fault into $R[0]$ during each execution of m^d.
2) The adversary analyzes each faulty signature S'_j with the plaintext m_j and he has
$$\begin{aligned} S'_j &= (m_j^{\sum_{i=0}^{j-1} d_i 2^i} \pm 2^b) \cdot m_j^{\sum_{i=j}^{n-1} d_i 2^i} \bmod N \\ &= S_j \pm (2^b \cdot m_j^{\sum_{i=j}^{n-1} d_i 2^i}) \bmod N, \end{aligned}$$
or $S_j = S'_j \pm 2^b \cdot m_j^{\omega} \bmod N$ where $\omega = \sum_{i=j}^{n-1} d_i 2^i$.
3) With the public exponent e and the collected pairs of (S'_j, m_j), the adversary tests all the possible candidates of b and ω by checking whether
$$m_j = (S'_j \pm 2^b \cdot m_j^{\omega})^e \bmod N.$$

To derive the value of ω in each test needs a known part of binary representation of d and at most l unknown bits where l denotes the longest distance between two nearby iterations at which random faults occurred. Suppose the position at which the random fault occurred on $R[0]$ is unknown (i.e., unknown $0 \leq b \leq n-1$) and the time at which the random fault occurred during the exponentiation is unknown (i.e., unknown j) and uniformly distributed over $[0, n-1]$. Let k be the number of necessary collected pairs of (S'_j, m_j). The number of tests necessary to recover d is at most $k \cdot (n \cdot k \cdot \sum_{r=1}^{l} 2^r)$ and the complexity of this attack is
$$\mathrm{O}(n \cdot k^2 \cdot 2^l).$$

In [8, Theorem 3], Boneh *et al.* proved that to recover d with probability at least $\frac{1}{2}$ requires about $(n/l)\log(2n)$ pairs of (S'_j, m_j) and the complexity of the attack becomes
$$\mathrm{O}(n^3 \cdot \log^2(n) \cdot 2^l / l^2).$$

C. Berzati et al.'s Fault Based Attack

In 2008, Berzati *et al.* [2] modified the Bellcore's attack by injecting random one-byte faults into the RSA public modulus N right after some iterations instead of the intermediate value of $R[0]$ of the exponentiation computation. In Berzati *et al.*'s attack, the faulty modulus N can be expressed as $N' = N \pm R_8 \cdot 2^{8i}$ where R_8 is a nonzero random byte value and $i \in [0, \frac{n}{8} - 1]$.

The principle of Berzati *et al.*'s attack. This attack needs to collect a correct signature S and k faulty signatures S'_j. The values of $R[0]$ and $R[1]$ after the computation within the iteration $(j-1)$ are $m^{\sum_{i=0}^{j-1} d_i 2^i} \bmod N$ and $m^{2^j} \bmod N$, respectively. Suppose the fault upon N occurs at the end of the iteration $(j-1)$. The value of the collected faulty signature S'_j becomes
$$S'_j = \Big((m^{\sum_{i=0}^{j-1} d_i 2^i} \bmod N) \cdot (m^{2^j} \bmod N)^{\sum_{i=j}^{n-1} d_i 2^i} \Big) \bmod N'.$$

Let $\omega = \sum_{i=j}^{n-1} d_i 2^i$, so the correct signature can be expressed as $S = m^{\omega + \sum_{i=0}^{j-1} d_i 2^i} \bmod N$. Based on the above expression of S and all the possible candidates of ω and N', the adversary can compute
$$S'_{(\omega, N')} = \Big((S \cdot m^{-\omega} \bmod N) \cdot (m^{2^j} \bmod N)^{\omega} \Big) \bmod N'$$
and the correct values of ω and N' can be determined by checking whether
$$S'_{(\omega, N')} \equiv S'_j \pmod{N'}$$
on all collected faulty outputs S'_j.

Suppose each check requires to determine l unknown bits of ω and $(2^8 - 1) \cdot \frac{n}{8}$ possible byte faults on N, hence the complexity of Berzati *et al.*'s attack is
$$\mathrm{O}((2^8 - 1) \cdot \frac{n}{8} \cdot k \cdot 2^l).$$
It was claimed that under the assumption of known values of all j (i.e., the time the faults occurred) [2] the complexity of the attack becomes
$$\mathrm{O}((2^8 - 1) \cdot \frac{n^2}{8l} \cdot 2^l), \quad \text{if } k = \frac{n}{l}.$$

IV. The Proposed Fault Based Attack against the Left-to-right Exponentiation Algorithm

The proposed fault based attack is based on Bellcore's attack [8] and Berzati *et al.*'s attack [2], but the attack targets at the left-to-right exponentiation algorithm with the additional knowledge of the group order which is a prime.

A. Fault Model

The proposed fault based attack is based on modifying the intermediate value of $R[0]$ within the left-to-right exponentiation algorithm (refer to Figure 1) by injecting random one-byte faults. This random one-byte fault model (i.e., the fault model #3 in [6]) has been considered practical and widely adopted in many fault based attacks [2], [3], [20]. The faulty value of $R[0]$ can be expressed as
$$R[0]' = R[0] \pm R_8 \cdot 2^{8i}$$
where R_8 is a nonzero random byte value and $i \in [0, \frac{n}{8} - 1]$ both are unknown to the adversary.

B. Principle of the Proposed Attack

The proposed attack needs to collect a correct output S and k faulty outputs S'_j by injecting random one-byte faults into the intermediate value of $R[0]$ each at the end of the iteration j where the iteration number j is unknown to the adversary and is uniformly distributed over $[0, n-1]$.

The intermediate value of $R[0]$ after the computation within the iteration j (denoted as $R[0,j]$) is $R[0,j] = m^{\sum_{i=j}^{n-1} d_i 2^{i-j}}$. The value of correct output $S = m^d$ can be expressed as

$$S = m^d = R[0,j]^{2^j} \cdot m^{\sum_{i=0}^{j-1} d_i 2^i}.$$

Suppose the fault upon $R[0]$ occurs at the end of the iteration j. The value of the collected faulty output S'_j becomes

$$S'_j = (R[0,j] \pm \varepsilon)^{2^j} \cdot m^{\sum_{i=0}^{j-1} d_i 2^i} = (R[0,j] \pm \varepsilon)^{2^j} \cdot m^{\omega}$$

where $\varepsilon = R_8 \cdot 2^{8i},\ i \in [0, \frac{n}{8} - 1]$ and $\omega = \sum_{i=0}^{j-1} d_i 2^i$.

To derive the partial value of d, say ω, the adversary needs a previously known $\sum_{i=0}^{r} d_i 2^i$ $(r < j-1)$ and needs to guess at most l unknown bits of d (say $(d_{j-1}, \ldots, d_{r+1})_2$) where l denotes the longest distance between two nearby iterations at which random faults injected.

Suppose that the order of the group is a prime integer and which is known to the adversary. The correct value of $R[0,j]$ can therefore be derived from the correct output S by

$$R[0,j] = (S \cdot m^{-\sum_{i=0}^{j-1} d_i 2^i})^{(2^j)^{-1}} = (S \cdot m^{-\omega})^{(2^j)^{-1}}.$$

The reason of the assumption of a known prime order is to enable the adversary to compute $(2^j)^{-1}$.

Based on the derived ω, $R[0,j]$, and all the possible candidates of ε, the adversary can compute

$$S'_{(\omega,\varepsilon)} = \left((S \cdot m^{-\omega})^{(2^j)^{-1}} \pm \varepsilon \right)^{2^j} \cdot m^{\omega}$$

and the correct values of ω and ε can be verified by checking whether

$$S'_{(\omega,\varepsilon)} \equiv S'_j$$

on all collected faulty outputs S'_j.

Based on the above proposed attack the adversary can recover the binary expression of the private exponent d from the least significant bits towards the most significant bits. However, the last few bits with the most significant weightings cannot be derived by the attack. These few bits, say $(d_{n-1}, \ldots, d_t)_2$ and t is the maximum value for which a fault occurred at the iteration t, can only be obtained by other approaches, e.g., a brute force search. Bellcore's attack and Berzati *et al.*'s attack share the same property of the proposed attack but the brute force search happens at the least significant bits.

C. Practicability of the Attack

We wish to point out that injecting a one-byte fault into the intermediate value of a register of an exponentiation algorithm assumed in the proposed attack would be more practical than injecting a one-bit fault into a register assumed in the Bellcore's attack. Moreover, the aforementioned assumption made in the proposed attack might be much more practical than injecting a one-byte fault into the storage of a cryptographic parameter, e.g., the RSA public modulus N assumed in Berzati *et al.*'s attack. The reason is that usually a cryptographic parameter will be stored in a non-volatile storage, e.g., flash memory or ROM, and a previous random one-byte fault once injected will be difficult to remove and a new one-byte fault to be injected again which is implicitly assumed in Berzati *et al.*'s attack. So, among the aforementioned three fault based attacks, the proposed attack in this paper demonstrates a higher feasibility.

The computation time of an exponentiation algorithm dominates the performance of many cryptosystems. To improve the performance of cryptosystems, especially for those with their security based on solving the discrete logarithm problem, the group $\mathbb{G}$ is usually replaced by a subgroup with a prime order q of which when binary represented the number of bits is much smaller than that of p. This technique was first employed in the Schnorr scheme [19]. For security reasons, elliptic curve based cryptosystems, e.g., elliptic curve Diffie-Hellman key exchange [17], usually choose a prime order [16]. Both the aforementioned prime order of a multiplicative subgroup and the prime order of an elliptic curve are public informations. So, the assumption made in the proposed attack is reasonable.

D. Complexity of the Proposed Attack and Comparison with Other Attacks

Suppose the byte-fault *pattern* $R_8 \in [1, 2^8 - 1]$, the byte *position* $i \in [0, \frac{n}{8} - 1]$ at which the random byte fault occurred on $R[0]$, and the *time* (say, the iteration number $j \in [0, n-1]$) at which the random byte fault occurred during the exponentiation are all unknown to the adversary. In the proposed attack, to perform test of the relationship $S'_{(\omega,\varepsilon)} \equiv S'_j$, the adversary needs to try all the possible candidates of ω and ε to identify the correct values of both ω and ε.

A segment of at most l least significant bits of d will be derived first when the correct value of ω can be identified, and the exact value of the corresponding iteration number j will be found as well. At most $\sum_{r=1}^{l} 2^r$ possible ω will be tested. Other portion of the binary representation of d can be derived in the same approach towards the most significant bits. The correct value of ε can be identify from one of the possible $(2^8 - 1) \cdot \frac{n}{8}$ one-byte faults occurred on $R[0]$. All in all, the number of tests necessary to recover d is at most $k \cdot ((2^8 - 1) \cdot \frac{n}{8} \cdot k \cdot \sum_{r=1}^{l} 2^r)$ and the complexity of this attack becomes

$$O((2^8-1)\cdot\frac{n}{8}\cdot k^2\cdot 2^l).$$

The complexity of the proposed attack is basically similar to Bellcore's attack, and the difference is that we assume random one-byte faults while Bellcore's attack assumes random one-bit faults. The numbers of possible fault patterns in the proposed attack and Bellcore's attack are $(2^8-1)\cdot\frac{n}{8}$ and n, respectively. However, the numbers of necessary faulty outputs (i.e., k) of both attacks are different, and both attacks assume different fault models.

With the knowledge of all values of iteration number j (i.e., the time the random byte faults occurred) as assumed in Berzati *et al.*'s attack [2], the complexity of the proposed attack can be reduced to $O((2^8-1)\cdot\frac{n}{8}\cdot k\cdot 2^l)$. Let $k=\frac{n}{l}$, the complexity becomes

$$O((2^8-1)\cdot\frac{n^2}{8l}\cdot 2^l)$$

which is the same as Berzati *et al.*'s attack.

E. Attack Extension to the Montgomery Ladder

In [12], an exponentiation algorithm based on the Montgomery ladder was proposed to prevent the SPA attack [14], the computational safe-error attack [22], and the memory safe-error attack [21]. The Montgomery ladder algorithm shown in Figure 3 behaves regularly and accordingly it is secure against the SPA attack. Most specially, there is no dummy computation within the Montgomery ladder algorithm so it can be secure against the computational safe-error attack. Any random computational fault occurred will lead to a faulty result of m^d.

```
    Input: $m, d = (d_{n-1}\cdots d_0)_2$
    Output: $m^d$
01  $R[0] \leftarrow 1$;  $R[1] \leftarrow m$
02  for $i$ from $n-1$ downto 0 do
03    $R[\overline{d_i}] \leftarrow R[0]\cdot R[1]$
04    $R[d_i] \leftarrow R[d_i]^2$
05  return $R[0]$
```

Figure 3. Montgomery ladder algorithm.

The proposed fault based attack can be extended to the Montgomery ladder algorithm with the same random one-byte fault model. The byte fault will be injected into the intermediate value of $R[0]$ at the end of a specific iteration j during the exponentiation. The adversary needs to collect sufficient faulty outputs S'_j and a correct output S.

Principle of the attack. According to the basic principle of Montgomery ladder, the intermediate values of $R[0]$ and $R[1]$ after the computation within the iteration j are $R[0,j]=m^{\sum_{i=j}^{n-1} d_i 2^{i-j}}$ and $R[1,j]=m^{(\sum_{i=j}^{n-1} d_i 2^{i-j})+1}=R[0,j]\cdot m$, respectively. So, the output of the algorithm can be expressed as

$$S=m^d=R[0,j]^{2^j}\cdot m^{\sum_{i=0}^{j-1} d_i 2^i}.$$

Providing two initial values $B_0=m^a$ and $B_1=m^{a+1}$ for some integer a, and a k-bit binary bit string $(e_{k-1}\cdots e_0)_2$ representing an exponent $e=\sum_{i=0}^{k-1} e_i 2^i$, we define a function $\mathtt{Mont}(B_0,B_1,(e_{k-1}\cdots e_0)_2)$ which represents the output B_0^e of the Montgomery ladder algorithm. The output $S=m^d$ of the Montgomery ladder algorithm can therefore be expressed as $\mathtt{Mont}(1,m,(d_{n-1}\cdots d_0)_2)$ or $\mathtt{Mont}(R[0,j],R[1,j],(d_{j-1}\cdots d_0)_2)$. If a fault ε is injected into $R[0]$ at the end of the iteration j, then the faulty output S'_j of the Montgomery ladder algorithm becomes

$$S'_j=\mathtt{Mont}(R[0,j]\pm\varepsilon,R[1,j],(d_{j-1}\cdots d_0)_2)$$

where $\varepsilon=R_8\cdot 2^{8i},\ i\in[0,\frac{n}{8}-1]$ and ω represents the value $\sum_{i=0}^{j-1} d_i 2^i$. Here we also assume that the order of the group is a public prime integer, hence the values of $R[0,j]$ and $R[1,j]$ can therefore be derived based on the correct output S by

$$\begin{aligned} R[0,j] &= (S\cdot m^{-\sum_{i=0}^{j-1} d_i 2^i})^{(2^j)^{-1}}=(S\cdot m^{-\omega})^{(2^j)^{-1}} \\ R[1,j] &= R[0,j]\cdot m. \end{aligned}$$

Based on all the possible candidates of bit string $(d_{j-1}\cdots d_0)_2$ (accordingly the value $\omega=\sum_{i=0}^{j-1} d_i 2^i$) and injected byte fault ε, the adversary can compute

$$S'_{(\omega,\varepsilon)}= \mathtt{Mont}\left((Sm^{-\omega})^{(2^j)^{-1}}\pm\varepsilon,(Sm^{-\omega})^{(2^j)^{-1}}m,(d_{j-1}\cdots d_0)_2\right).$$

The correct values of ω and ε can be verified by checking whether $S'_{(\omega,\varepsilon)}\equiv S'_j$ on all collected faulty outputs S'_j.

The complexity of the above attack on the Montgomery ladder algorithm is basically the same as that attacking the left-to-right exponentiation algorithm. An alternative attack approach is that the byte faults are injected to the intermediate value of $R[1]$ instead of $R[0]$ and in this case the faulty output is assumed to be $S'_j=\mathtt{Mont}(R[0,j],R[1,j]\pm\varepsilon,(d_{j-1}\cdots d_0)_2)$.

Practicability of the attack. The proposed fault based attack can break not only the well-known conventional left-to-right exponentiation algorithm but also the enhanced algorithm against side-channel attack and safe-error attack, say the Montgomery ladder algorithm. In fact, the Montgomery ladder algorithm might be more vulnerable to the proposed attack because the algorithm behaves regularly in each iteration.

In the Montgomery ladder algorithm, each iteration performs two similar operations and the total number of operations to be performed within the algorithm is always $2n$. Therefore, a very precise estimation of the computation time for a single iteration is accessible after a few experiments. These experiments can even be performed upon other similar devices. From the view point of controllability of fault occurrence time (e.g., fault injected at the end of an iteration) and accordingly the feasibility of an attack, the proposed extended attack on the Montgomery ladder algorithm becomes more practical than all the previous attacks.

V. Conclusion

In this paper, based on the previous fault based attacks against the right-to-left exponentiation algorithm, we propose a new attack against the left-to-right exponentiation algorithm on some public key cryptosystems, such as Diffie-Hellman key exchange and ECC, if they are constructed under a group with a prime order. The complexity of the proposed attack is the same as that of the previous related attacks. Moreover, the proposed attack can also be extended to threaten the Montgomery ladder algorithm and this extended attack could be even more practical than all other related attacks.

Acknowledgment

This research was supported in part by the National Science Council of the Republic of China under contract NSC 98-2221-E-008-047-MY3.

References

[1] F. Bao, R. H. Deng, Y. Han, A. Jeng, A. D. Narasimbalu, and T. Ngair, "Breaking public key cryptosystems on tamper resistant devices in the presence of transient faults," in *Proc. Security Protocols Workshop 1997*, LNCS 1361, Springer-Verlag, pp. 115–124.

[2] A. Berzati, C. Canovas, and L. Goubin, "Perturbing RSA public keys: an improved attack," in *Proc. CHES 2008*, LNCS 5154, Springer-Verlag, pp. 380–395.

[3] A. Berzati, C. Canovas, and L. Goubin, "(In)security against fault injection attacks for CRT-RSA implementations," in *Proc. Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2008*, pp. 101–107.

[4] I. Biehl, B. Meyer, and V. Muller, "Differential fault attacks on elliptic curve cryptosystems," in *Proc. CRYPTO 2000*, LNCS 1880, Springer-Verlag, pp. 131–146.

[5] E. Biham and A. Shamir, "Differential fault analysis of secret key cryptosystems," in *Proc. CRYPTO 1997*, LNCS 1294, Springer-Verlag, pp. 513–525.

[6] J. Blömer, M. Otto, and J. P. Seifert, "A new CRT-RSA algorithm secure against bellcore attacks," in *Proc. ACM CCS 2003*, ACM Press, pp. 311–320.

[7] J. Blömer, M. Otto, and J. P. Seifert, "Sign change fault attacks on elliptic curve cryptosystems," in *Proc. Workshop on Fault Diagnosis and Tolerance in Cryptography – FDTC 2006*, LNCS 4236, Springer-Verlag, pp. 36–52.

[8] D. Boneh, R. A. DeMillo, and R. J. Lipton, "On the importance of checking cryptographic protocols for faults," in *Proc. EUROCRYPT 1997*, LNCS 1233, Springer-Verlag, pp. 37–51.

[9] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[10] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," in *Proc. CRYPTO 1984*, LNCS 196, Springer-Verlag, pp. 10–18.

[11] M. Joye, J. J. Quisquater, F. Bao, and R. H. Deng, "RSA-type signatures in the presence of transient faults," in *Proc. Cryptography and Coding 1997*, LNCS 1355, Springer-Verlag, pp. 155–160.

[12] M. Joye and S. M. Yen, "The Montgomery powering ladder," in *Proc. CHES 2002*, LNCS 2523, Springer-Verlag, pp. 291–302.

[13] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, pp. 203–209, 1987.

[14] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," in *Proc. CRYPTO 1999*, LNCS 1666, Springer-Verlag, pp. 388–397.

[15] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1997.

[16] National Institute of Standards and Technology, *Recommended Elliptic Curves for Federal Government Use*. In the appendix of FIPS 186-2.

[17] National Institute of Standards and Technology, *Special Publication 800-56A: Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography*. March, 2006.

[18] R. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Comm. of the ACM*, vol. 21, no. 2, pp. 120–126, 1978.

[19] C. P. Schnorr, "Efficient identification and signatures for smart cards," in *Proc. Crypto 1989*, LNCS 435, Springer-Verlag, pp. 239–252.

[20] D. Wagner, "Cryptanalysis of a provably secure CRT-RSA algorithm," in *Proc. ACM CCS 2004*, ACM press, pp. 311–320.

[21] S. M. Yen and M. Joye, "Checking before output may not be enough against fault-based cryptanalysis," *IEEE Trans. Computers*, vol. 49, no. 9, pp. 967–970, 2000.

[22] S. M. Yen, S. Kim, S. Lim and S. Moon, "A countermeasure against one physical cryptanalysis may benefit another attack," in *Proc. ICISC 2001*, LNCS 2288, Springer-Verlag, pp. 414–427.

Implementations of Block Cipher SEED on Smartphone Operating Systems

HwanJin Lee, DongHoon Shin, and Hyun-Chul Jung
Security R&D Team
Korea Internet & Security Agency (KISA)
Seoul, Korea
{lhj79, dhshin, hcjung}@kisa.or.kr

***Abstract*—As more and more people are using smartphones these days, a great deal of important information, such as personal information and the important documents of corporations among other things, are being saved on smartphones. Unlike a PC, people can access another person's smartphone without great difficulty, and there is a high possibility of losing one's smartphone. If smartphone is lost without encryption, important information can be exploited. In addition, the open cryptographic library for PCs cannot be used due to the limited performance of the smartphone. This paper introduces the optimization implementation technique for the smartphone OS and the results of using that technique. In addition, the results of a speed comparison with the open cryptographic library will be presented. According to the results of comparing the one-time encryption implementation time with the open cryptographic library, the performance time was improved by 12% for Windows Mobile, 8.57% for iOS, and 39.62% for Android.**

Keywords-SEED; Windows mobile; iOS; Android; implementation; blockcipher.

I. Introduction

Use of the smartphone is increasing due to the rapid development of IT technologies. Now, we can make a call or use various functions such as e-mailing, web surfacing, and Office programs simply with a small smartphone. However, the risk of loss or theft is also increasing due to the device's small size and light weight. Due to their inconvenient portability, around 200,000 smartphones are lost or stolen every month on average.

Loss of a smartphone can lead to a serious leak of an personal information, as smartphones contain a large amount of such information (call details, received messages, phone numbers, schedules, location information, financial transaction information, etc.). And smartphones are also used for business and sales purposes. So, secondary damage can be caused if a smartphone containing a corporation's sensitive information is lost or stolen.

Data encryption is very important to protect the various types of personal information and confidential information stored in the smartphone. SEED is block cipher that can be used for data encryption. Because that SEED is Korean and International Standard, the usage of SEED has been covered the security service applications in Korea. So, application of smartphone which needs security service must be implemented SEED in Korea. However, a smartphone has limited power and offers inferior performance compared to a PC. Therefore, it is difficult to use an open cryptographic library such as OpenSSL, which is designed for the PC environment, in a smartphone. We need to study on the way for the effective use of SEED in smartphone.

This paper presents the results of implementing the block cipher SEED to a smartphone. The results of a comparison with open cryptographic libraries (OpenSSL, BouncyCastle) will also be presented. The SEED is a block cipher established as an international standard ISO/IEC and the Korean standard. Section 2 introduces the SEED and open cryptographic libraries; Section 3 introduces smartphone operating systems; Section 4 presents the implementation method; Section 5 presents the implementation and comparison results; and Section 6 presents the conclusion.

II. SEED and Open Cryptographic Libraries

A. SEED

SEED is a 128-bit symmetric key block cipher that had been developed by KISA (Korea Internet & Security Agency) and a group of experts since 1998. SEED has been adopted by most of the security systems in Korea. SEED is designed to utilize the S-boxes and permutations that balance with the current computing technology. The input/output block size and key length of SEED is 128-bits. SEED has the 16-round Feistel structure. A 128-bit input is divided into two 64-bit blocks and the right 64-bit block is an input to the round function, with a 64-bit sub-key generated from the key scheduling [1].

SEED has been adopted as an industrial association standard of Korea (TTA, Telecommunication Technology Association) at 1999 and ISO/IEC and IETF International Standard at 2005 [2, 3].

Classification	**Number and Title**
Korean Standard	TTAS.KO-12.0004 : 128-bit Symmetric Block Cipher(SEED)
International Standard	Standard ISO/IEC 18033-3 : Information technology - Security techniques - Encryption algorithms - Part 3 : Block ciphers.
	IETF RFC 4269 : The SEED Encryption Algorithm ※ RFC4269 obsoletes RFC 4009.

Furthermore, several standards (Cryptographic Message Syntax, Cipher Suites to Transport Layer Security, IPsec etc.) have been adopted as Korean standard and International standards [4~6].

Classification	Number and Title
Korean Standard	TTAS.KO-12.0025 : Modes of Operation for The Block Cipher SEED
International Standard	IETF RFC 4010 : Use of the SEED Encryption Algorithm in Cryptographic Message Syntax (CMS)
	IETF RFC 4162 : Addition of SEED Cipher Suites to Transport Layer Security(TLS)
	IETF RFC 4196 : The SEED Cipher Algorithm and Its Use with IPsec

The usage of SEED has been covered the security service applications such as, e-Commerce, e-mail, dedicated receiver with Broadcasting, financial service, data storage, electronic toll collection, VPN, Digital Right Management, etc.

In particular, under the auspices of the Bank of Korea, eleven banks and one credit card company has launched a pilot service of K-cash for about 600 franchisees in Seoul since July of the year 2000. SEED has been used to protect the privacy of the users and the transaction data in this service.

B. *Cryptographic Libraries*

1) OpenSSL

OpenSSL [7] is an open cryptographic library written in C language. OpenSSL implements most of the encryption algorithms we use in our daily life, such as symmetric-key ciphers, hash functions, public-key ciphers, message authentication codes, and SSL/TLS. OpenSSL complies with and can be used in various platforms such as Unix, Linux, and Windows.

2) BouncyCastle

BouncyCastle [8] is an open cryptographic library written in Java and C# language. BouncyCastle can be implemented with J2ME, JDK 1.6, and C# API. Like OpenSSL, most encryption algorithms used in our daily life have been implemented.

III. Smartphone Operating System

A. *Windows Mobile*

Windows Mobile [9] is a mobile operating system developed by Microsoft that was used in smartphones and mobile devices. It is used in a variety of devices such as smartphone, vehicle on board and portable media devices etc. The current and last version is "Windows Mobile 6.5".

Windows Mobile can be classified as Application, Operating System and Cryptographic Service Provider (CSP). And it includes the Cryptography API set (CryptoAPI), which provides services that enable application developers to add encryption and decryption of data.

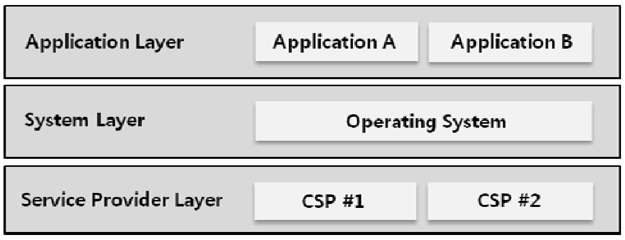


Figure 1. Architecture overview of Windows Mobile

Windows mobile provides service encryption, hashing and digital signature, etc. Windows mobile supports many block ciphers such that DES, 3DES, IDEA, CAST, RC5 and AES-128/192/256. Supporting hash functions are MD2, MD4, MD5, SHA1/256/384/512 and HMAC. And digital signatures are RSA, DSS and ECDSA. But it does not support SEED. So, application developer must program SEED or port OpenSSL for using SEED in Windows mobile.

B. *iOS*

iOS [10] is the operating system that runs on iPhone, iPod touch, and iPad devices. Although it shares a common heritage and many underlying technologies with Mac OS X, iOS was designed to meet the needs of a mobile environment, where users' needs are slightly different.

The iOS security APIs are located in the Core Services layer of the operating system and are based on services in the Core OS (kernel) layer of the operating system. Core Services layer includes key chain service, certificate, key, trust service and randomization service and supports library for a symmetric-key Cipher, digital signature etc. The iOS security APIs are based on services in the Core Services layer, including the Common Crypto library in the libSystem dynamic library. Common Crypto library supports DES, 3DES, AES-128/256 block cipher. And it supports MD2/4/5, SHA-1/224/256/384/512 hash functions. But it does not support SEED as windows mobile. So, SEED be programmed or ported Openssl in application of iOS.

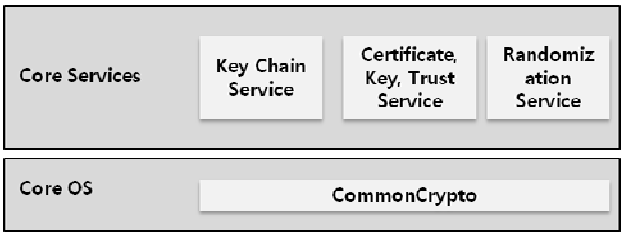


Figure 2. Architecture overview of iOS

C. Android

Android [11] is an open-source software stack for mobile devices that includes an operating system, middleware and key applications. Google Inc. purchased the initial developer of the software, Android Inc., in 2005.

Android can be classified as system layer, crypto library and crypto class. Android developer can use java.security package and Javax.crypto package in crypto library. java.security package provides all the classes and interfaces that constitute the Java security framework. java.security package supports certificate and signature. javax.crypto package provides the classes and interfaces for cryptographic applications implementing algorithms for encryption, decryption, or key agreement. javax.crypto package supports stream cipher, block cipher, hash function and MAC. Android does not support SEED. To use SEED, android developer must program SEED or port BouncyCastle.

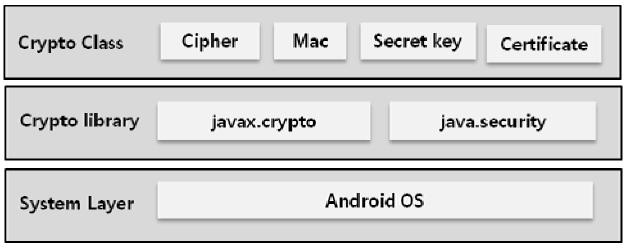


Figure 3. Architecture overview of Android

IV. 3. OPTIMIZATION ON SMARTPHONE OS

A. General

1) *32-bit processing*
- ARM core version 6x version runs on 32bit. Therefore, data processed by the algorithm is implemented in 4 bytes.

2) *Little endian*
- For the ARM core environment, an algorithm was designed and implemented, based on the little endian.

3) *Memory management*
- The embedded system has a limited and inefficient memory allocation system. The memory space for temporary variables was allocated in advance and re-used.

4) *Loop optimization*
- Unnecessary loops were reduced and repetition was removed. For example, the ARM7 and ARM9 processors require one cycle for subtraction processing, and three cycles for selection control processing. That is, if subtraction is configured in a loop, four cycles are required to process one loop. In addition, we reduce the number of iteration as much as possible, when "for loop" is used.

5) *Variable declaration and operation*
- The unsigned type variable was used. Regular processes handle "unsigned integer" operation much faster than "signed" operation.

B. Android

Android application development codes are compiled in a machine-independent byte code, and executed by a Dalvik virtual machine in the Android device.

That is, as the Java code is executed in the Java Virtual Machine, the Android app is executed in the Dalvik Virtual Machine. Therefore, Android's processing speed is slower than that of native codes.

To improve the processing speed, Java provides JNI (Java Native Interface), which accesses the source coded in other languages, and executes the source code.

The operation of memory copy and XOR (exclusive OR), which are most frequently used in actual encryption algorithms, were compared. It takes about 380ms on average when the System.arraycopy Java method is used for 4096 bytes of data. However, it takes 266ms on average when implementing memory copy using the memcpy C function in the JNI. As a result, we can see that the performance of memory copy was improved by 140%. For XOP operation, Java takes 830ms and JNI takes 161ms, which implies that the performance of XOR operation was improved by 500% or more.

Therefore, if the algorithm is implemented appropriately in the Android environment using the JNI, a very efficient encryption algorithm can be implemented.

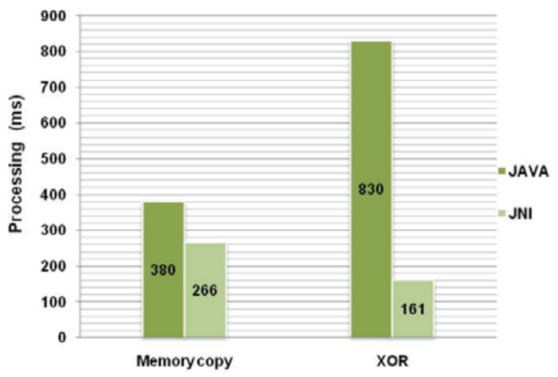
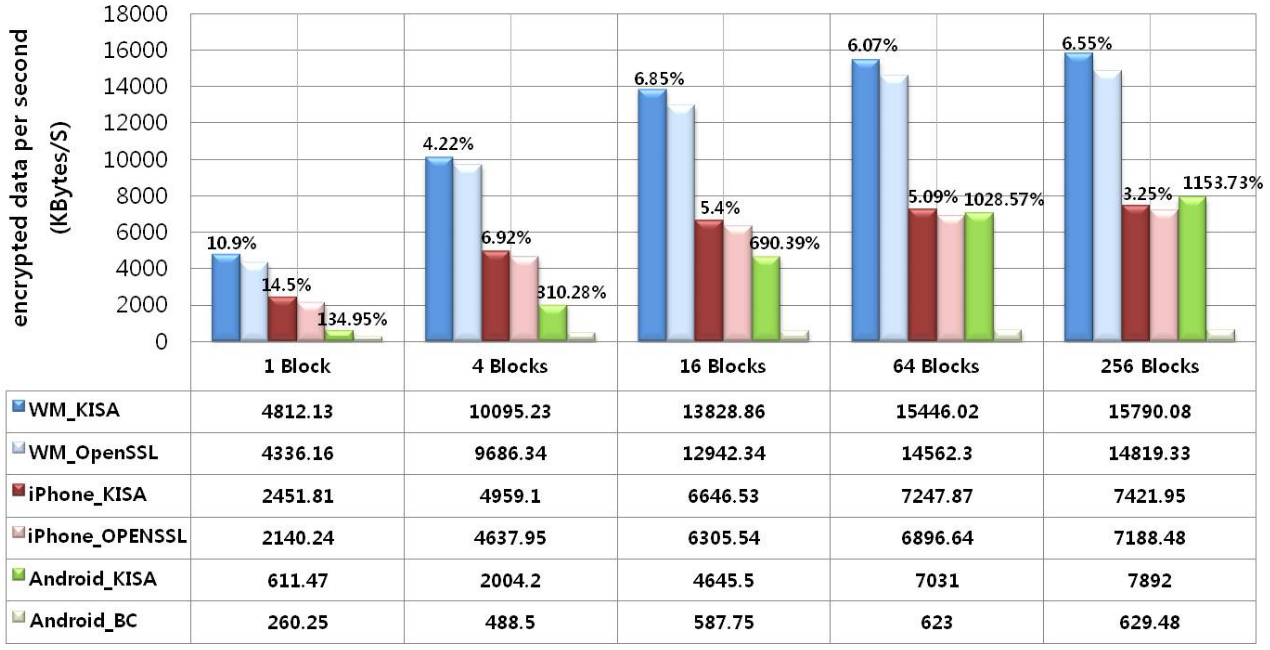


Figure 4. Processing time of JAVA and JNI

V. IMPLEMENTATION RESULTS

This paper presented the results of the SEED encryption speed and power consumption for each smartphone OS. In addition, the results of the comparison with open cryptographic libraries (OpenSSL, BouncyCastle) are also presented. Speed was compared with the amount of encrypted data per second and the one-time encryption time. Power consumption was compared, based on the time used to consume 1% of the battery. The algorithm test was conducted in the following environment.



	1 Block	4 Blocks	16 Blocks	64 Blocks	256 Blocks
WM_KISA	4812.13	10095.23	13828.86	15446.02	15790.08
WM_OpenSSL	4336.16	9686.34	12942.34	14562.3	14819.33
iPhone_KISA	2451.81	4959.1	6646.53	7247.87	7421.95
iPhone_OPENSSL	2140.24	4637.95	6305.54	6896.64	7188.48
Android_KISA	611.47	2004.2	4645.5	7031	7892
Android_BC	260.25	488.5	587.75	623	629.48

Figure 5. The amount of encrypted data per second

TABLE I. TEST DEVICE SPECIFICATIONS

Platform	Device	OS	H/W specification	Open cryptographic libraries
Windows Mobile	HTC HD2	Windows Mobile 6.5	- CPU : Qualcomm, Snapdragon 1Ghz - RAM : 448MB	OpenSSL
iPhone	Apple iPhone 3GS	iOS 4.0	- CPU : ARM, Cortex A8 600Mhz - RAM : 256MB	OpenSSL
Android	HTC Nexus One	Android 2.2 (Froyo)	- CPU : Qualcomm, Snapdragon 1Ghz - RAM : 512MB	BouncyCastle

A. Results of speed comparison

The data processing amount per second refers to the amount of data be encrypted by SEED for one second. CBC (Cipher-block chaining) was used as a mode of operation. The length of the input plaintext was set to 1, 4, 16, 64, and 256 blocks respectively (1 block = 128 bits). The length of the input plaintext is 4 means input of SEED-CBC is 4 blocks plaintext and operating number of CBC is 4. The length of the input plaintext is considered for size of information in storage of smartphone such as phone numbers, Social Security number etc. To reduce the possibility of error, we repeat more than 1000 seconds. And results were divided by process time. For example, in case 4 blocks, SEED-CBC only operates 4 blocks plaintext until time is more than 1000 seconds.

According to the results of the comparison of the data encrypting amount per second, Windows Mobile was improved by 4.22~10.98%, whereas iOS and Android were improved by 3.25~14.56% and 134.95~1153.73%, respectively. As a result, windows mobile and iOS did not show a great improvement. But Android showed high improvement rate for using JNI. We think that increasing of encrypted data per second depending on the length of the input plaintext because memory input/output. With smaller block, it is decreasing number of memory access. 1 block access memory 256 times when 256 blocks access memory once. For the cases of Windows Mobile and iOS, the improvement rate decreases while increasing the size of input data to encrypt. We think that the reasons are way of optimization. It's not optimization of algorithm structure but optimization of algorithm implementation. It is depend on time of access memory and smartphone OS. For the case of Android, the improvement rate increases continuously, because that BouncyCastle is very slow. But it will reach the limit.

The one-time encryption time refers to the time period required to execute an encryption algorithm once. To reduce the possibility of error, the average of the results of 80,000 repeated executions was calculated. According to the results of the comparison, Windows Mobile improved by 12%, whereas iOS improved by 8.57% and Android by 39.62%. The results are different from results of encrypted data per second. We think that the reasons are difference of calculation methods. Results of encrypted data per second include checking time whether more than 1000 seconds or not.

TABLE II. ONE-TIME ENCRYPTION TIME

(ms/1call)	Windows Mobile	iOS	Android
Ours	0.0044	0.0064	0.032
OpenSSL (BouncyCastle *)	0.0050	0.0070	*0.053
Improvement rate (%)	12%	8.57%	39.62%

B. *Electricity consumption*

The time taken to use 1% of the battery was measured, when executing an encryption algorithm. The results of the comparison of electricity consumption show a power saving of 19.9% for Windows Mobile, and of 14.85% and 12.36% for iOS and Android respectively.

TABLE III. ELECTRICITY CONSUMPTION COMPARISON

sec	Windows Mobile	iOS	Android
Ours	120.21	125.3	112.45
OpenSSL (BouncyCastle *)	100.2	109.1	100.08
Improvement rate (%)	19.97%	14.85%	12.36%

VI. CONCLUSION

It is essential to protect the information stored in smartphones owing to their increasing popularity and various functions. However, a method of optimization other than a PC is required due to the limited performance of the smartphone. This paper presents the results of the optimal application of the block cipher SEED, which was selected as both the international standard ISO/IEC and the Korean standard, to the smartphone. Also, the results of comparing the open cryptographic library OpenSSL (including the SEED) with BouncyCastle were presented.

According to the results of optimizing and implementing the SEED in smartphones, SEED provided better performance than the open cryptographic library in all areas (i.e., data processing amount per second, one-time encryption execution time, and electric consumption). In particular, Android showed a remarkably enhanced performance than other OS when optimized with the JNI.

Recently, the smartphone has been attracting ever more attention among the general public, and the number of service environments that capitalize on this increasing attention is also rising continuously. Accordingly, the number of environments that require a high level of security is also increasing, such as smart office and mobile cloud. To create a safe smartphone use environment, methods of optimizing the various encryption algorithms are needed. Consequently, research on optimizing the algorithms, such as public key cipher and SSL/TLS, is needed.

ACKNOWLEDGMENT

This research was supported by the ICT Standardization program(2011-PM10-18) of MKE(The Ministry of Knowledge Economy)

REFERENCES

[1] Korea Internet and Security Agency, Block Cipher Algorithm SEED, Available from http://seed.kisa.or.kr/eng/about/about.jsp. [Accessed: June 5, 2011]

[2] ISO/IEC 18033-3, Information Technology–Security Techniques–Encryption Algorithms–Part 3: Block Ciphers, ISO, 2005.

[3] Lee, H., Lee, S., Yoon, J., Cheon, D., and J. Lee, "The SEED Encryption Algorithm", RFC 4269, December 2005.

[4] J. Park, S. Lee, J. Kim, and J. Lee, "Use of the SEED Encryption Algorithm in Cryptographic Message Syntax (CMS)", RFC 4010, February 2005.

[5] H.J. Lee, J.H. Yoon, and J. Lee, "Addition of SEED Cipher Suites to Transport Layer Security (TLS)", RFC 4162, August 2005.

[6] H.J. Lee, J.H. Yoon, S.L. Lee, and J. Lee, "The SEED Cipher Algorithm and Its Use with IPsec", RFC 4196, October 2005.

[7] OpenSSL, Available from http://www.openssl.org/. [Accessed: June 5, 2011]

[8] BouncyCastle, Available from http://www.bouncycastle.org/index.html. [Accessed: June 5, 2011]

[9] Windows Mobile 6.5 MSDN, Available from http://msdn.microsoft.com/en-us/library/bb158486.aspx. [Accessed: June 5, 2011]

[10] iOS Refernece Library, Available from http://developer.apple.com/library/ios/navigation/. [Accessed: June 5, 2011]

[11] Android, Available from http://www.android.com/. [Accessed: June 5, 2011]

[12] C. R. Mulliner, “Security of Smartphones”, Master’s Thesis submitted in University of California, Santa Barbara, 2006.

[13] Lisonek, D. and Drahansky, M., "SMS Encryption for Mobile Communication", the 2008 International Conference on Security Technology, 2008, pp. 198-201.

[14] F. Fitzek and F. Reichert, "Mobile Phone Programming and its Application to Wireless Networking", Springer, 2007.

[15] L. Shurui, L. Jie, Z. Ru, and W. Cong, "A Modified AES Algorithm for the Platform of Smartphone", 2010 International Conference on Computational Aspects of Social Networks, 2010, pp. 749-752.

[16] F. M. Heikkila, "Encryption: Security Considerations for Portable Media Devices," IEEE Security and Privacy, vol. 5, no. 4, 2007, pp. 22-27,

[17] JLC. Lo, "A framework for cryptography algorithms on mobile devices", Master's Thesis, 2007.

[18] S. M. Habib and S. Zubair, "Security Evaluation of the Windows Mobile Operating System", Master's Thesis, 2009.

[19] Asghar, M.T., Riaz, M., Ahmad, J. and Safdar, S., "Security model for the protection of sensitive and confidential data on mobile devices", International Symposium on Biometrics & Security Technologies, 2008, pp. 1-5.

[20] A. Visoiu and S. Trif, "Open Source Security Components for Mobile Applications," Open Source Science Journal, vol. 2, no. 2, 2010. pp. 155-166.

An Approach for Enhancing the Security of Remotely Accessible, Sensitive Services with On-Site Monitoring

Tuomas Kekkonen
VTT Technical Research Centre of Finland
Oulu, Finland
tuomas.kekkonen@vtt.fi

Teemu Kanstrén
VTT Technical Research Centre of Finland
Oulu, Finland
teemu.kanstren@vtt.fi

***Abstract*—Commonly, the deployment environment of software based services cannot be predicted in advance, which leads to need to have specific solutions to monitor their security and reliability during runtime. These services are also commonly accessed remotely, leading to further complexity in their monitoring and analysis. The work presented in this paper proposes a solution for enhanced security monitoring of such services, providing for increased confidence in the service security and reliability. The proposed solution uses near-real time information collected about the service and its environment during its use. The approach is evaluated using a case study of monitoring a mobile payment service showing increased awareness and confidence of service security.**

Keywords-network monitoring; security situation awareness; security management; security policy; network capture

I. Introduction

Modern software intensive systems are increasingly pervasive and used to perform critical and sensitive operations. In many cases, the service is provided as remotely accessible through various terminals, such as mobile devices. For example, a push-mail service can be used to deliver email to mobile devices from corporate and Internet service provider networks, or a mobile payment system can be used to make payments with a mobile device over various vendor and provider networks. These services deal with sensitive corporate or personal information, and handle transactions related to real world assets such as money. These types of services are commonly deployed in unpredictable environments, where their security and reliability are impacted by varying constraints and evolves over time. The hosting infrastructure itself varies across deployments and the clients used to access these services can be varied and mobile.

Typically for such services, their criticality and sensitivity is recognized and thus a security policy is defined describing their security aspects and the required countermeasure to possible security threats. However, extensive and relevant management of such security policies is an exhaustive and resource draining task. Tasks such as monitoring of specific services and their auditing are important but often difficult to make cost effective. Especially in multi-domain environment where services are widely deployed and serving sensitive data, such as described above, the assurance of the operation is vital but difficult to maintain. The operating environment sets limitations to the monitoring and also affects the operation efficiency. However, the affected efficiency may not be perceived without some means of monitoring.

A perceived weakness of security monitoring is also the inability to respond to different situations during monitoring. In a complex system the management of countermeasures is often tedious and handled by the administration. However, in case of remotely accessed services, it is often possible to apply an approach where a certain profile of the expected system behavior is defined and a specific response is defined for such observed situations. As a response, for business purposes it is often enough to deny the service when problems are observed until the situation has been analyzed and resolved. If this early response is not applied, later problems can escalate into more serious issues, complaints and reclamations.

This paper presents an easily deployable and flexible monitoring solution for remotely accessed services. It gives the security management a source of information for the observed services and the ability to evaluate the capability of the service prior to transactions. We present our approach from the viewpoint of generally monitoring different aspects of software-intensive remotely accessible services, and use a case study of its application on a mobile payment system to evaluate the efficiency of the approach. The nature of the mobile payment service as widely spread to multiple and technically varying locations helps illustrate the different aspects of the approach.

The rest of the paper is structured as follows. In Section II, we discuss the problem domain of monitoring networked services. In Section III, we present our approach for security monitoring. In section IV, we describe a case study of applying our approach on a mobile payment system. In Section V, we discuss the results and the observed applicability of our approach. Finally, conclusions end the paper.

II. Network Monitoring

Network monitoring can have different motives and targets. A common goal is to detect failing or slow components to be able to address possible issues promptly and to gain more successful operation. Network monitoring is also used to enhance security or performance. Typical solutions applied for network security monitoring are intrusion detection and prevention systems [1]. These solutions are based on detecting anomalies and tracking signatures on network traffic and behavior (e.g., [2], [3]). These systems can support a wide range of analysis and reaction, providing protection for, and information about, vulnerable states in a system. They operate best in large scale networks and can identify great amount of different types of events. However, due to their operational constraints such as the vast amounts of information for large-scale network traffic that needs to be processed, they generally can only focus on a shallow analysis of e.g., network packet headers. A comprehensive analysis would require deep packet inspection level of analysis, which would require large-scale resources that would make their applicability non cost-effective. This enhancement is studied by [4].

The effectiveness of these tools also depends on the quality of their signature databases and algorithms that are used to analyze the captured information. These are provided and updated by the product manufacturer or community. Thus their relevance depends on the activity of the signature providers.

The target of intrusion detection and prevention is specifically to provide information about the activity the system is facing. This information is usually used to block some traffic in the network and is an administrative task. Security management on the other hand has the intent to define security policies and to know that the system is implemented according to the specification. In a growing and dynamically changing systems where maintenance and installations are done regularly the upkeep of the security policy requires audits on systems to evaluate its conformity with security policy.

In any case, continuous monitoring of the defined security policies is needed to gain confidence on the secure and reliable operation of the deployed services. To be commonly applicable, such monitoring solutions need to be defined as addressing the security and reliability requirements while at the same time to be defined within capabilities of network capture based feature inspection. The goal should be to minimize the intrusiveness of the monitoring by making the implementation of security monitoring possible without requiring additional installation or changes in the environment where the product is delivered.

Requirements and capabilities for network monitoring are set by the target network and service. Optimally, we should be able to monitor as much as possible with a minimal set of deployed monitoring points. For example, a single Ethernet based subnet environment could be monitored from a single accessible location when all data is broadcast to all parties. When possible, the network architecture can also be optimized for the monitoring purposes by adding monitoring to a central location.

A. Network Security Management

A product, which requires to be deployed in varying types of environments can face various issues such as malicious users and excessively loaded networks. When deployed to such unsafe environments, the operation cannot be assured and the cause for this can be difficult to track down. Again, security management needs to assure that the system implements the required security policy. In an optimal case, this can be assured by providing the service product with a complete setup including hardware and software that is configured according to all the security policy requirements. Other approaches include providing just the software to be run on the customer environment, which is often a more practical scenario, especially for smaller customers. The choice of deployment strategy impacts also the complexity of governing the service security and reliability in its environment. A more complete deployment can mitigate the possible risks but not completely eliminate them, including the need to manage the infrastructure, its updates and other security and reliability aspects.

Most network security breaches originate from poorly defined or implemented security and lack of security management. According to annual Symantec survey [5], the most common reason for data breaches in 2009 was the theft or loss of material. The second highest category of the survey is named insecure policy, referring to the failure to create and administer policies to enhance security, including the user operation.

Technical breaches in software security are caused by vulnerabilities, which are exposed to the hacker through their own investigation or through other venues. Effective security management needs to react to the discovery of these vulnerabilities before they are widely in use by the hacker community. The required responses such as software updates and counter-measure configurations need to be managed by the users or the administration.

Overall, we can state that the overall security of a network is a sum of the combined operation of all participants in the network. Therefore the behavior of all users in the network is of interest to the different parties using the network and should be monitored. For example, the presence of clients with diverging traffic patterns or outdated software can be taken as indicators of potential security and reliability confidence lowering aspects in the overall system. Similarly, any issues observed on the server side are obvious indicators for the expected service quality. Our approach adds the valuable information of the network monitoring to cover the shortages of typical security management solutions to make it more interactive.

III. Our approach

The goal of the work presented in this paper is to enhance security awareness for remotely deployed services handling sensitive information. Here this information is provided not only for the service administration but also the end user as a mobile payment application. This includes defining a security policy for the remotely deployed software that can be used as a basis to provide assurance of its operation. Because we cannot assume to have complete knowledge of the actual deployment environment, the policy needs to specify generic requirements that reflect security awareness in the context of that specific service. To cover the security policy, the available monitoring information in an actual deployment environment is mapped to each requirement specified in security policy. This model is then used as a basis to monitor the system and to evaluate its security capabilities in a continuous manner.

Security management usually defines the monitored policy by evaluating the risks the system might encounter. In business minded information security the risks can be classified into assets, vulnerabilities, countermeasures and threats [6]. Here these classes are used to define an example security policy and security features based on this risk analysis. This is to show an example of the process of adapting a risk analysis into security policy and network monitoring features. We will present an example of a concrete security policy for a mobile payment system in Section IV-D.

Our approach consists of the following steps

1) Risk analysis
2) Security policy definition
3) Infrastructure analysis
4) Monitoring point definition
5) Continuous monitoring
6) Information synthesis and presentation to user

We focus in this work on the monitoring aspects, but it is worth noting that we use input from the previous steps that are assumed to be present. In this work we build on the work presented in [7]. As a security policy definition assume the presence of documents such as Assurance Profiles as described in [8].

Risk analysis evaluates the key points in the system and determines the value of their successful operation. It also attempts to list the situations the system might face and prioritize the risks related to those situations according to their severity. These risks are covered with certain security measures defined in a security policy. It specifies, which features the system components need to implement to lower the possibility of a risk and to decrease its effect in case of occurring. What also needs to be defined is the details of the actual monitored variables used to determine the state of requirements on the security policy.

After this it is up to the security management to assure that the policy is followed. To arrange an automated monitoring on the security awareness inside the system the security management needs to analyze the infrastructure to identify the points where a certain security policy should be present. This analysis is a source of information when the monitoring is deployed into the system. Successful deployment of monitoring with appropriate security policy definitions results in providing a continuous view of the security policy compliance on the system.

A multi-domain monitoring solution has to be flexible in supporting different types of target network environments. This includes being able to perform within the limitations set by the communication protocols and network monitoring capabilities available. To organize the monitoring information, we use a four way model based on security events, traffic rations, security presence and online testing. The following subsections describe each of these attributes and the type of variables they consist of. Variables are defined in accordance to security policy that is defined according to the risk analysis. In addition to these aspects, the security policy needs to define the expected values for the variables, including the limits for each value to remain within accepted range. Practically, choosing the variables is also affected by what is possible to monitor in the service infrastructure, which is affected by factors such as infrastructure access and available monitoring tools.

A. Security events

With the term security events we refer to the input events generated from security monitoring tools such as intrusion detection systems. Security situation awareness is a related term used in research to gather, combine, and understand constantly updating security state of the overall system. Methods for security situation awareness are presented, for example, by [9] and [10], largely based on combining of observed security events. The security events are the events that are used as a basis to for the analysis of the current security situation, providing a basis for assessing the confidence in the security and reliability status of the observed system.

For example, an intrusion detection system generates alerts based on rules, which have different priorities and cause different types of alarms. Alarm priority can also be used to provide added value for security situation awareness, for example, to make a more informed decision on how to respond. Automated analysis systems such as intrusion detection systems often cause false alerts and therefore the response needs to be managed manually.

In this work, the information provided by security events constitutes of alerts generated by Snort intrusion detection system and errors reported by the target mobile payment service. When the analysis of their combination is observed as revealing a potential issue, the security situation in the network is considered reduced.

B. *Traffic Ratios*

We define expected network traffic patterns in terms of network protocol distributions. This describes the expected ratio of observed traffic in terms of different protocols, including the encrypted and non-encrypted versions of the same protocols. Additionally, specific protocols can also be classified as insecure and thus their presence at all can be considered a feature for analysis. For example, existence of bittorrent protocols can lead to high traffic loads and decreased performance on network, and thus bittorrent can be classified as unwanted because of possibly causing decreased availability.

The ratio of these different types of protocols is compared to the amount of total observed traffic and to observed ratios between different types of communication in the network. These defined ratios are taken as rules for the expected traffic distribution between the different traffic types. Traditional methods of anomaly detection or pattern recognition methods can be applied to enforce these rules. However, more specific rules can also be defined to monitor specific protocols in a presumably known network, simplifying the required monitoring process.

C. *Security Presence*

Security presence is a term we use to refer to a system having security related features and mechanisms present in the network. Its security value is in estimating the general security awareness. Security presence related variables are those detected from the application variables communicated over the network protocols.

Practically, security presence is detected by monitoring identifiers in the traffic such as short passwords, clear text passwords, old software versions and insecure operating system versions reported in communication messages and security events. Those are mainly gathered from HTTP messages and this is implemented by analyzing pcap files produced by Wireshark.

D. *Online Testing*

Online testing refers to active stimulation of running service elements and observation and analysis of the results. An online testing tool can test the service with simulated requests and evaluate the correctness of responses. While it is possible to create customized tools for these purposes, we prefer to use existing tools for genericity and cost-effectiveness purposes. Basic examples of online test tools available on existing systems, which are also used in this work are ping and port scanning tools such as nmap.

Online tests provide information of the systems conformity with requirements and current capability to serve requests with tests that simulate actual operation. A false reply to a request can identify an issue in the authenticity of the communication partner. Additionally the delay in responses imply a general failure in service to respond to requests. These are clear indication of service operation capabilities.

IV. Mobile Payment Case Study

In this section, we present a case study of applying our approach in the domain of mobile payment. Based on a previously performed risk-analysis and the available monitoring options for the service infrastructure of a mobile payment service, we define a set of relevant security events and network traffic properties and ratios. From this we define what we consider the relevant properties for observing the security presence of the mobile payment service in our environment. Further, we show how using a set of existing network monitoring tools we monitor this security presence from the service infrastructure and use the information in the process of the mobile payment.

A. *Mobile Payment*

Mobile payment services allow customers to make purchases using their mobile devices as means of payment. A scenario with monitoring goals and end user delivery is presented on Figure 1 Presented is the typical architecture for a mobile payment that includes:

- Vendor node that receives information of payment for the product delivery.
- Cellular operator provides a phone number for the product and informs the mobile payment operator that a call has been made.
- Payment network instructs the vendor node to commit the delivery when cellular operator informs about the call. Also manages the communications to possible mobile vendor nodes.
- Client device makes a call to the cellular operator provided number that is operated by the mobile payment operator.

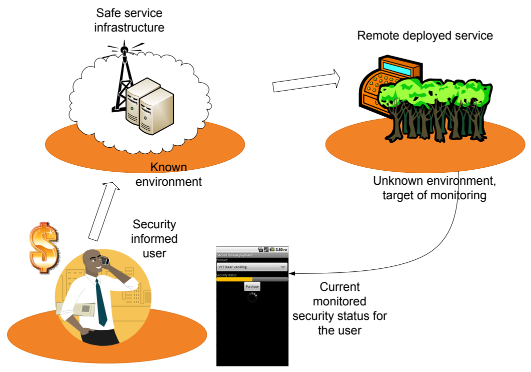

Safe service infrastructure
Remote deployed service
Known environment
Unknown environment, target of monitoring
Security informed user
Current monitored security status for the user

Figure 1. Architecture of the monitoring approach including the mobile payment process with the user, mobile payment provider and a product.

Mobile payment systems are not standardized and implementations in different countries and companies are varied

[11]. Connection from the mobile payment provider to the vendor node is implemented by the mobile payment provider and does not have specific standards. The different aspects of the provided service such as the information security vary and their quality is up to the different operators. Our target systems operate in Finland, where currently most implementations use a premium billing method based on billing the customer through the cellular provider. As mobile payment systems operate over networks provided by different operators and varying communication links to vendor nodes, it serves as a suitable case to evaluate our approach.

In premium billing the customer places a call to a given number and uses the phone keypad to provide input to the service. Product is billed as a normal service number without any form of identity verification, allowing anyone to use the service with any available phone. This can be a negative factor enabling the illicit use of the device but also a usability enhancing feature increasing sales because of the ease of use. Downsides in this form of billing is the traceability of transaction and reliability of delivery. Network provider might charge the customer even if theres a problem with delivery and the customer needs to place a reclamation to get refunded. The mobile payment provider and cellular network provider operate as separate entities without direct communication, which makes tracing the transactions over network boundaries difficult. Possible technologies in mobile payment field to address these issues include secured sim-cards capable of requesting pin code on authorization request, near field communication and credit card involvement but none of these are standardized for this domain.

There is a need to improve the reliability of the delivery of products and security of the information exchange. The system cannot observe if the product is successfully delivered. Therefor it is beneficial to have a solution to monitor that the system is operating in desired conditions. Even if we cannot completely remove these issues, we need to strive for an optimal operational environment for the service. We cannot monitor the mobile payment service parameters directly due to confidentiality requirements and due to available access over different networks. For this reason, we aim to monitor generic network parameters according to monitoring parameters defined as relevant for the service in the security policy definition phase.

For confidentiality reasons, we describe here a simulated service environment rather than a production system. We constructed our simulation system using actual production systems as input to define a realistic environment for our evaluation. This also included defining a security policy and test scenarios to address real situations on network operation and typically appearing problems according to input gathered from a mobile payment operator.

To illustrate how the provided monitoring information could be utilized by the service end user, we developed a mobile phone application to display the observed payment service security status to the user. This application is described in Section IV-E.

Considering our general targeted domain of remotely accessed services being hosted over several partners infrastructure, we see mobile payment as a good example. It needs to have parts installed in every vendor that provides possibility for the payment to be done with a mobile phone. Because of this, the environment for those services is varying and assuring its operation through monitoring is beneficial.

B. Environment

Our test environment was constructed with a set of virtual machines, requiring some specific monitoring approaches. To optimize our ability to capture monitoring information, the network in a virtual environment can operate in a way where all the virtual hosts receive all the traffic. This is possible because the typical network latencies and transmission capabilities are not limited in a way they are in actual networks. Ease of setup and maintenance also helped us in our experiments.

The virtual network environment consisted of three hosts. One host was running the target service and had been set up as described below on Table I. Second was running monitoring software. All the tools required for monitoring were installed on this host and run on intervals. Third host was running simulation software, responsible for generating traffic and different situations on the network. These situations are described in Table II.

The tests were run on two different systems. Both were put under test with two different usage scenarios. With this setup the capability of the measuring system for providing useful information was evaluated. The first system was to demonstrate a typical dated system, which cannot perform according to todays security requirements. The second system was an up to date system with operating system less vulnerable to security attacks and up to date software. The setups of both systems are described in Table I. The setups and simulations were set so that not only mobile payment related features but general security features that can affect performance in the target could be detected. This is why HTTP server and browser types are relevant.

C. Simulation Scenarios

Both systems were put under test with two types of simulation. This results in four different scenarios where reaction of the two systems can be observed. The scenarios are described on Table II.

Network load and behavior was simulated using ping flooding and simulated HTTP requests. This is used to cause computers to suffer from the increased network and therefor processing load. HTTP requests were made with the Curl application and scripted to be performed in intervals. Slowloris is a HTTP flooding tool that attempts to create a denial of service situation by sending partial HTTP requests

Table I
DESCRIPTION OF SYSTEMS USED IN TEST RUNS.

Type	Feature	State
Less secure	Operating system	Windows XP
	Operating system version	SP2 retail
	Apache version	2.2.10
	Browser type	Internet Explorer
	Browser version	6
	Other services	2
Highly secure	Operating system	Linux
	Operating system version	Kernel 2.6.32-26
	Apache version	2.2.16
	Browser type	Mozilla Firefox
	Browser version	3.6.8
	Other services	0

Table II
DESCRIPTION OF USAGE SIMULATION DURING TEST RUNS.

Load level	Description	Method	Value
Low load	Network load	ping flood	none
	System load	cpuburn	none
	HTTP load	Timed requests	normal
	HTTP vulnerability	Slowloris	none
	Payments	Scripted	few
	Logins	scripted ssh logins	few
	Behavior	HTTP passwords	none
High load	Network load	ping flood	full
	System load	cpuburn	full
	HTTP load	Timed requests	10/second
	HTTP vulnerability	Slowloris	run
	Payments	Scripted	10/second
	Logins	scripted ssh logins	1/second
	Behavior	HTTP passwords	few

to keep multiple sockets to the server open. It was used to exploit the old version of apache HTTP server.

The goal of the scenarios was to evaluate the ability of the monitoring system to prevent the billing of any excess payment from the customer. This can be handled either at the service provider end or at the customer end. The service provider can refuse the service request based on observed issues in the service infrastructure when a request is received. At the client end the terminal can produce a warning to the user about potential security and reliability issues observed in the mobile payment service that is being accessed, when this information is available.

Payment requests are included in the simulation scenarios. Their success rate is the measure of successful operation of the mobile payment system. The success rate of providing useful security status information to the monitoring system client (service provider or customer) in terms of correctly identifying the simulated security problem scenarios is the measure of our approach in evaluation. Payments were simulated with tool developed by a mobile payment service provider.

D. Monitoring

As described before, our solution is mainly targeted as an automated and easily deployed security policy monitoring system. For our case study we defined a security policy to see different levels of implemented security in the set of chosen scenarios. The policy defines a feature and a risk type it is covering. Some features target the host running the service and some measure the relevant properties of the overall network environment. The target system was desired to comply with features specified on Table III. In this table examples of monitoring sources used are also described. Single requirement is usually covered with multiple sources.

Table III
DEFINED SECURITY POLICY. DESCRIBES THE SECURITY REQUIREMENTS ORDERED BY THE RISK TYPES THEY ARE COVERING. ONE MONITORING SOURCE IS ALSO DESCRIBED FOR EACH.

Definition	Source
Asset	
Hardware operates properly.	Produces traffic
Stays connected to network.	Ping response time
Responds quickly to requests.	Service response time
Service is not under heavy load.	No service errors
Vulnerability	
Follows generally secure behavior.	Vulnerability scan
Does not have extra ports open.	Port scan
Traffic profile.	Ratio of HTTP
Does not send clear text passwords.	HTTP passwords
Does not use short passwords.	HTTP auth
Uses latest software versions.	HTTP Server field
Uses secure operating system.	TCP fingerprint
Countermeasure	
Uses firewall.	Port scan
Encryption is used.	HTTPS messages
Uses secure browser.	HTTP agent field
Monitor system does not fail.	New monitor data
Threat	
Non familiar users on network.	Failed SSH logins
Service is not abused.	IDS alerts
Host is not under attack.	IDS alerts

This model was constructed to detect issues in implementation of the security policy and the relevant security features. This is based on identifying security related features through monitoring the messaging in the on-site network. The monitored values were combined to derive a binary statement of each defined security feature. These statements construct the model for the overall security. Variables from network traffic were chosen to provide needed information to cover the points defined in security policy. The variable types are described on Section III.

E. Deployment

A mobile application was developed to list available mobile payment services and to display their current security level. This was to demonstrate the usability of the information also for the service end user. Information was delivered through a socket connection from the monitoring network to a mobile application. The information was visible to the user at any time as a status bar indicating the current level of security in terms of the number of features reported as ok. Then it is up to the users decision to determine when the security level is acceptable. In a more refined version the service provider can decide how a certain security level affects the transaction and the detailed security information can be hidden from the end user if that is desired.

V. Results and Discussions

The goal of our case study was to evaluate the effectiveness of the applied approach in detecting any reliability and security issues in the current security implementation level and the usage of the network. Gained benefit would have been shown as better success rate on service. This would result in reduction of failed attempts due to early rejection of requests, which is desired for the transaction process.

Running the tests shows that the pre defined rules are easily detected in the deployed systems and the security features are correctly mapped. Results of each scenario is listed on Table IV. This specific set of scenarios also shows that the information is usable in determining the reliability of the service delivery. The less secure system setting is clearly vulnerable. This is mainly based on the research on outdated software being less secure and effect of security awareness of users in overall security. Outdated Windows system and Apache server alone constitute vast amount of vulnerabilities. This is caused by the fact that they are highly popular and therefor highly exploited. Resulted product delivery effect was however only observed as slight increased delay. The best scoring scenario is the high security setup with low load. On that scenario the monitoring detects the situation to be less loaded and security setup to be proper. This is what was intended to be discovered with the monitoring.

A more detailed simulation scenario could have exploited the target systems more and affected the operation more dramatically. This was not intended but a general simulation scenario was more useful to illustrate the capabilities to detect general security situation, not a scenario where the system is under carefully planned attack.

With this monitoring two aspects of the network security are known. The intrusion detection system style monitoring provides information about the current usage the target system is facing. Here this is combined with knowledge of the level of security implemented in the target system. This information is used to evaluate the capability of the system under different circumstances since both details are known. Then the service requests can be rejected when there is a bigger change of information leakage or failed transaction.

The viewpoint here is to observe the monitoring as the service provider. Their goal is not to let consumers make requests when there is a high possibility that the request will fail or information will get captured. They can define rules to remain in a certain level of confidence in the successful delivery. They do not need to require all the requirements to be fulfilled but some might complement others. For illustration in this work the service provider sets their rules as defined here:

- Hardware must not fail.
- Must respond on network.
- If service is under load software needs to be up to date.
- If service is abused firewall has to be used.

Table IV
Results from the four scenarios. 1. low security, high load 2. high security, high load 3. low security low load 4. high security low load

Feature	1	2	3	4
Hardware operates properly.	OK	OK	OK	OK
Stays connected to network.	OK	OK	OK	OK
Does not have extra ports open.	FAIL	OK	FAIL	OK
Uses firewall.	FAIL	OK	FAIL	OK
Responds quickly to requests.	FAIL	FAIL	OK	OK
Traffic profile.	FAIL	OK	OK	OK
Follows generally secure behavior.	FAIL	FAIL	OK	OK
Encryption is used.	FAIL	OK	OK	OK
Does not send clear text passwords.	FAIL	FAIL	OK	OK
Does not use short passwords.	FAIL	OK	OK	OK
Uses latest software versions.	FAIL	OK	FAIL	OK
Uses secure operating system.	FAIL	OK	FAIL	OK
Uses secure browser.	FAIL	OK	FAIL	OK
Service is not abused.	FAIL	FAIL	OK	OK
Host is not under attack.	OK	OK	OK	OK
Service is not heavy load.	FAIL	FAIL	OK	OK
Non familiar users on network.	FAIL	FAIL	OK	OK
Monitor system does not fail.	OK	OK	OK	OK

With these rules the service provider would refuse requests on scenario 1. When put under loaded situation the system evaluated as a high security system would still allow requests to be made according to the rules specified. The low security system would reach a state of refusing requests when facing the load. The efficiency to reject requests in early stage based on the assumption that the request would fail later anyway is the key to provide enhanced operability for the system in business sense. Without this type of information, each failed transaction has to be dealt individually to refund the customer.

Various viewpoints can be taken on the application of the monitoring information we provide. From the service provider viewpoint it may be bad to show detailed security level information to the user. Instead it may be better to just refuse service and notify the administration to address any observed issues. However in some cases the user can make better use of the information such as when reading email using a publicly accessible network. Simple level of security was presented to the user in our case as illustrated on Figure 1. In this case, the more detailed information can be provided to let the user make a more informed decision. This is ultimately a business decision based on different properties such as the operating environment and the business domain.

Depending on the interests of the company in question and the liability responsibilities the service provider may or not have interests in securing the service or identifying the customer. In an environment where the legislation is highly consumer protective the provider needs to have mechanisms for traceability and strengthened security.

In implementing any monitoring solution there is also always the trade-off between implementing specific monitoring for a chosen service and system and in implementing more generic monitoring approaches. Here we apply a generic monitoring approach that aims to make use of generic network parameters, although the same approach could also be used to make use of more service specific parameters. However, in many cases there are factors such as legislation that prevent the use of specific service information such as customer identifiers or email message identifiers for any such purposes and the generic approach is the best suited one. The generic approach is also easier to reuse across different systems. On the other hand, we recognize the possibility of more specific monitoring approach to provide more service specific information. Here our environment and domain has limited our access to service specific information and thus we apply a more generic approach. In other cases, an analysis of different possibilities is needed to identify the best suitable approach in this regard.

The used infrastructure was based on virtualized network, which slightly affects the credibility of the results. The virtual network has greatly reduced latencies and higher transfer rate capabilities than physically built network. However if the monitoring was deployed on a physically implemented network, the results could actually be more accurate. Then the effect of the simulations would be more easily observable because of the reduction in performance. The latencies measured from the network could be affected by load more easily since the physical network performs worse than virtual one. Switching into physical network would cause the need of monitoring to be deployed in a way where the network infrastructure would allow the monitoring to see all the traffic in the network.

VI. Conclusion

In this paper, we have presented a monitoring approach for providing increased confidence in the security and reliability of remotely accessed services. While it is not a silver bullet, it helps in providing increased confidence in the service operation and to mitigate damage from observed issues. The presented approach is based on six specific steps starting from risk analysis and ending in information synthesis and presentation, and makes use of four types of monitoring information. The application of the approach in a mobile payment case study was used to illustrate the approach in practice.

Results of our case study show a benefit in maintaining a security situation aware monitoring and using it when determining the current capability of the service. The used simulation scenarios can be more refined to show a higher or lower advantage and the monitored security policy can be further developed. While our use of a virtualized environment is slightly different from a typical situation in the mobile payment domain, it can also be taken to provide insights into the increasingly virtualized domain of services and cloud computing of today.

References

[1] F. Sabahi and A. Movaghar, “Intrusion detection: A survey,” in *3rd International Conference on Systems and Networks Communications, 2008. ICSNC ’08.*, oct. 2008, pp. 23 –26.

[2] G. Shen, D. Chen, and Z. Qin, “Anomaly detection based on aggregated network behavior metrics,” in *International Conference on Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007.*, Sept. 2007, pp. 2210 –2213.

[3] H. A. Nguyen, T. Tam Van Nguyen, D. I. Kim, and D. Choi, “Network traffic anomalies detection and identification with flow monitoring,” in *5th International Conference on Wireless and Optical Communications Networks, 2008. WOCN ’08.*, May 2008, pp. 1 –5.

[4] H. Salehi, H. Shirazi, and R. Moghadam, “Increasing overall network security by integrating signature-based nids with packet filtering firewall,” in *International Joint Conference on Artificial Intelligence*, April 2009, pp. 357 –362.

[5] M. Fossi, D. Turner, E. Johnson, T. Mack, T. Adams, J. Blackbird, S. Entwisle, B. Graveland, D. McKinney, J. Mulcahy, and C. Wueest, “Symantec Global Internet Security Threat Report–Trends for 2009,” Technical Report XIV, Symantec Corporation, Tech. Rep., 2009.

[6] A. Herzog, N. Shahmehri, and C. Duma, “An Ontology of Information Security,” *International Journal of Information Security and Privacy*, vol. 1, no. 4, pp. 1–23, 2007.

[7] E. Bulut, D. Khadraoui, and B. Marquet, “Multi-agent based security assurance monitoring system for telecommunication infrastructures,” in *Communication, Network and Information Security*, July 2007, pp. 1 –5.

[8] B. Marquet, S. Dubus, and C. Blad, “Security assurance profile for large and heterogeneous telecom and it infrastructures,” in *The 7th International Symposium on Risk Management and Cyber-Informatics: RMCI 2010*, July 2010, pp. 1 –5.

[9] Z. Yong, T. Xiaobin, and X. Hongsheng, “A novel approach to network security situation awareness based on multi-perspective analysis,” in *International Conference on Computational Intelligence and Security, 2007*, 15-19 2007, pp. 768 –772.

[10] F. Lan, W. Chunlei, and M. Guoqing, “A framework for network security situation awareness based on knowledge discovery,” in *2nd International Conference on Computer Engineering and Technology (ICCET), 2010*, vol. 1, 16-18 2010, pp. V1–226 –V1–231.

[11] S. Mohammadi and H. Jahanshahi, “A study of major mobile payment systems’ functionality in europe,” in *11th International Conference on Computer and Information Technology, 2008. ICCIT 2008.*, dec. 2008, pp. 605 –610.

Toward Engineering of Security of Information Systems: The Security Acts

Wilson Goudalo
ABE Research
Advanced Business Engineering – ABE
Paris, France
wilson.goudalo@laposte.net

Abstract: **Business professionals and researchers have made considerable efforts and significant technical breakthroughs in information security in the last decades. Nevertheless, companies and organizations continue to incur losses associated with security issues. In order to remedy to this situation, we propose a new approach to information security engineering for companies and organizations. First, this approach is based on the standards and good practices of security, second, is inspired from the best practices and feedback of advances in the engineering of enterprise information systems security, and third, its design takes advantage of more than twelve years of experience in system architecture and information security for reknown banks and financial institutions.**

Our approach to engineering of information systems security aims at:

- reducing losses relating to security issues in companies and organizations, operating on an enhanced and sustained information security;

- improving the reliability of processes in companies and organizations, and assisting companies in legal and regulatory compliance efforts, operating on security indicators and checkpoints at various levels of management;

- helping companies gain competitive advantages through their security management solutions, operating on a global security monitoring system with feedback.

As further development of the basic principle of Security know-how Encapsulation into UML profiles [14], we have introduced the mapping global picture of the Process of Security engineering into the formalism of Business Processes. The purpose of this paper is to provide a clear methodology based on the elaboration of the key Security Acts of the process of information systems security engineering. The paper consists of three major parts:

- Part One recalls the reasons why BPM has been chosen for our process of system information security.

- Part Two develops the key security acts of the process of information systems security engineering.

- Part Three shows some security metrics to illustrate the aims of our works.

Keywords: security acts; security engineering; BPM; enterprise information system security.

I. INTRODUCTION

Business professionals and researchers have made considerable efforts and significant technical breakthroughs in information security [1] - [7]. Nevertheless, companies and organizations continue to incur losses due to security issues [8] – [11]. Taking this unsatisfying situation into account, we have developed a new approach to information security engineering for companies and organizations. Our works aim at bringing a methodological and organizational approach to bond all the stakeholders of a company around the security issue. We use a federator approach based on Business Process Management and provide a real bridge between the enterprise top management level and the daily technical operations level.

Nowadays, business is incorporated in technology. Business security depends on technology security on the one hand side and, on the other hand side, technology security improves business. To achieve an efficient enterprise security, we propose a joint-work of the different teams of a company: the management, business, functional and technical teams [12]. Our approach is reinforced by the McKinsey strategy [13]: *"When business and IT executives jointly take an end-to-end look at business processes, the resulting investments can have up to ten times the impact (to the business) of traditional IT cost reduction efforts"*.

In our recent articles, we showed the encapsulation of Security know-how into UML profiles [14] and introduced the global view on mapping the Process of security engineering into the formalism of Business Processes [15]. The purpose of this paper is to give detailed information on the key security acts of the process of information systems security engineering.

II. BPM NOTIONS AND SECURITY CONCEPTS

Business Process Management is activity undertaken by businesses to identify, evaluate, and improve business processes. A business process is a set of activities organized in a network and performed sequentially or in parallel that combine and implement multiple resources, capabilities and skills to produce a valuable result or output.

The business process is the most important asset of a company [16]. It is the first step of an enterprise strategy, coming before the other business functions like the functional, the applicative and the technical or technological views [17]. All enterprise architecture methodologies deal with this paradigm: the Praxeme approach [18] relates to the pragmatic and semantic aspects, the Togaf approach [19] to business architecture (Architecture Development Methodology) and business scenario; the USI approach ([20] and [21]) relates to the business view and the Zachman approach ([22], [23] and [24]) relates to the cells « Function/Scope» and « Function/Enterprise Model ». All the other points of view (levels, layers, aspects) of the information system will gradually be developed from the process point of view ([25], [26], [18] and [27]) regardless to the methodology adopted by the company (Top-Down approach) in order to come down to a concrete implementation of the security measures.

Our approach to security engineering covers the security of information systems at each level of their life cycle, by working on four security concepts simultaneously in an attempt of an on-going enhancement. These are the assets and related risks and the security solutions and monitoring indicators.

The security concept of assets describes the main company assets to protect and their value in terms of security. A company or organization asset would then be described as the set of all its properties having value which are necessary to reach its business objectives. It could be information, services provided to clients or partners, transformation processes, know-how and skills inherent to the activity of the company and having value with respect to the stakes of the company. An IS asset is an IS component which supports one or many company assets; therefore the company assets security requires the security of the IS assets supporting them.

The security concept of risks is linked to the security needs of the company assets. Security needs are expressed through three basic criteria: confidentiality, integrity and availability. Security risk depends on the exposure of company assets to risks, the probability of occurrence of a security-relevant event and the actual resulting damage to the assets. The different components of security risk are: the risk itself, the risk factor, the risk impact, threat and vulnerability.

The security solution defines the measures taken to protect company assets against risks, to which they are exposed. When elaborating a security solution, the decision about how an identified risk must be processed, would be to avoid, reduce, transfer or retain that risk.

A solution in itself is a function of the factors listed hereafter: the security policy in force, the asset quotation, the results of the analysis of the identified risk, the nature of the solution and how the latter is designed, implemented, run and managed.

The security concept of monitoring indicators is defined through security quality, in order to reach harmonized monitoring of security efficiency. Quality is evaluated on the basis of all the functional and non-functional requirements [28]. It is an abstract concept whose meaning would be different from one stakeholder to the other (e.g. clients, partners, users, managers, designers, developers and operators). It refers to different concepts depending on the qualified object. To define the security quality requirements concretely, we divided the main concept into some kinds of conceptual quality model [15].

III. Security acts of Information Systems

Information systems security engineering is defined as a process that aims at providing global security to enterprise information systems in their eco-system, in order to meet the company stakes. We present below the new design of the global picture of the information security engineering process.

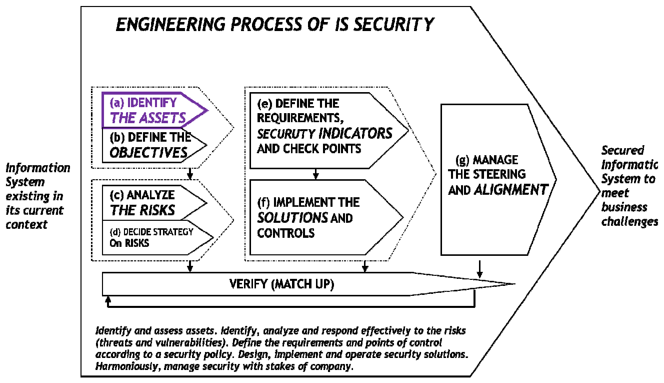

ENGINEERING PROCESS OF IS SECURITY
Information System existing in its current context
(a) IDENTIFY THE ASSETS
(b) DEFINE THE OBJECTIVES
(c) ANALYZE THE RISKS
(d) DECIDE STRATEGY On RISKS
(e) DEFINE THE REQUIREMENTS, SECURTY INDICATORS AND CHECK POINTS
(f) IMPLEMENT THE SOLUTIONS AND CONTROLS
(g) MANAGE THE STEERING AND ALIGNMENT
VERIFY (MATCH UP)
Identify and assess assets. Identify, analyze and respond effectively to the risks (threats and vulnerabilities). Define the requirements and points of control according to a security policy. Design, implement and operate security solutions. Harmoniously, manage security with stakes of company.
Secured Informatic System to meet business challenges

Figure 1. Process of ESIS – the security acts

The security acts of our security engineering process are:

- (a) Identify the company assets.
- (b) Define the security objectives to be met.
- (c) Analyze the security risks.
- (d) Decide strategy on security risks.
- (e) Define the security requirements to cover the risks.
- (f) Select, implement security solutions and verify control tools.
- (g) Manage security and its alignment to companies' objectives.

After the security solutions have been defined, new security risks may occur. The latter might render some security solutions obsolete or might be ignored by the security solutions already implemented. The process of information systems security engineering must be iterative and cyclic to achieve an on-going enhancement. This guarantees the efficient fulfillment of the primary goals of companies' security at any time. For this purpose, we added a transversal security act to insure the auto-adaptation of the process; this is called "Matching". This security act does not exist on its own; it goes together with the security act "Management of security and its alignment with the company objectives" (g).

We split up each security act into a set of security activities and present it according to the business sub-processes formalism.

A. *Identify the company assets*

"Identify assets of companies" is one of the seven security acts of our process of information systems security engineering. In this security act, we include the following security activities:

- List the major assets (such as hardware, software, human, documentary, physical and intangible).
- Specify the use case contexts and the corresponding responsible person (the person who has deep knowledge of its use, its value, and the consequences of compromise) for each asset.
- Define the values held by those assets and their sensitivity to business enterprise, regulation and legislation
  Classify assets and define their quotation, in relation with given security criteria.

We illustrate below the security act "Identify the assets of company", as a sub-process.

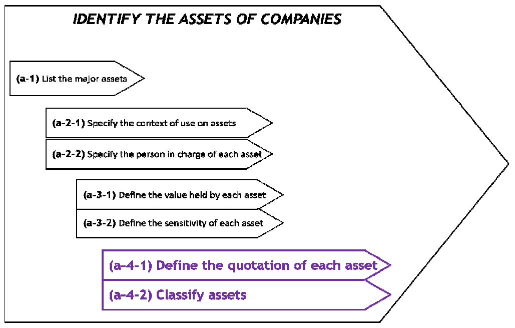

IDENTIFY THE ASSETS OF COMPANIES
(a-1) List the major assets
(a-2-1) Specify the context of use on assets
(a-2-2) Specify the person in charge of each asset
(a-3-1) Define the value held by each asset
(a-3-2) Define the sensitivity of each asset
(a-4-1) Define the quotation of each asset
(a-4-2) Classify assets

Figure 2. Process of ESIS – the security activities as sub-processes

We recall the characteristics of information system security defined by the three invariant criteria below.

- Confidentiality: Information should not be, made accessible, or disclosed to a user, to an entity or to a system process that is not allowed.
- Integrity: Information should not be amended, altered or destroyed in an unauthorized manner.
- Availability: Access, by an authorized entity, authorized user or authorized process, to services provided by the system, must be always possible. Operations that occupy processing time illegally or that attempt to reach such goal, must be detected and eliminated in time.

Other properties of the information security, such as Proof, Traceability and Authenticity are derived from these three invariants criteria.

The security criteria characterize the constraints on the properties of business assets, describing their security needs. The stakes of companies may be human, financial, branding, regulatory or legal.

We illustrate below the process of quotation those results in secure assets classification.

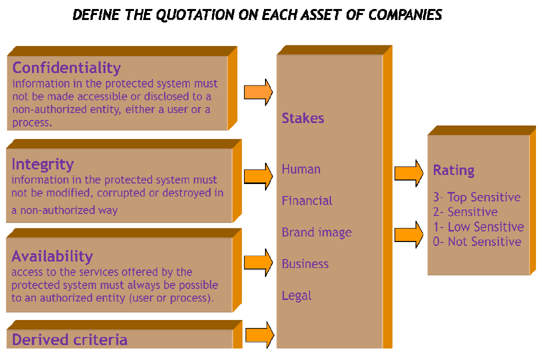

DEFINE THE QUOTATION ON EACH ASSET OF COMPANIES
Confidentiality
information in the protected system must not be made accessible or disclosed to a non-authorized entity, either a user or a process.
Integrity
information in the protected system must not be modified, corrupted or destroyed in a non-authorized way
Availability
access to the services offered by the protected system must always be possible to an authorized entity (user or process).
Derived criteria
Stakes
Human
Financial
Brand image
Business
Legal
Rating
3- Top Sensitive
2- Sensitive
1- Low Sensitive
0- Not Sensitive

Figure 3. Process of ESIS – the quotation of assets

B. *Classify the company assets*

It's not possible to secure every asset of a company against every imaginable risk. Thus, we must classify assets on which we operate, in order to efficiently protect the intellectual property, protect confidential information from unauthorized use or disclosure and facilitate SLA (Service Level Agreements) and business continuity management. Security classification of assets becomes necessary to provide and improve business activities.

Security classification of assets meets both business and operational needs. It is based on the real value of the assets for companies in terms of business, brand image, human resources, and financial, legal and regular aspects. Actually, we dissociate the security classification of assets from the activities related to threat and risk analysis. In our process of information system security engineering, the security activity "security classification of assets" is an element of security act "Identify assets". The results of the three security acts “Identify assets” “Define the objectives of security” and “Analyze the risks” are used in the security act “Decide strategy on risks”.

We illustrate below the main steps of the security activity “Secure classification of assets”, as a sub-process.

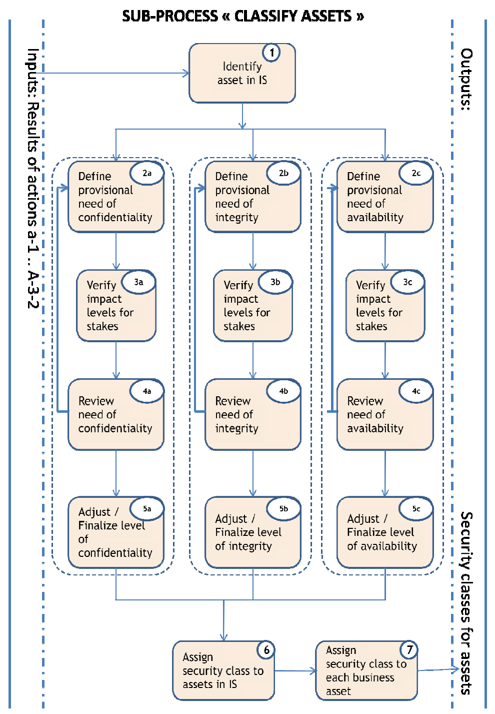

SUB-PROCESS « CLASSIFY ASSETS »
Inputs: Results of actions a-1…A-3-2
Outputs:
Security classes for assets
1 Identify asset in IS
2a Define provisional need of confidentiality
2b Define provisional need of integrity
2c Define provisional need of availability
3a Verify impact levels for stakes
3b Verify impact levels for stakes
3c Verify impact levels for stakes
4a Review need of confidentiality
4b Review need of integrity
4c Review need of availability
5a Adjust / Finalize level of confidentiality
5b Adjust / Finalize level of integrity
5c Adjust / Finalize level of availability
6 Assign security class to assets in IS
7 Assign security class to each business asset

Figure 4. Process of ESIS – the sub-processes of security classification of assets

The security classification of assets

- Operates on three main security criteria (Confidentiality, Integrity, Availability) and derived criteria;
- Takes into account companies stakes (from strategic management);
- Provides four sensitivity classes (non sensitive, low sensitive, sensitive and top sensitive).

We descript the sub-process “classify assets of company” in seven steps, and we elaborate the input and output of each step.

Step-1: Identify asset in the Information System of company. For each asset defined in asset list (business asset or essential asset to support the business), we identify the related IS assets, according to conceptual model of company assets [15], and we specify (verify) the owner of each asset.

Step-2: Define provisional level, about need of security criteria Define security needs for the related IS asset, in terms of confidentiality (a), integrity (b) and availability (c), according to the real (intrinsic) value of each asset.

Step-3: Verify the impact levels for stakes of company. Verify the impact levels for stakes of company, in terms of human, financial, brand image, business and legal, according to the organization, environment and mission of a company.

Step-4: Review provisional level, about need of security criteria. Review the appropriateness of the provisional need of security criteria (confidentiality, integrity and availability), according the level of impacts assessment on stakes of company.

Step-5: Adjust/Finalize level, about need of security criteria. Adjust/Finalize need of security criteria (confidentiality, integrity and availability) to each asset in IS, according to security strategy of the company, the real environmental context of the company, and the security policy in force in the company.

Step-6: Assign security class to assets in IS. Assign the resulting security classes to main assets in IS, and determine the classification.

Step-7: Assign security class to each business asset. Assign the resulting security class to main business assets according the classification determined in step-6, and document the classification in a language understandable by the business.

We recall that security engineering is assumed through ongoing process, like Deming wheel [14]. So it is necessary to maintain classification and conduct continuous review.

We synthesize the provided four classes of sensitivity, in the table below.

TABLE 1. SECURITY CLASSIFICATION OF ASSETS

Security classes	Description	Examples of assets *(*)*
Class-0: Not sensitive	Assets that enter in the normal course of business or support of business, and that don't present any stake for a company in case of loss of confidentiality, integrity, availability, or derived criteria.	Provided courses, planning of courses, dates of exams, professional information on organizers.

Security classes	Description	Examples of assets *(*)*
Class-1: Low sensitive	Assets that enter in the normal course of business or support of business, and that present low stake (low levels of financial loss) for a company in case of loss of confidentiality, integrity, availability, or derived criteria.	Professional information on students, detailed course material, earnings per course.
Class-2: Sensitive	Assets that enter in the normal course of business or support of business, and that present serious stake (important level of financial loss, loss of competitive advantage, loss of confidence in business strategy, damage to partnerships, relationships and reputation) for a company in case of loss of confidentiality, integrity, availability, or derived criteria.	User access of students, examination subjects, enlistment strategy of students.
Class-3: Top sensitive	Assets that enter in the normal course of business or support of business, and that present vital stake (extreme damage to the integrity, effective service delivery, loss of life, substantial financial loss, major economic impact) for a company in case of loss of confidentiality, integrity, availability, or derived criteria.	User access of organizers, personal medical record of students, online exam service.

(*) – In order to comply with the confidentiality clause of our business agreements with customers, we have used a fictive e-Learning company as basis to build all the examples presented here.

We elaborate the security classification of assets according to their value and importance for the target companies. We assign to a handled group of assets, at least the highest security class level of all assets forming the (no dissociable) group. In the case of some combination of assets, the real security class of the resulting combination must be determined. The security classification of an asset is adjustable depending on its life cycle and on its participation in the business cycles in a company.

For each security class we develop a set of procedures for storage, copying, access, transmission, communication, disclosure, destruction and accountability.

C. *Define the security objectives to be met*

In defining the security objectives, we must,

- Express the security needs on the assets, for the first time.
- Establish the specifications appropriate to the triptych: “identified security needs”, “analyzed security risks” and “security policy of the company”.

D. *Analyze the security risks*

In this security act, we mention the following activities:

- Identify the inherent vulnerabilities of each asset that expose it to potential threats
- Identify threats to which each asset is exposed
- Estimate the probability of occurrence of each threat (ignoring the context and the environment).
- Analyze the impact of the occurrence of each threat.
- Estimate the likelihood of occurrence of each threat (taking into account the facts, the existing measures, and environment).
- Define the probability of occurrence, depending on the context.
- Assess the overall level to which each asset is exposed, taking into account all evidence obtained above.

There are different methods of risk analysis. We have developed the key security activities that are in use in the industries.

E. *Decide strategy on security risks*

In order to make a decision about the treatment of security risks, we must, for the first time, decide which type of treatment to apply for each identified risk. The decision of treatment belongs to one of these four types: acceptance, avoidance, reduction, transfer.

After a strategy has been adopted, we deduce the residual risks.

If the inferred residual risks are *not acceptable, we make another decision about the type* of risk treatment. In the general case, the act of deciding which treatment of risk to adopt leads to the reasonable decision of reducing the risk. This results in the definition of the security requirements, to mitigate each risk.

F. Define the security requirements to cover the risks

The security requirements lead to define the type of controls: Correction, Detection, Deterrence and / or Prevention. We insist on the consistency of security requirements with the security objectives and with the treatment strategy. We recall that the security objectives are themselves consistent with the risks, needs and security policy.

The security requirements cover all security objectives and define two types of requirements: functional requirements (security features to provide) and the associated insurance requirements (evidence on the quality of features).

G. Select, implement and verify control tools

In this security act, we first choose the security measures to meet the security requirements. On the next step, those measures are implemented, tested, integrated deployed.

Security measures can be physical, organizational and / or software. They are kind of products from suppliers or specific developments.

In all cases, this security act is the sound basis on which the IS security stands. Most approaches to security and security research are limited to this security act.

H. Manage of security and its alignment to company objectives

In this security act, we mention the following activities:

- Develop a document of the security policy in accordance with the corporate objectives, in its ecosystem competitive, regulatory and legal.
- Decline of the security policy features in security and safety indicators qualify, quantifiable and verifiable.
- Review regularly (at intervals of time defined) the security policy and occasionally (not compulsory and optional) in case of significant events.
- Ensure the harmony between the different activities of security (from “a” to “g”) and make them comply with the security policy.

IV. Illustration - Security Metrics

In a Top-Down approach (from strategic management level to technical daily operations level), applying the methodology in place in the company to the process of Information System Security Engineering will lead to the concrete implementation of security solutions. We suggest below an architectural diagram which illustrates the concrete implementation of security solutions and the control points providing security metrics.

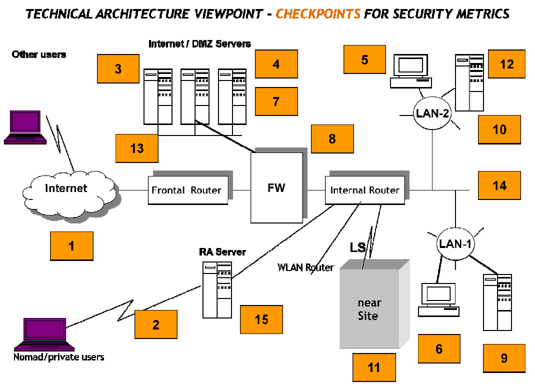


Figure 5. Process of ESIS – the technical implementation diagram

Security metrics are necessary to manage strategic alignment. The top management is interested in:

1. Corporate Reports as money, ratios, index
2. Measures of broad matters as quality compare to that of competitors; time required to launch new products
3. Measures that help to establish departmental quality goals and to evaluate departmental performance against goals.
4. Technological units of measure for individual elements of product, process, service

The main objective is “Moving up the stack without losing clarity”. Our approach of Engineering of Information Security applied with Enterprise Architecture framework in place provides the real solution.

V. Conclusion

To stay under the admissible size of this article, we will develop the other security activities related the seven security acts (from “a” to “g”) in future works,.

In this paper we have presented the security activities related to assets classification in more details. The preference of security activity “Classify assets” is motivated by the nowadays period to which companies are facing.

After the era of the extended enterprise where the information system was opened to customers and partners, we are witnessing two phenomena, simultaneously: “the consumerism of computing” and “the cloud computing”. With "the consumerism of computing" era, we observe the import innovative uses from home to work, the blurring the boundary between home and work, the mutual influence between technology and uses. With “the Cloud Computing”, PAAS, IAAS, SAAS, and others, the IS assets of enterprise are no more storage in well known geographic place. In these contexts the perimeter based security is obsolete. The challenge is in assisting. The CSO has to get closer to the core business of a company. While having an adaptive communication to the management and to all levels of the company, he also has to understand the strategy, and the technical applications.

The CSO classifies the information of the companies and their assets. So in this context, access to information assets is defined according to:

- Who is the person.
- The trust level of the terminal.
- Where is the terminal located.

The access level to asset depends on the classification of asset and the trust level of these three parameters.

VI. References

[1] David Elliott Bell and Leonard J. LaPadula. “Secure Computer Systems : Unified Exposition and Multics Interpretation”. Technical Report ESD-TR-75-306, MTR-2997, MITRE, Bedford, Mass, 1975.

[2] Morrie Gasser. “Building a secure computer system”, Van Hoostrand and Reinhold ed., 1988.

[3] Charles Bennett. “Quantum Cryptography – Uncertainty in the Service of Privacy”. Science Vol. 257. no. 5071, pp. 752 - 753. August 1992.

[4] Fredéric Cuppens. “A Logical Analysis of Authorized and Prohibited Information Flows”. IEEE Symposium on Security and Privacy. Oakland, 1993.

[5] Richard J. Hughes, Jane E. Nordholt, Derek Derkacs and Charles G. Peterson. “Practical free-space quantum key distribution over 10 km in daylight and night”. New journal of physics 4 (2002). http://www.iop.org/EJ/abstract/1367-2630/4/1/343/

[6] William Stallings. “Cryptography and Network Security – Principles and practice”. Fourth edition. ISBN 978-0-1318-7316-2. Prentice Hall Editions. New Jersey - USA 2006.

[7] SANS (SystAdmin, Audit, Network, Security), “Rapport 2009 CWE/SANS des 25 erreurs de programmation les plus dangereuses”, SANS 2009. http://www.sans.org/top25errors/ Last access date : may 2011.

[8] Security for Business Innovation Council; “The Time is now: making information security strategic to business innovation”; RSA Security; Bedford MA. 2008.

[9] Benoît Dupont et Bénoît Gagnon. “La sécurité précaire des données personnelles en Amérique du Nord", Note de recherche n°1, Chair de recherche du Canada en sécurité, identité et technologie, 2008.

[10] http://www.cnis-mag.com
Site de magasine sur les problématiques de l’informatique et de la sécurité. Last access date : may 2011.

[11] Jeremy Epstein, "Security Lessons Learned from Société Générale", IEEE SECURITY & PRIVACY, Vol. 6, 03, pp. 80-82, MAY/JUNE, 2008.

[12] Goudalo Wilson, “Business Security” in Engineering Secure Complex Software Systems and Services. Proceedings of the European Research Consortium for Informatics and Mathematics Seminar on ICT Security. Brussels, 16 October 2008. http://www.ercim.org/

[13] “Managing IT in a Downturn - Beyond Cost Cutting”. McKinsey on Business Technology. Fall 2008

[14] Goudalo Wilson et Seret Dominique. "Toward the Engineering of Security of Information Systems (ESIS): UML and the IS Confidentiality". Proceedings of the 2008 Second International Conference on Emerging Security Information, Systems and Technologies. Pages 248-256. IEEE Computer Society Washington, DC, USA. ISBN: 978-0-7695-3329-2.

[15] Wilson Goudalo et Dominique Seret. "The Process of Engineering of Security of Information Systems (ESIS): The Formalism of Business Processes ". Securware, pp.105-113, 2009 Third International Conference on Emerging Security Information, Systems and Technologies, 2009.

[16] Marwane El Kharbili, Sebastian Stein, Ivan Markovic and Elke Pulvermüller. “Towards a Framework for Semantic Business Process Compliance Management”. Proceedings of the Workshop SBPM 2007, Innsbruck 2007, ISSN 1613-0073

[17] Ivan Markovic, Alessandro Costa Pereira. “Towards a Formal Framework for Reuse in Business Process Modeling”. In Workshop on Advances in Semantics for Web services (semantics4ws), in conjunction with BPM '07. Brisbane (Australia) 2007.

[18] Praxeme institute – Public methodology of Enterprise Architecture. http://www.praxeme.org. Last access date : may 2011.

[19] TOGAF, The Open Group Architecture cadre. Version 8.1 “Enterprise Edition” 2003.

[20] Longépé Christophe. “The Enterprise Architecture IT Project – the Urbanisation Paradigm”. Penton, 2002

[21] Club Urba SI. "Pratiques de l’Urbanisme des Systèmes d’Information en entreprises". Publibook, 2003.

[22] John Zachman and John Sowa. “Extending and Formalizing the Framework for Information Systems Architecture”. IBM Systems Journal. Volume 31, No. 3, pp 590-616, 1992.

[23] Jaap Schekkerman. “How to Survive in the Jungle of Enterprise Architecture Frameworks”, Trafford, Third edition, 2006

[24] Tony Brown. “The Value of Enterprise Architecture”. ZIFA report, 2005.

[25] Douglas W. McDavid. "A standard for business architecture description" in Enterprise Solutions Structure. IBM Systems Journal, Volume 38, Number 1, 1999

[26] Celia Talma Martins and António Lucas Soares. “Dissecting Inter-Organizational Business Process Modeling: A Linguistic and Conceptual Approach” in Network-Centric Collaboration and Supporting Frameworks. pp 221-228. Springer Boston, 2006. ISBN : 978-0-387-38266-1.

[27] [GAN-08] Eswar Ganesan, Ramesh Paturi. “Bulding Blocks for Enterprise Business Architecture”. Infosys Research Publications. SETILabs Briefings. Volume 6, Number 4, 2008.

[28] Beatriz. Bernardez, Amador Duran, and Marcela Genero. “Metrics for use cases: a Survey of Current Proposals”. In M. Genero, M. Piattini, and C. Colero Editors, Metrics for Software Conceptual Models. Pages 59-98. Imperial College Press, 2005.

Establishing Authentication Trust in Open Environment Using Social Approach

Hidehito Gomi
Yahoo! JAPAN Research
Yahoo! Japan Corporation
Tokyo, Japan
hgomi@yahoo-corp.jp

***Abstract*—A trust metric is described for a user to ensure the authenticity of another user who is not known to system entities in an open environment. On the basis of the metric, an identity federation framework is proposed for propagating an authentication assertion for an unknown user across system entities. The unknown user directly interacts with an authenticating user with the support of an entity mediating the authentication. By use of the proposed framework, an entity receiving an authentication assertion can derive and evaluate the trust value of its corresponding user in a quantitative fashion to flexibly control his or her access.**

***Keywords*-trust metric;identity federation;delegation**

I. INTRODUCTION

Conventional security systems usually need to authenticate users to control their accesses to restricted services. For this, the users are required to register their personal information and obtain credentials to be used for authentication. This is a common procedure and is a convenient way to clarify which entity is responsible for authorizing accesses. However, a user who would like to use a service but cannot be authenticated by the service often becomes unwilling to use the service because the process of registering personal information is troublesome or time-consuming. Namely, there is a trade-off between flexible service provisioning and secure access control.

A use case requiring flexible service provisioning is delegation, in which a user provides all or some of his or her privileges to another user to perform tasks. In delegation, the complexity of the system increases because the user attempting to execute a task (delegatee) is different from the user who already has privileges to perform the task (delegator) but who delegates the task. When a service provider (SP) does not have any information about the delegatee, access control based on his or her identity becomes impossible even if the SP knows about the delegator delegating the task to the delegatee. This situation occurs in many scenarios for open and distributed applications. If the SP obtains a certificate containing information about the delegatee issued by a trusted entity, the SP may possibly grant the delegated access by evaluating the trustworthiness of the delegatee's identity.

Much work on the exchange of security- and privacy-related information in terms of identity management has been conducted. Recently emerging technical specifications [1]–[3] provide a framework for federated identity management (FIM) systems. In this framework, an identity provider (IdP) authenticates a user by means of particular authentication methods and issues an authentication assertion about the user to an SP that provides a restricted service to authorized users. Although this framework enables propagation of information about an authenticated user based on trust between an IdP and SP, it still cannot propagate information about a user who has not been registered at the IdP, which is the same as in conventional security systems. In many delegation scenarios, a delegator trusts in a delegatee and can recognize his or her identity based on many criteria. Therefore, the information that a delegator has on a delegatee can effectively be shared for an SP to authorize the delegatee's access even if an IdP cannot authenticate him or her directly.

A trust metric for a user to authenticate another user who has not been registered in the system is proposed, as is the metric's framework for propagating the authentication information among the system entities, based on the author's previous work on FIM systems [4]. The framework introduces a specific authentication federation method by which a direct interaction between the two users can be reflected in the authentication assertion with the support of an entity that mediates the authentication. With the framework, an SP receiving the assertion about the unregistered user can flexibly control his or her access in a quantitative fashion.

The rest of this paper is organized as follows. Section II reviews related work. Section III introduces an authentication trust metric for FIM systems. Section IV proposes a user-centric authentication federation scheme. Section V describes the derivation of the authentication trust value for an entity calling a resource on behalf of its owner. Section VII concludes the paper and presents future work.

II. RELATED WORK

Trust models, metrics, and formalization have been frequently researched. Prior work on general trust has focused on defining the semantics of trust and modeling trust-based systems. A variety of trust classes, trust types, and reputation systems [5] has been investigated.

Existing work on trust formalization focuses on assigning numerical values of trustworthiness to paths representing relationships between entities [6]–[8]. Beth et al. [6] presented a formal representation of trust relationships and the algorithms for deriving them to estimate the trustworthiness of entities in open networks. Reiter and Stubblebine [7] presented a set of guiding principles for the design of authentication metrics. Huang and Nicol [8] introduced a formal representation of trust in a public key infrastructure (PKI) and proposed a mechanism for quantifying trust in certificate chains. However, their models focused on general trust and do not deal with user-to-user authentication.

Work related to identity and trust management can also be found in the literature [9]. Thomas et al. [10] defined the semantics of the authentication trust level and provided a method for combining two trust levels of a multifactor authentication in a FIM environment. Although their work shares the author's goal of flexible and quantitative access control, it did not consider a social authentication approach as proposed in this work. Gomi [4] proposed an authentication trust metric for FIM systems. The author enhances his approach for propagating the trust level of a person to a more general framework for controlling access by an unregistered user by using social authentication in an open environment.

Another related line of research is end-to-end trust establishment methods in limited network environments. Seigneur et al. [11] proposed an entity recognition approach in which dynamic enrollment enables spontaneous interactions with unknown entities in pervasive computing environments. Theodorakopoulos and Baras [12] proposed a method for evaluating trust between entities in ad-hoc networks. The limitation of these methods is that they do not support the derivation of authentication results among entities.

III. Authentication Trust for Identity Federation

This section describes the semantics and the formal representation of authentication trust for FIM systems [4].

A. Trust Semantics

The model's trust relationships are defined as follows.

Definition 1 (Identity Trust). *Identity trust is the certainty that the identity of an entity is identical to the identity claimed by the entity itself or by other parties regarding the entity on the basis of the authentication context.*

In the above statement, *authentication context* [13] denotes the information about the characteristics of the mechanisms and processes by which the authentication confirms the entity's identity. For example, the information includes authentication methods such as presentation of a password over a protected transport channel and verification of a digital signature using an X.509 certificate. The authenticating entity validates the presented credential and determines the trustworthiness of the entity with some level of certainty depending on the above authentication context.

Identity trust is a foundation for authorizing an interacting entity to access restricted resources or for regulating interactions with the entity in trust-based systems, including FIM systems. Accordingly, the semantics of identity trust can be defined as follows:

$$trust^{(i)}_{pq}(x) \equiv authn(p,q,x) \qquad x \in AC, \quad (1)$$

where AC stands for a set of information about the authentication context, $trust^{(i)}_{pq}(x)$ expresses that entity p has identity trust in entity q on the basis of a specific authentication context x, and $authn(p,q,x)$ means that entity p authenticates entity q by means of an authentication method represented in an authentication context x. This axiom represents a practical procedure for entity authentication in FIM systems associated with the identity trust relationship in this trust model.

Many metrics for evaluating this trust relationship have already been proposed. For example, many PKI trust models focus on the relationships among certification authorities for X.509 certificates. In such a *credential-focused* system, long-term, non-transitive cryptographic credentials such as X.509 are issued without involving an IdP [14].

In contrast, FIM systems are *relationship-focused* ones in which an online IdP dynamically issues short-term security tokens while restricting its transitivity on the basis of the relationships between the issuer (IdP) and recipient (SP). In this paper, a formal representation of authentication trust by introducing the above trust relationship is examined. The following definitions are related to the identity-trust-deriving capabilities that are specific to FIM systems.

Definition 2 (Attestation Trust). *Attestation trust is the certainty about an entity's capability to accurately create and assert information necessary for a recipient in a format appropriate for the recipient and to securely transmit the information to the recipient.*

On the basis of the above definition, if $trust^{(a)}_{pq}(x)$ designates attestation trust by trustor p in trustee q regarding information x, its semantics are given in first-order logic as

$$trust^{(a)}_{pq}(x) \equiv assert(q,x) \Rightarrow accept(p,x), \quad (2)$$

where $assert(q,x)$ means that q creates an assertion containing information x and $accept(p,x)$ represents p accepting that x is true. The $\Rightarrow$ operator designates the implication that whenever the antecedent (expression to the left of the operator) is true, the consequent (expression to the right of the operator) is true.

In the above axiom, x is general propagated information. However, if x corresponds to an authentication context, the

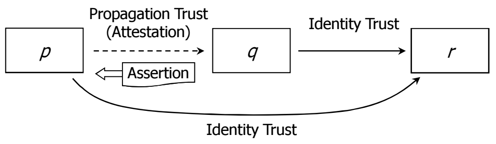


Figure 1. Authentication trust transitivity.

axiom specifically denotes that the authentication context is propagated from q to p in a particular encoded format. This leads to the establishment of authentication trust as a basic principle in FIM systems.

B. Authentication Trust Derivation

For derivation of user authentication, *propagation trust* is defined as follows.

$$trust_{pq}^{(i,a)}(x) \equiv trust_{pq}^{(i)}(y) \wedge trust_{pq}^{(a)}(x). \tag{3}$$

In this axiom, $trust_{pq}^{(i,a)}(x)$ denotes the attestation mode of propagation trust from p to q regarding information x that q propagates to p. This means that p has attestation trust in q regarding information x and p also has identity trust in q. Note that the authentication context propagated to p is not q's authentication context y but x.

The basic principles of trust-based systems are described in previous literature [5]. In general, trust transits from entity to entity. The idea behind *trust transitivity* is that when Alice trusts Bob, and Bob trusts Charlie, and Bob refers Charlie to Alice, then Alice can derive a measure of trust in Charlie based on Bob's referral combined with her trust in Bob. Although this principle holds true for authentication trust in FIM systems, it additionally involves end-to-end authentication procedures for deriving trustworthy information about the authenticity of an entity as well as the attesting capability of an entity propagating the information. This is illustrated in Figure 1. The solid and dashed lines indicate identity trust and propagation trust, respectively, from the viewpoint of r's authentication.

From this observation, the following rule is obtained.

Rule 1 (Authentication Trust Derivation).

$$trust_{pr}^{(i)}(x) \Leftarrow trust_{pq}^{(i,a)}(x) \wedge trust_{qr}^{(i)}(x). \tag{4}$$

This rule means that p has assurance in an authentication assertion containing x regarding r attested by q since p trusts in q's identity and attestation capability. It clearly explains a typical identity federation scenario in which p, q, and r correspond to an SP, an IdP, and a user, respectively, in FIM systems. The authentication trust in r transits from q to p in the opposite direction of the propagation of the assertion encapsulating x.

IV. User-managed Authentication

This section proposes a new scheme for obtaining authentication trust within the scope of the trust model described in Section III.

A. User-managed Authentication Trust

First, a definition for a trust semantic is given.

Definition 3 (User-managed Authentication Trust). *User-managed authentication trust is the certainty about a user's capability to authenticate another user within a particular authentication context with the support of an entity mediating the authentication (authentication mediator).*

In the above statement, "user-managed authentication" (UA) means an authentication procedure in which a user (trustor) him or herself directly authenticates another user (trustee) online. The trustor has some capability of identifying the trustee, and confirming the identity of the trustee by means of an authentication method and procedure. However, the trustor does not have any capability for attestation as defined in Definition 2, so he or she needs assistance in performing the authentication and in demonstrating the trustworthiness of the authentication results.

The authentication mediator (AM) mediates UA by providing an infrastructure for the above assistance to a trustor. The AM can provide a secure transport channel for interactions with and between a trustor and a trustee, and can monitor the interactions so that the validity of the authentication procedure between the trustor and trustee is ensured. Since the AM has some capability for attestation, it can produce and issue an assertion stipulating an authentication event within a specific authentication context regarding UA. With this scheme, an entity receiving an assertion of UA can evaluate the trustworthiness of a trustee's identity within the scope of FIM systems described in Section III.

Various types of authentication contexts for UA can be considered. Following are some examples.

Examples (User-managed Authentication Contexts).

- Secret code sharing. A trustor provides a trustee with a secret code unique to the trustee as a password that the trustee needs to present during authentication in a secure way such that it is not disclosed to other users.
- Secret questions. A trustor asks a trustee questions about information that is shared only with the trustee as a means of authenticating the trustee in a secure communication mediated by an AM. If the trustor receives answers from the trustee and accepts them as appropriate, the authentication successfully ends, with the trustor having confidence in the trustee's identity.
- Context validation. A trustor specifies the type of a trustee's context (e.g., geo-location) to an AM and then validates the appropriateness of the context information obtained by the AM using its functionality (e.g., GPS). For example, if the trustor is close to the trustee, the geo-location should indicate that the trustee is within a short distance from the trustor's current location.

As shown in the above examples, there are variations about which entity (trustor or AM) has the knowledge and

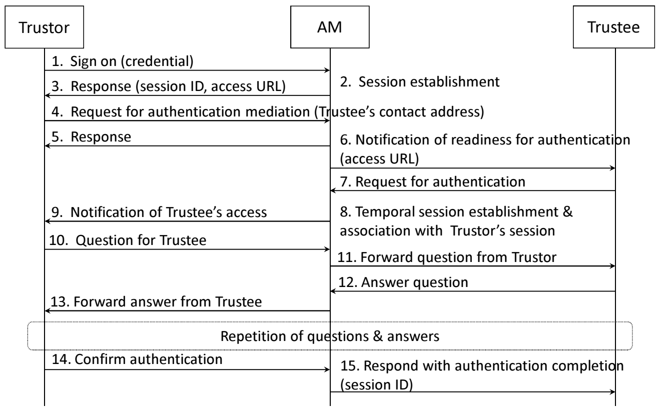

Trustor
AM
Trustee
1. Sign on (credential)
2. Session establishment
3. Response (session ID, access URL)
4. Request for authentication mediation (Trustee's contact address)
5. Response
6. Notification of readiness for authentication (access URL)
7. Request for authentication
8. Temporal session establishment & association with Trustor's session
9. Notification of Trustee's access
10. Question for Trustee
11. Forward question from Trustor
12. Answer question
13. Forward answer from Trustee
Repetition of questions & answers
14. Confirm authentication
15. Respond with authentication completion (session ID)

Figure 2. Interactions for user-managed authentication.

functionality for performing authentication and validating the procedure. In the contexts, the entity and how it validates are specified for SPs to evaluate the strength of the authentication.

B. User-managed Authentication Interactions

The following authentication interactions are considered as instances of UAs, shown in Figure 2.

1) First, the trustor signs on, presenting his or her credentials to the AM.
2) The AM establishes a session and assigns it with an identifier after validating the trustor's credential.
3) The AM returns a response including the session identifier and a URL that the trustor can access to request UA mediation.
4) The trustor sends a request for UA at the received URL attaching the trustee's contact address.
5) The AM responds in acknowledgement and tells the trustor to wait for the trustee to access the URL.
6) The AM informs the trustee that the trustor will authenticate the trustee at the access URL of the AM.
7) The trustee is given access to the specified URL for authentication.
8) When the AM receives the authentication request from the trustee, it assigns a temporary session to the trustee and associates the session with the trustor's session.
9) The AM informs the trustor of the trustee's authentication request and prompts the trustor to input his or her question for the trustee.
10) The trustor presents his or her question to the AM.
11) The AM shows the trustor's question to the trustee in the session of the trustee associated with the trustor's one.
12) The trustee sends the answer to the received question to the AM.
13) The AM forwards the received answer to the trustor. The trustor determines whether the answer is appropriate to trust in the trustee's identity.

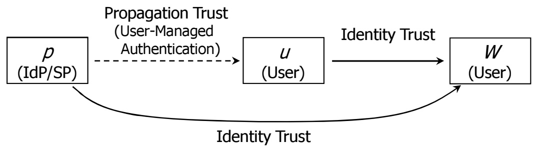


Figure 3. User-managed authentication trust.

14) The trustor obtains more confidence in the trustee's identity by repeating Steps 10–13. If the trustor has adequate assurance in the trustee's identity, the trustor notifies the AM of the authentication completion.
15) The AM updates the trustee's session so that it indicates that the trustee has a legitimate identity as claimed by the trustor.

During the above authentication interactions, the messages are transmitted in a secure communication channel. For each interaction, a specific authentication context is defined for characterizing its authentication method and strength.

C. User-managed Authentication Trust Derivation

In the authentication interactions described above, the AM is not involved with the sharing and validation of the credentials or questions. Instead, it mediates the exchange and confirmation of that information between the trustor and the trustee. Although the AM does not directly execute the procedure for authenticating the trustee, it can accept the trustee's identity as authenticated by the trustor in accordance with the AM's identity trust in the trustor, i.e., UA trust. These relationships are illustrated in Figure 3. The trust relationships among the entities shown in this figure are similar to the ones depicted in Figure 1. However, entity q in Figure 1 has an attesting capability whereas user u in Figure 3 does not have such a capability for propagating an assertion including authentication results to another party using a secure communication channel.

Let $trust_{pq}^{(ua)}(x)$ designate UA trust of trustor p in trustee q regarding authentication context x. By using this, Definition 3 is formally given as follows:

$$trust_{pu}^{(ua)}(x) \equiv authn(u, w, x) \Rightarrow accept(p, x). \qquad (5)$$

In this axiom, $trust_{pu}^{(ua)}(x)$ means that if u authenticates w in authentication context x, p accepts x.

The UA trust of p in user u naturally depends on the identity trust in u since it is from u that p receives the authentication context. Here, another propagation trust for UA from p to u, $trust_{pu}^{(i,ua)}(x)$, is defined as follows:

$$trust_{pu}^{(i,ua)}(x) \equiv trust_{pu}^{(i)}(y) \wedge trust_{pu}^{(ua)}(x). \qquad (6)$$

With this axiom, the rule of the authentication trust derivation for UA is as follows.

Rule 2 (User-managed Authentication Trust Derivation).

$$trust_{pw}^{(i)}(x) \Leftarrow trust_{pu}^{(i,ua)}(x) \wedge trust_{uw}^{(i)}(x). \qquad (7)$$

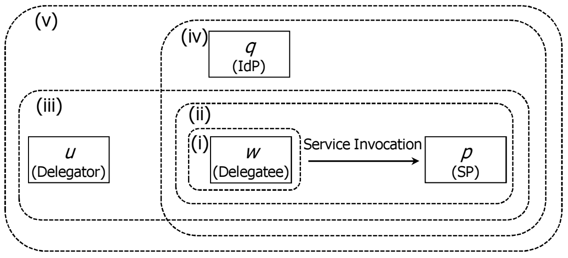


Figure 4. Combinational cases for delegatee authentication.

V. Authentication Trust Evaluation for Delegatees

This section examines the authentication trust in a delegation situation in which a delegator delegates to a delegatee the delegator's privilege to access his or her personal information. Let us assume that there are an IdP, an SP, and two users (delegator and delegatee). The IdP has an attesting capability and the SP grants or denies the delegatee's access in accordance with his or her authentication trust.

A. Authentication Trust in Delegatees

The following cases for collaboratively authenticating a delegatee are possible. They are shown in Figure 4, where the areas surrounded by the dotted lines indicate the scope of the entities involved with a delegatee's authentication. Consider the authentication trust of an SP in a delegatee for each case.

(i) **Anonymous access.** SP p does not identify delegatee w's identity because p does not have any information about w.

(ii) **Authentication by SP.** SP p successfully authenticates w directly by itself. In this case, the authentication trust in w corresponds to $trust^{(i)}_{pw}(x)$ if its authentication context x is given, as defined in (1).

(iii) **UA supported by SP.** SP p obtains UA trust in w by means of delegator u's corresponding direct authentication of w. Its authentication trust $trust^{(i)}_{pw}(x)$ is obtained using (7).

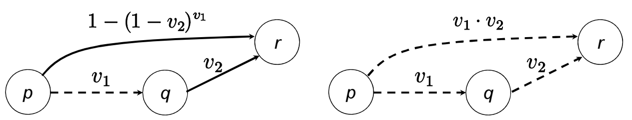
(iv) **Authentication federation between IdP and SP.** IdP q directly authenticates w and provides p with its corresponding authentication assertion related to authentication context x for w. Its authentication trust is given by (4).

(v) **UA supported by IdP plus authentication federation between IdP and SP.** First, IdP q obtains UA trust in w by means of delegator u's corresponding direct authentication of w. Its authentication trust is represented by $trust^{(i)}_{qw}(x)$ using (7),

$$trust^{(i)}_{qw}(x) \Leftarrow trust^{(i,ua)}_{qu}(x) \wedge trust^{(i)}_{uw}(x). \quad (8)$$

Then, by applying (7) and (8) to (4), the authentication trust $trust^{(i)}_{pw}(x)$ is obtained as follows:

$$trust^{(i)}_{pw}(x) \Leftarrow trust^{(i,a)}_{pq}(x) \wedge trust^{(i)}_{qw}(x). \quad (9)$$



(a) Propagation trust aggregation. (b) Direct trust aggregation.

Figure 5. Trust aggregation with Beth-Borcherding-Klein metric.

In this way, possible cases for authentication trust in a delegatee are driven using the proposed trust model for an SP to control the delegatee's access.

B. Authentication Trust Calculation

A direct authentication occurs if an entity authenticates a user in an authentication context. The value of entity p's direct authentication trust in entity q in authentication context x, i.e., $v^{(i)}_{pq}(x)$, is defined on the basis of the semantics of (1) as

$$v^{(i)}_{pq}(x) \overset{\text{def}}{=} \Pr(authn(p,q,x)). \quad (10)$$

This trust value can be a probability derived from the data accumulated in transactions between p and q. Alternatively, p's administrator can set a value as a subjective probability or as an assurance level in the range [0,1]. For example, NIST Special Publication 800-63 (NIST: National Institute of Standards and Technology) [15] describes four assurance levels for the certainty values associated with an assertion according to the types of authentication mechanisms.

For propagation authentication trust, the acceptance of propagated information depends on the authentication of the entity propagating the information. On the basis of the relationship between probability and conditionals and the semantics of (2) and (5), the following values of authentication trust for attestation and mediation are defined:

$$v^{(a)}_{pq}(x) \overset{\text{def}}{=} \Pr(accept(p,x) | assert(q,x) \wedge authn(p,q,y)), \quad (11)$$

$$v^{(ua)}_{pq}(x) \overset{\text{def}}{=} \Pr(accept(p,x) | accept(q,x) \wedge authn(p,q,y)). \quad (12)$$

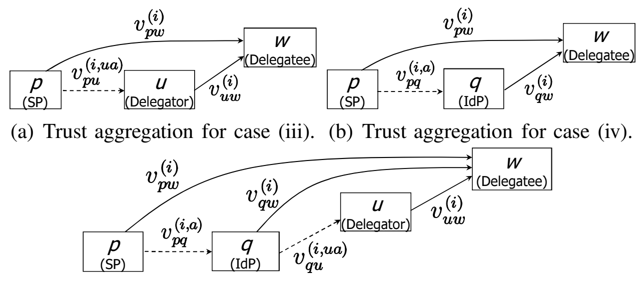
Let $v^{(i,a)}_{pq}(x)$ and $v^{(i,ua)}_{pq}(x)$ be the authentication trust values for $trust^{(i,a)}_{pq}(x)$ and $trust^{(i,ua)}_{pq}(x)$, respectively, in (3) and (6). The following equations are then obtained:

$$v^{(i,a)}_{pq}(x) = v^{(a)}_{pq}(x) \cdot v^{(i)}_{pq}(x), \quad (13)$$

$$v^{(i,ua)}_{pq}(x) = v^{(ua)}_{pq}(x) \cdot v^{(i)}_{pq}(x). \quad (14)$$

On the basis of the above definitions of authentication trust values, the Beth-Borcherding-Klein (BBK) metric [6] is applied to calculate the values of direct and propagation trust, shown in Figure 5. In an aggregated trust value, the BBK metric reflects the direct trust value more than the propagation value when the two values aggregated are sequentially located in the trust chain.

Next, each trust value for the five cases explained in Section V-A is calculated. The trust value for w's anonymous access, shown in case (i), is 0, since p does not have any information about w. The trust value for case (ii) is



(a) Trust aggregation for case (iii). (b) Trust aggregation for case (iv).

(c) Trust aggregation for case (v).

Figure 6. Authentication trust aggregation for delegatees.

equal to $v_{pw}^{(i)}(x)$, as defined in (10). The trust aggregations for cases (iii), (iv), and (v) are shown in Figures 6(a), 6(b), and 6(c), respectively. If we let a utility function $F(v_1, v_2)$ calculate $1 - (1 - v_2)^{v_1}$ for propagation trust aggregation in Figure 5(a), $v_{pw}^{(i)}(x)$ for cases (iii) and (iv) are obtained as $F(v_{pu}^{(i,ua)}(x), v_{uw}^{(i)}(x))$ and $F(v_{pq}^{(i,a)}(x), v_{qw}^{(i)}(x))$, respectively. Similarly, $v_{pw}^{(i)}(x)$ for case (v) is derived as $F(v_{pq}^{(i,a)}(x) \cdot v_{qu}^{(i,ua)}(x), v_{uw}^{(i)}(x))$ using the above results.

VI. Discussion

The proposed model enables authentication of a trustee who is not known to an AM by a trustor's identification of the trustee. This is effective because the scheme does not require registration of the trustee, which is needed for authentication in conventional security systems and often causes users to forgo proceeding with use of the service.

An IdP in the proposed framework can propagate authentication contexts using UA to SPs. In this approach, there is some vulnerability to a malicious trustor's attestation of a false or invalid authentication context. In this case, an SP receiving the context about an illegitimate trustee may grant his or her access inappropriately. Therefore, the framework needs to have a method for evaluating the trustworthiness of users and assessing the risk associated with their access as well as a formal and semantic representation of authentication contexts for UA, which will be future work.

UA can especially be strengthened if an AM can gain a trustee's accurate authentication context using its functionality that is validated by the trustor, as explained in the context validation example. In this sense, the strength of UA varies in authentication methods and their combinations. However, the flexibility of UA arises from the responsibility for authentication lying not with an AM, but with a trustor. This is an open issue in terms of which entity ensures a trustee's identity and how its integrity is assured.

VII. Conclusion and Future Work

A trust metric for deriving authentication context in an open environment was described. Based on the model, an authentication federation framework was proposed for propagating authentication trust in a person. In the framework, a user trusting in the person and an entity mediating the user's authentication of the person share the corresponding authentication context in a user-centric way. The logically derived authentication trust enables flexible access control in a quantitative fashion. Future work includes further investigation on the representation of authentication contexts, the responsibility of authentication, and risk assessment using the proposed framework.

References

[1] OASIS, "Assertions and protocol for the OASIS security assertion markup language (SAML) v2.0," Mar. 2005.

[2] IBM, Microsoft, BEA, RSA, and VeriSign, "Web Services Federation Language," Jul. 2003.

[3] OpenID, "OpenID Authentication 2.0 - Final," Dec. 2007.

[4] H. Gomi, "An Authentication Trust Metric for Federated Identity Management Systems," in *Proceedings of the 6th International Workshop on Security and Trust Management (STM'10)*, Sep. 2010, pp. 113–128.

[5] A. Jøsang, R. Ismail, and C. Boyd, "A Survey of Trust and Reputation Systems for Online Service Provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, Mar. 2007.

[6] T. Beth, M. Borcherding, and B. Klein, "Valuation of Trust in Open Networks," in *Proceedings of the 3rd European Symposium on Research in Computer Security (ESORICS'94)*, 1994, pp. 3–18.

[7] M. Reiter and S. Stubblebine, "Authentication Metric Analysis and Design," *ACM Transactions on Information and System Securiry*, vol. 2, no. 2, pp. 138–158, 1999.

[8] J. Huang and D. Nicol, "A Calculus of Trust and Its Application to PKI and Identity Management," in *Proceedings of the 8th Symposium on Identity and Trust on the Internet (IDtrust'09)*, Apr. 2009, pp. 23–37.

[9] A. Jøsang, J. Fabre, B. Hay, J. Dalziel, and S. Pope, "Trust Requirements in Identity Management," in *Proceedings of the Australasian Workshop on Grid Computing and E-research*, 2005, pp. 99–108.

[10] I. Thomas, M. Menzel, and C. Meinel, "Using Quantified Trust Levels to Describe Authentication Requirements in Federated Identity Management," in *Proceedings of the ACM Workshop on Secure Web Services (SWS'08)*, Oct. 2008, pp. 71–80.

[11] J.-M. Seigneur, S. Farrell, C. D. Jensen, E. Gray, and Y. Chen, "End-to-End Trust Starts with Recognition," in *Proceedings of the 1st International Conference on Security in Pervasive Computing (SPC'03)*, 2003.

[12] G. Theodorakopoulos and J. Baras, "Trust Evaluation in Ad-Hoc Networks," in *Proceedings of the 3rd ACM Workshop on Wireless Security (WiSe'04)*, 2004, pp. 1–10.

[13] OASIS, "Authentication Context for the OASIS Security Assertion Markup Language (SAML) V2.0," Mar. 2005.

[14] A. Bhargav-Spantzel, J. Camenish, T. Gross, and D. Sommer, "User Centricity: A Taxonomy and Open Issues," *Journal of Computer Security*, vol. 15, no. 5, pp. 493–527, 2007.

[15] W. Burr, D. Dodson, and W. Polk, "Electronic Authentication Guideline," Apr. 2006, http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf.

Migration towards a more secure authentication in the Session Initiation Protocol

Lars Strand
Norwegian Computing Center / University of Oslo
Oslo, Norway
lars.strand@nr.no

Wolfgang Leister
Norwegian Computing Center
Oslo, Norway
wolfgang.leister@nr.no

Alan Duric
Telio Telecom AS
Oslo, Norway
alan.duric@telio.no

***Abstract*—This paper specifies a two-step migration towards a stronger authentication in the Session Initiation Protocol. First, we add support for a Password Authenticated Key Exchange algorithm that can function as a drop-in replacement for the widely adopted Digest Access Authentication mechanism. This new authentication mechanism adds support for mutual authentication, is considered stronger and can rely on the same shared password used by the digest authentication. A more long-term solution is to replace the authentication scheme with the Simple Authentication and Security Layer. The Simple Authentication and Security Layer separates the authentication mechanisms from the Session Initiation Protocol, and adds support for a range of more secure authentication mechanisms in a generic and unified way. Both methods are presented, discussed, and shown how to integrate into the Session Initiation Protocol.**

***Keywords*—VoIP, SIP, authentication, PAKE, SASL.**

I. INTRODUCTION

Voice over IP (VoIP) is rapidly taking over for the traditional, public switched telephone networks (PSTN). Although there exist several competing network protocols that are capable of delivering VoIP, the Session Initiation Protocol (SIP) [1] and the Real-time Transport Protocol (RTP) [2] developed by the IETF have become the *de facto* industry standard. These two protocols fulfill two different functions – SIP is used for signaling, e.g., responsible for setting up, modifying and tearing down multimedia sessions, while RTP transports the actual media stream (voice). Although the SIP protocol is flexible and rich in functionality [3], several vulnerabilities and security attacks have been found [4]–[6].

Securing a SIP-based VoIP system has proven challenging and the reasons are multi-faceted:

- The scale and complexity of the SIP protocol specification, with primary focus on functionality rather than a sound security design [7].
- SIP usage of intermediaries, expected communication between nodes with no trust at all, and its user-to-user operation make security far from trivial [1, page 232].
- A large number of threats against VoIP systems have been identified [8]. Several security mechanisms for countermeasures have been proposed, but no single security mechanism is suited to address all these security threats concerning VoIP and SIP [9], [10].
- Since the SIP and RTP protocols share the same infrastructure as traditional data networks, they also inherit the security problems of data communication.
- VoIP services have strict requirements to the network performance with respect to Quality of Service since it is a duplex communication with low tolerance for latency, packet loss and saturation. Introducing strong security mechanisms might affect network performance [11].

Signaling in PSTN has traditionally involved trust between carriers, and by end users (caller-id). To achieve the same trust level using SIP, we need to employ secure authentication.

In VoIP, authentication tries to validate the identity of the communication peers and to bind that identity to a subject (peer). It must be stressed that the user is not authenticated, but the user's phone. In VoIP terminology, a subject could be a User Agent (UA), such as a phone, identified by a phone-number/username and IP-address/hostname pair, denoted as an Address-of-Record (AoR). The authentication in VoIP is therefore the assurance that a communicating entity, the UA, is the one that it claims to be [12]. However, the authentication in SIP has proven weak [13] and vulnerable to a real-world security attack [14].

Equally important for the UA is to establish the identity of the communicating peer, i.e., the SIP server. If the client does not authenticate the SIP server, it might risk to communicate and send content to a hostile SIP server.

The main contribution of this paper is to propose a migration towards a more secure SIP authentication. First, we introduce an authentication method based on the Password Authenticated Key Exchange (PAKE) [15], which provides mutual authentication based on a shared secret, and can function as a drop-in replacement of the digest authentication currently used. However, a more flexible authentication method is desired. As a second authentication method, we propose the Simple Authentication and Security Layer (SASL) [16], which enables SIP to transparently support and use more secure authentication methods in a unified and generic way.

The rest of the paper is organized as follows: In Section II we give an introduction to authentication in SIP and discuss related work. In Section III we show how a modified PAKE can be used to add mutual authentication in SIP. The second authentication mechanism added to SIP, SASL, is explained

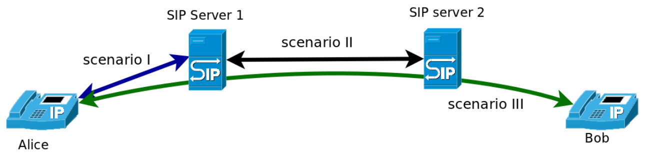


Figure 1: Three different usage scenarios where authentication in SIP is desired.

and discussed in Section IV. Conclusion and future work is presented in Section V.

II. Identity in the Session Initiation Protocol

We identify three scenarios where identity in SIP needs to be handled, as depicted in Figure 1: Scenario *I* between the UA and the local SIP server; Scenario *II* between SIP servers; and Scenario *III* end-to-end.

Scenario I between the UA and the local SIP server is relevant when the UA comes online and before any outgoing calls can be placed. Then, the UA must register itself to a local SIP server. During the SIP register handshake, the server usually challenges the UA to authenticate. Before placing a call (sending a SIP `INVITE`), the UA might be challenged again by the server to authenticate. The most common authentication method used between UA and server today is the Digest Access Authentication (DAA) [17].

Scenario II handles the authentication between SIP servers to achieve trust between SIP servers. It is not desirable to have SIP traffic handled by an unknown or untrusted SIP server that might have malicious intent. However, since most SIP servers today use some kind of SIP peering [18], the relationships between servers are often static and pre-defined. Therefore the identities between SIP servers are often predetermined by other security mechanisms than what are offered by SIP (like IPSec, TLS etc.).

Scenario III is about end-to-end authentication, which determines the identity of both the caller and the callee across different SIP domains. This is of particular importance and not easily attained in SIP. There is an increased threat and fear for both VoIP phishing and SPIT (Spam over Internet Telephony), that might seriously affect SIP-based VoIP services. By enforcing end-to-end authentication in SIP, these threats might be mitigated or prevented.

We list authentication mechanisms in SIP and their support in these three SIP scenarios in Table I. We shall discuss these authentication mechanisms in the following.

The DAA is currently the most common authentication mechanism for SIP. DAA is simple but rather insecure. It is the only authentication mechanism which support in SIP is mandatory [1, Section 22]. DAA uses the MD5 hash function and a challenge-response pattern, and relies on a shared secret between client and server within a SIP domain [17]. DAA is performed during the SIP `REGISTER` handshake between the UA and the SIP server, as depicted in messages 1-3 and 6 in Figure 2. The UA receives a nonce value from the SIP server, computes a digest hash value over the nonce, the shared secret and some other SIP header values, and send it to the SIP server. The SIP server computes the same digest hash. If both digests are identical, the UA is authenticated. The DAA is weak and vulnerable to a serious real-world attack [14]. Since the DAA relies on a shared secret and is only meaningful for a specific realm, its usage is limited to Scenario I.

Secure MIME (S/MIME) [19] is an authentication mechanism presented in the SIP core specification document RFC3261 [1]. S/MIME intends to achieve end-to-end authentication between UAs. The entire SIP message is encapsulated in a specific SIP message using MIME, which is signed and optionally encrypted. The receiving UA checks whether the sending UA's certificate is signed by a trusted authority. Since S/MIME depend on end-user certificates, the UAs must support multiple root certificates since no consolidated certificate authority exists. Additionally, certificate handling issues, such as revocation and renewal, complicate the use of certificates. There has been rather limited industry support for S/MIME.

Palmieri et al. [20] introduce a new authentication mechanism using digital signatures. But since they rely on certificates, their solution suffers under similar certification handling issues as S/MIME. They also admit that relying on public key infrastructure (PKI) is both difficult and costly to implement. Liao et at. [21], propose an improved authentication in SIP with self-signed public keys on elliptic curves. However, Liao's proposal uses smart-cards to store authentication data and rely on a trusted third party [22].

The SIP protocol needs an authentication mechanism that avoids the security vulnerabilities the currently used DAA has. A replacement authentication mechanism should preferably not rely on PKI, have support for strong mutual authentication, and support all three scenarios listed previously in this section.

III. Password authenticated key exchange

We propose to add support for a variant of PAKE denoted as "Key Agreement Method 3" (KAM3) as a cryptographic protocol [15, page 17]. PAKE has the following attractive features: *1*) PAKE provides mutual authentication between UA and the SIP server, and thus a rogue SIP server can not claim that the authentication succeed without knowing the shared password. PAKE assures the UA that the SIP server knows the UA's encrypted password. *2*) Reuse of the shared password used by DAA as the UA's credential, which enables our approach to easily replace DAA used within a local SIP domain (scenario I). *3*) PAKE offers strong protection of the shared secret if the communication is eavesdropped, that prevents brute-force attacks, including dictionary-based offline attacks, to which the DAA is vulnerable to.

Our approach follows the work of Oiwa et al. [23]. They use KAM3 to introduce a stronger authentication in HTTP and their initial design and specification is submitted to the IETF as an Internet Draft [24]. We have adapted their approach to SIP, since SIP closely resembles HTTP in both message structure and flow, and we need to prevent the `REGISTER` hijack attack presented earlier [14].

Authentication mechanisms	Supported authentication scenarios			Supported SIP methods	
	scenario I	scenario II	scenario III	`REGISTER`	`INVITE`
Digest authentication	yes	no	no	yes	yes
PAKE	yes	no	no	yes	yes
SASL	yes	yes	yes	yes	yes
GSS-API	yes	yes	yes	yes	yes
S/MIME	no	no	yes	no	yes

Table I: List of SIP authentication mechanisms and their support.

In KAM3, the UA and the SIP server compute cryptographic keys based on the shared password. These keys are exchanged, and a shared session secret is computed based on these keys. Each peer then computes a hash value of the session secret and some other values, which is sent to the requesting peer. The receiving peer computes the same hash value, and compares it with the received hash value. If these are identical, the sending peer is authenticated.

PAKE supports several authentication algorithms, which differ in their underlying mathematical groups and security parameters [24]. The only mandatory supported authentication algorithm, the *iso-kam3-dl-2048-sha256*, uses the 2048-bit discrete-logarithm defined in RFC3526 [25] and the SHA-256 hash function.

A. Initial requirements

In the following section, we let q an odd prime integer defining the number of elements in $F(q)$ which is a representation of a finite group. We let g the generator of a subgroup of r elements in $F(q)$. The one-way hash function is denoted as H.

Before the authentication starts, username and password must be set and configured. We compute a weak secret π used by the client as a one-way hash of the values $realm$, $username$ and $password$:

$$\pi = H(realm, username, password)$$

Here, $realm$ is the protection domain where SIP authentication is meaningful for a set of $username$ and $password$. The server does not need to store the shared password directly, only a specially encrypted version $J(\pi)$, where J is the password verification element derivation function defined as:

$$J(\pi) = g^{\pi} \bmod q$$

B. Message exchange

We need to extend the current SIP `REGISTER` handshake by one extra round-trip of SIP messages between the UA and the SIP server. These two extra messages are depicted in blue and numbered (4) and (5) in Figure 2. A more detailed specification is given in the following paragraphs, where the numbers refer to the protocol clauses depicted in Figure 2.

The UA registers to a SIP *location service* (SIP server). The initial SIP `REGISTER` message (1) from the UA is not authorized, and must be authenticated. The SIP server responds with a `401 Unauthorized` status message (2),

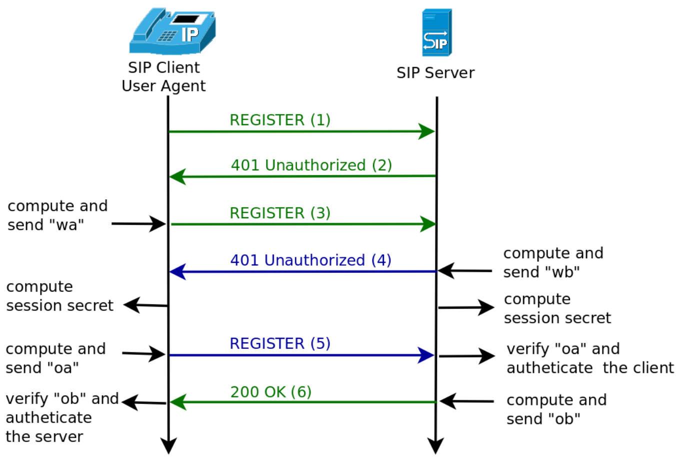


Figure 2: SIP `REGISTER` message flow with mutual authentication security using PAKE.

which contains a `WWW-Authenticate` header with details of the challenge, including *realm* and *algorithm*. The UA constructs a cryptographic value w_a generated from a random integer s_a:

$$w_a = g^{s_a} \bmod q$$

This value is sent in a new SIP `REGISTER` message (3) to the SIP server. The SIP server proceeds to generate and send another cryptographic value w_b, which is generated from $J(\pi)$, the received value w_a and a random integer s_b:

$$w_b = (J(\pi) \times w_a^{H(1,w_a)})^{s_b} \bmod q$$

At the next step, each peer computes a session secret z. The UA derives z based on π, s_a, w_a and w_b:

$$z = w_b^{(s_a + H(2,w_a,w_b))/(s_a * H(1,w_a)+\pi) \bmod r} \bmod q$$

Likewise, the SIP server derives z based on s_b, w_a and w_b using the following function:

$$z = (w_a \times g^{H(2,w_a,w_b)})^{s_b} \bmod q$$

The session secret z matches only if both peers have used the secret credentials generated from the same shared secret. The above equations are directly derived from the PAKE HTTP authentication specifications [24]. The next step is to validate the value of z at the communicating peer.

The UA sends a third SIP `REGISTER` message (5) and includes the value o_a which is a hash value computed as:

$$o_a = H(4, w_a, w_b, z, contactURIs)$$

Here, $contactURIs$ is the value of the UA's `Contact` SIP header value. This value is integrity-protected to prevent register hijacking attacks as presented in [14]. The SIP server, upon receipt of o_a, performs the same hash operation, and compares the results. If these results are identical, the UA is authenticated. The SIP server then sends a final message (6), with the value o_b computed as:

$$o_b = H(3, w_a, w_b, z, contactURIs)$$

When the UA receives o_b, it verifies this value by computing its hash value. If the results are identical, the SIP server is authenticated to the UA. After a complete message exchange, the UA is authenticated to the SIP server, and the SIP server has been authenticated to the UA.

C. SIP message syntax

A SIP message consists of several headers and a body. The SIP header fields are textual, always in the format `<header_name>: <header_value>`. The header value can contain one or more parameters. We embed the cryptographic values derived in the previous section as base64-encoded [26] SIP header values. We re-use the SIP DAA headers to carry PAKE authentication data, so that PAKE can be used as a drop-in replacement for DAA. A SIP `REGISTER` message with a DAA `Authorization` header is depicted in Figure 4. Again, we refer to the protocol clauses with a number in parentheses as depicted in Figure 2.

The UA first sends a SIP `REGISTER` without any authentication credentials (1). The SIP server responds with a `401 Unauthorized` status message (2), which contains a `WWW-Authenticate` header with header values *realm* and *algorithm*:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: Mutual realm="asterisk",
  algorithm="iso-kam3-dl-2048-sha256"
```

The UA then computes w_a and sends it to the SIP server using a new SIP `REGISTER` message (3), with the required values embedded in the `Authorization` header:

```
SIP/2.0 REGISTER
Authorization: Mutual user="alice",
  algorithm="iso-kam3-dl-2048-sha256",
  wa="Q29tcHV0ZWQgd2E...ljaCBcyBsb25nCg=="
```

The next required values in the authentication mechanism w_b, o_a and o_b are embedded and sent using these two SIP headers.

IV. Simple Authentication and Security Layer

The Simple Authentication and Security Layer (SASL), defined in RFC4422 [16], provides an interface for authentication and an authentication negotiation mechanism. The SASL specification is developed and maintained within the IETF, and have been scrutinized by security professionals over the years. It has been extensively tested, and is now classified as a mature standard by the IETF. SASL is also implemented and used in several popular communications protocols applications like IMAP, SMTP and LDAP[1].

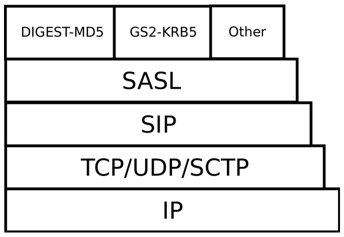


Figure 3: The SIP protocol stack with SASL and underlying security mechanisms.

The SASL framework does not provide authentication mechanisms in itself, but supports different underlying authentication mechanisms through a standardized interface[2]. SASL does not provide a transport layer and thus relies on the application to encapsulate, send and extract SASL messages between client and server. The SASL messages sent between client and server contain authentication data, and are opaque from the viewpoint of the calling application. The application only needs to add support to a SASL software library implementation, and thus have support to a range of underlying authentication mechanisms the library supports.

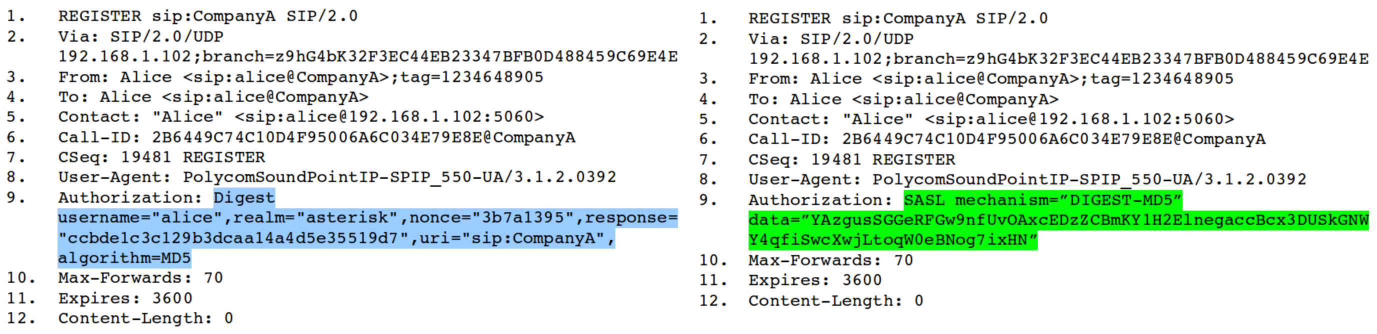
Adding support for the security abstraction layer framework Generic Security Services API (GSS-API) has been done earlier [27]. While the GSS-API is intended for use with applications, SASL is used in, and intended for, communication protocols. The functionalities offered by the GSS-API and SASL are alike, but the SASL specification is more high-level, and allows more freedom in implementing the SASL requirements. SASL also supports more underlying security mechanisms than the GSS-API. By using the "GS2" mechanism family, the GSS-API can be used as an underlying security mechanism in SASL. However, the GSS-API negotiation mechanism cannot be used due to security concerns [28, Section 14].

A. SASL profile for SIP

A modified PAKE authentication can more easily replace the current digest (DAA) authentication used in SIP, since they both rely on a shared secret and use the same SIP headers. PAKE also introduces a stronger authentication than DAA. However, a more flexible authentication mechanism is desired. Different VoIP scenarios require different security requirements, and the communicating peers should be able

[1]The Carnegie Mellon University's implementation: http://asg.web.cmu.edu/sasl/ and the GNU SASL library: http://www.gnu.org/software/gsasl/ are two popular and freely available SASL libraries.

[2]A list of registered SASL mechanisms is maintained by IANA: http://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xml



```
REGISTER sip:CompanyA SIP/2.0
Via: SIP/2.0/UDP
192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
From: Alice <sip:alice@CompanyA>;tag=1234648905
To: Alice <sip:alice@CompanyA>
Contact: "Alice" <sip:alice@192.168.1.102:5060>
Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
CSeq: 19481 REGISTER
User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
Authorization: Digest
username="alice",realm="asterisk",nonce="3b7a1395",response=
"ccbde1c3c129b3dcaa14a4d5e35519d7",uri="sip:CompanyA",
algorithm=MD5
Max-Forwards: 70
Expires: 3600
Content-Length: 0
```

```
REGISTER sip:CompanyA SIP/2.0
Via: SIP/2.0/UDP
192.168.1.102;branch=z9hG4bK32F3EC44EB23347BFB0D488459C69E4E
From: Alice <sip:alice@CompanyA>;tag=1234648905
To: Alice <sip:alice@CompanyA>
Contact: "Alice" <sip:alice@192.168.1.102:5060>
Call-ID: 2B6449C74C10D4F95006A6C034E79E8E@CompanyA
CSeq: 19481 REGISTER
User-Agent: PolycomSoundPointIP-SPIP_550-UA/3.1.2.0392
Authorization: SASL mechanism=”DIGEST-MD5”
data=”YAzgusSGGeRFGw9nfUvOAxcEDzZCBmKY1H2ElnegaccBcx3DUSkGNW
Y4qfiSwcXwjLtoqW0eBNog7ixHN”
Max-Forwards: 70
Expires: 3600
Content-Length: 0
```

Figure 4: A SIP `REGISTER` message with the original DAA `Authorization` header to the left, and the same header carrying SASL data to the right.

to negotiate the best possible authentication mechanism supported.

Instead of adding numerous different authentication mechanisms to SIP based on different security requirements, it is desirable to keep the changes to the SIP standard to a minimum. The industry might also be reluctant to adopt immature and non-standardized security services, like different (new) authentication mechanisms. Adding support for SASL requires only small changes to the SIP standard, and can utilize several underlying authentication mechanisms. The SIP protocol stack with SASL is shown in Figure 3.

In SASL terminology, the description on how to encapsulate SASL negotiation and SASL messages for a given protocol, is called a "SASL profile". We create a SASL profile for SIP by reusing the `WWW-Authenticate` and `Authorization` SIP headers used by the digest authentication, shown earlier. Instead of encapsulating DAA data, we embed SASL messages, as depicted in Figure 4.

When discussing PAKE authentication earlier, we added one round-trip of SIP messages between the UA and the SIP server. When using SASL, the number of messages going back and forth depends on the underlying authentication mechanism. We therefore extend the SIP `REGISTER` handshake with an arbitrary number of round-trips, until the underlying authentication mechanism has completed communication.

In the following paragraphs, the numbers in parentheses refer to the SIP message numbers in Figure 2. The SASL specification only outlines a very high-level method of how the server should advertise its supported mechanisms to the client. We implement the mechanism negotiation in the first three messages in the SIP `REGISTER` handshake (1-3). The UA starts by requesting authentication from the SIP server, with no `Authorization` header (1). The SIP server responds with a `401 Unauthorized` SIP message (2), with the supported and available mechanisms embedded in the `WWW-Authenticate` header:

```
SIP/2.0 401 Unauthorized
WWW-Authenticate: SASL
  negotiate="DIGEST-MD5 NTLM GS2-KRB5"
```

The client selects the best mechanism from the received list that it supports and sends a new SIP `REGISTER` message (3). This message includes an `Authorization` header requesting authentication with "GS2-KRB5" as the preferred mechanism. The initial authentication data is embedded base64 encoded to the *data* parameter:

```
SIP/2.0 REGISTER
Authorization: SASL mechanism="GS2-KRB5",
  data="SUZZT1VDQU5SR...JUPVVQU5FUkQK="
```

The server retrieves the SASL data, and passes the message to the SASL library which handles the authentication. The selected authentication method continues to pass SASL messages between client and server as many times as necessary to complete the authentication (messages 4-5 are repeated). Once the authentication is complete, the SIP server sends a `200 OK` SIP message. Should the server have some last SASL data to be communicated to the client to complete the authentication, it can be carried in a `WWW-Authenticate` header embedded in the `200 OK` message:

```
SIP/2.0 200 OK
WWW-Authenticate: SASL mechanism="GS2-KRB5",
  data="TFoG9rP56zrVH...YaAOndwPew6NdxKr"
```

As soon as the `200 OK` message is received and processed, the client is authenticated to the SIP server. Since the mechanism negotiation is not integrity-protected, the UA is vulnerable to a "down-grade" attack. An attacker can intercept and modify the negotiation messages so that the least favorable authentication method is used.

V. Conclusion and future work

In our earlier work, we have shown and implemented a real-world attack to the widely deployed DAA method [27]. In this paper we have added support to a new improved authentication mechanism that can easily replace DAA based on a modified PAKE algorithm. This new authentication mechanism adds support for mutual authentication and is more secure than DAA. We have also shown that the modified PAKE authentication can easily function as a drop-in replacement for DAA. However, a more flexible authentication mechanism is desired in the long-term.

Our second authentication mechanism supported in SIP is SASL, which is not an authentication mechanism *per se*, but introduces a security abstraction layer. This abstraction layer adds support to a range of underlying authentication mechanism in a unified way. As long as SIP supports SASL, new authentication mechanisms can be added later to the SASL library, without requiring any change to the SIP protocol. We have also introduced a SASL mechanism negotiation that enables the communicating peers to agree upon the "best" available authentication mechanism.

We envisage a two-step migration towards a stronger authentication scheme in SIP. First, the modified PAKE authentication is implemented and deployed. Second, the long-term solution is to deploy SASL with support for a range of underlying authentication mechanisms.

Future work will look into implementing a proof of concept for PAKE-enabled UA and SIP server, including overhead evaluation benchmarks for the new authentication algorithm. We also plan to evaluate different SASL security mechanism and their implications for SIP, and decide which authentication mechanisms should be mandatorily supported through SASL. A co-operation with the IETF and the kitten Working Group to further elaborate a SASL profile for SIP is also planned. We hope our work will gain acceptance and industrial deployment, so that the previously mentioned security attacks can be countered.

ACKNOWLEDGMENT

This research is funded by the EUX2010SEC project in the VERDIKT framework of the Norwegian Research Council (Norges Forskningsråd, project 180054). The authors would like to thank the anonymous reviewers for their valuable comments on earlier drafts of this paper.

REFERENCES

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261 (Proposed Standard), Internet Engineering Task Force, Jun. 2002, updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630, 5922, 5954, 6026. [Online]. Available: http://www.ietf.org/rfc/rfc3261.txt 11. Apr 2011

[2] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," RFC 3550 (Standard), Internet Engineering Task Force, Jul. 2003, updated by RFCs 5506, 5761. [Online]. Available: http://www.ietf.org/rfc/rfc3550.txt 11. Apr 2011

[3] H. Sinnreich and A. B. Johnston, Internet communications using SIP: Delivering VoIP and multimedia services with Session Initiation Protocol, 2nd ed. New York, NY, USA: John Wiley & Sons, Inc., August 2006.

[4] H. Dwivedi, Hacking VoIP: Protocols, Attacks, and Countermeasures, 1st ed. No Starch Press, Mar. 2009.

[5] D. Endler and M. Collier, Hacking Exposed VoIP: Voice over IP Security Secrets and Solutions. McGraw-Hill Osborne Media, November 2006.

[6] A. M. Hagalisletto and L. Strand, "Designing attacks on SIP call set-up," International Journal of Applied Cryptography, vol. 2, no. 1, pp. 13–22(10), July 2010.

[7] D. Sisalem, J. Floroiu, J. Kuthan, U. Abend, and H. Schulzrinne, SIP Security. WileyBlackwell, Mar. 2009.

[8] VoIPSA, "VoIP security and privacy threat taxonomy," Public Realease 1.0, Oct. 2005. [Online]. Available: http://voipsa.org/Activities/VOIPSA_Threat_Taxonomy_0.1.pdf 1. Nov 2011

[9] D. York, Seven Deadliest Unified Communications Attacks. Syngress, Apr. 2010.

[10] P. Park, Voice over IP Security. Cisco Press, Sep. 2008.

[11] S. Salsano, L. Veltri, and D. Papalilo, "SIP security issues: The SIP authentication procedure and its processing load," Network, IEEE, vol. 16, pp. 38–44, 2002.

[12] International Telecommunication Union (ITU), "Security Architecture For Open Systems Interconnection (OSI)," The International Telegraph and Telephone Consultative Comittee (CCITT), X.800 Standard X.800, 1991.

[13] A. M. Hagalisletto and L. Strand, "Formal modeling of authentication in SIP registration," in Second International Conference on Emerging Security Information, Systems and Technologies SECURWARE '08. IEEE Computer Society, August 2008, pp. 16–21.

[14] L. Strand and W. Leister, "Improving SIP authentication," in Proceedings of the Tenth International Conference on Networking (ICN2011). Xpert Publishing Services, Jan 2011, pp. 164 – 169.

[15] International Organization for Standardization and ISO, "ISO/IEC 11770-4:2006: Information technology – Security techniques – Key management – Part 4: Mechanisms based on weak secrets," 2006.

[16] A. Melnikov and K. Zeilenga, "Simple Authentication and Security Layer (SASL)," RFC 4422 (Proposed Standard), Internet Engineering Task Force, Jun. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4422.txt 11. Apr 2011

[17] J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Luotonen, and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication," RFC 2617 (Draft Standard), Internet Engineering Task Force, Jun. 1999. [Online]. Available: http://www.ietf.org/rfc/rfc2617.txt 11. Apr 2011

[18] L. Strand and W. Leister, "A Survey of SIP Peering," in NATO ASI - Architects of secure Networks (ASIGE10), May 2010.

[19] J. Peterson, "S/MIME Advanced Encryption Standard (AES) Requirement for the Session Initiation Protocol (SIP)," RFC 3853 (Proposed Standard), Internet Engineering Task Force, Jul. 2004. [Online]. Available: http://www.ietf.org/rfc/rfc3853.txt 11. Apr 2011

[20] F. Palmieri and U. Fiore, "Providing true end-to-end security in converged voice over IP infrastructures," Computers & Security, vol. 28, no. 6, pp. 433–449, Sep. 2009.

[21] Y. Liao and S. Wang, "A new secure password authenticated key agreement scheme for SIP using self-certified public keys on elliptic curves," Computer Communications, vol. 33, no. 3, pp. 372–380, Feb. 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366409002631 11. Apr 2011

[22] Y. Liao, "Secure password authenticated key exchange protocols for various enviroments," Ph.D. dissertation, Tatung University, Dec. 2009.

[23] Y. Oiwa, H. Watanabe, and H. Takagi, "Pake-based mutual http authentication for preventing phishing attacks," CoRR, vol. abs/0911.5230, 2009. [Online]. Available: http://arxiv.org/abs/0911.5230 11. Apr 2011

[24] Y. Oiwa, H. Watanabe, H. Takagi, Y. Ioku, and T. Hayashi, "Mutual Authentication Protocol for HTTP," Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://tools.ietf.org/html/draft-oiwa-http-mutualauth-08 11. Apr 2011

[25] T. Kivinen and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)," RFC 3526 (Proposed Standard), Internet Engineering Task Force, May 2003. [Online]. Available: http://www.ietf.org/rfc/rfc3526.txt 11. Apr 2011

[26] S. Josefsson, "The Base16, Base32, and Base64 Data Encodings," RFC 4648 (Proposed Standard), Internet Engineering Task Force, Oct. 2006. [Online]. Available: http://www.ietf.org/rfc/rfc4648.txt 11. Apr 2011

[27] L. Strand, J. Noll, and W. Leister, "Generic security services API authentication support for the session initiation protocol," in Proceedings of The Seventh Advanced International Conference on Telecommunications (AICT2011). Xpert Publishing Services, Mar 2011, pp. 117 – 122.

[28] S. Josefsson and N. Williams, "Using Generic Security Service Application Program Interface (GSS-API) Mechanisms in Simple Authentication and Security Layer (SASL): The GS2 Mechanism Family," RFC 5801 (Proposed Standard), Internet Engineering Task Force, Jul. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5801.txt 11. Apr 2011

PASER: Position Aware Secure and Efficient Route Discovery Protocol for Wireless Mesh Networks

Mohamad Sbeiti, Andreas Wolff and Christian Wietfeld
Communication Networks Institute (CNI)
Faculty of Electrical Engineering and Information Technology
Dortmund University of Technology, Germany
Email: {Mohamad.Sbeiti, Andreas.Wolff, Christian.Wietfeld}@tu-dortmund.de

***Abstract*—In this paper we address an acceptable trade-off between security and performance of the route discovery process in wireless mesh networks. We propose a Position Aware Secure and Efficient reactive hierarchical Route discovery protocol (PASER). The proposed protocol is tailored for rescue and emergency operations and aims to combat unauthorized nodes of joining the network or manipulating the route look-up process. In addition, it deals with efficiency and real-time capability requirements in such environments. From a security perspective, the novelty of PASER is the combination of digital signature with lightweight authentication tree and symmetric block cipher to secure routing messages. Another key feature is the support of nodes' geo-positions to increase the security while enabling an advanced network management. Apart from that, PASER treats the network in a hierarchical way and establishes the route discovery process to a large extent upon reactive unicast messages. PASER is generally applicable, as it does not make restrictive assumptions on the network nodes. It provides generic metrics for the constituent links of the discovered routes, allowing the implementation of any route selection algorithm. As a result, PASER enables secure and efficient routing in a wide range of wireless mesh network applications.**

***Keywords*-Secure routing protocols; wireless mesh networks; emergency and rescue operations.**

I. Introduction

Wireless Mesh Networking (WMN) is an emerging technology, which is receiving increased attention as a high-performance, low-cost and rapid deployment solution for next generation wireless communication systems. A WMN is defined as a dynamic, self-organized and self-configured wireless multi-hop network, consisting of gateways, mesh routers and mesh clients. Gateways and mesh routers build the network backhaul and are responsible for client data transmission. Typically, gateways provide connection to the Internet, whereas mesh routers are responsible for setting up and maintaining the ad hoc network routes. Due to its ubiquitous architecture and wireless transmit channel, it is possible though that mesh routers deviate from the protocol definition and exhibit malicious behavior. The challenge is to prevent such nodes, which we term attackers, from misleading other nodes that a path is better than it actually is. If successful, an adversary can attract network traffic and degrade or disable the communication of other nodes, which might be very crucial in many WMN applications.

Rescue and emergency operations, for instance, is a WMN application field addressed by the research project SPIDER [1], where rescue fighters deploy an ad hoc incident network using dropped units [2]. These operations are very time sensitive and dangerous minds might be on board. Without a satisfactory level of security, terrorists or benefiting organizations may try to disrupt the communication route between rescue fighters and the Command and Control System (CCS). They might try to inject fraud packets to falsify CCS decisions or create a routing black hole, which attracts and sniffs data packets, where any release of such sensitive data could cause a mass hysteria across countries.

Thus, one of the fundamental challenges of the WMN technology is the design of a route discovery protocol that can efficiently establish accurate routes in presence of attackers. Hereby, it needs to deliver data packets between mobile clients with minimum communication overhead, low end-to-end delay and high throughput.

The rest of this paper is organized as follows: Section II reports on related work. Section III presents a review on security threats in WMN and outlines the needed security characteristics to secure routing protocols. In Section IV, PASER is demonstrated, where in Section V its security and performance are discussed. Finally, in Section VI we conclude the paper and give some outlook for future work.

II. Related Work

Most WMN mesh routers nowadays, e.g., HiMoNN [3], are built upon routing protocols designed by the IETF MANET working group: AODV [4], DYMO [5] and OLSR [6]. These protocols along with a plenty of other MANET routing protocols can not be fully applied in WMN for the following reasons:

1) They are designed without having security in mind. Retrofitting pre-existing cryptosystem (e.g., IPsec) to secure them is inefficient. These cryptosystems impose huge overhead and processing delay, hence affecting strongly the overall performance.
2) They deal with the network as a flat network, which is absolutely reasonable in MANET. Thereby, they do not consider the different roles of WMN nodes, namely, mesh routers and gateways. Thus, these protocols are not able to take advantage of WMN characteristics, i.e., most data flow is destined to the gateway (e.g., from rescue fighters to officer in charge).

Many security solutions to secure routing in MANET have been recently proposed, however most of them comprise either high computational complexity [7] or impose a lot of configuration and management [8] or are still vulnerable to several attacks [9][10][11].

To exploit the WMN characteristics, IEEE has been discussing since 2003 the release of the IEEE 802.11s standard, which deals with hierarchical mesh networks. The current draft has defined a routing mechanism for WMN and termed it Hybrid Wireless Mesh Protocol (HWMP) [12]. However, security in routing or forwarding functionality is not specified in that standard. The protocol does not provide any authentication and integrity of routing messages. Apart from that, HWMP as well as many protocols applied in WMN incorporate a proactive part [13]. Though this part is essential to keep the route to the gateway valid, it is very resource consuming and is always active even when it is not necessary.

In PASER, we address the latter point by adopting the reactive route discovery method with two differences:

- Mesh routers are always responsible for maintaining a route to the gateway. This brings the same advantage as a proactive part in hybrid protocols while using the resources only when needed.
- Route requests are forwarded, when possible, in a unicast manner rather than flooding them blindly, as in conventional on-demand route discovery methods.

From a security point of view, PASER provides a novel hybrid scheme, a combination of asymmetric and lightweight symmetric cryptography, to secure route discovery messages. This novel combination yields a huge performance gain while providing a very high security level. Apart from that, PASER supports the exchange of geo-positioning, hence mitigating a wider range of attacks and facilitating the network management.

III. Security Vulnerabilities in WMN Route Discovery

As mentioned before, PASER's main target scenarios are disaster rescue and relief operations. In such environments, safeguards are indirectly applied to the nodes preventing their compromise, e.g., nodes are mounted on fire brigades tubes. Thus, internal attacks, where nodes from within the network are involved, are a less realistic threat in these environments, whereas external attacks, which are performed by illegitimated nodes, are of paramount importance. The latter class of attacks essentially aims to violate the reliability of the network and the availability of its services. The most relevant attacks of this class with respect to the route discovery process are listed below:

Impersonation attack: Using MAC and IP spoofing, an attacker fakes the identity of authorized nodes and thereby joins the network. The attacker can then carry out all types of insider attacks - *the lack of proper authentication of nodes is the main reason for the success of impersonation attack.*

Location disclosure attack: This attack reveals information regarding the location of nodes or the structure of the network. It gathers the node location information, such as a route map, and attempts to learn the network traffic pattern. By analyzing changes in the traffic pattern, attackers try to figure out the identities of communication parties and plans further attack scenarios - *the lack of anonymity and confidentiality of routing information is the ground of this attack.*

Malign attack: An attacker blackmails an uncompromised node, causing other nodes to exclude it from the network, thus, prohibiting that node to exchange data - *a weak node revocation mechanism is the essential reason of the network vulnerability to such an attack.*

Man-in-the-middle attack: An attacker impersonates a sender and a receiver by establishing independent connections to them and making them believe that they are talking directly to each other. The success of such an attack gives the attacker a full control of the entire conversation. The attacker must be able to intercept all messages going between the two victims and inject new ones, which is straightforward in many circumstances (for example, an attacker within reception range of two nodes, can insert himself as a man-in-the-middle). A man-in-the-middle attack can only succeed when the attacker can impersonate each endpoint to the satisfaction of the other - *weak mutual authentication is the main reason for the attacker's ability of man-in-the-middle attack.*

Replay attack: An attacker records another node's valid control messages and resends them later. First, this causes other nodes to update their routing table with stale routes. Second, unnecessary packets are processed and forwarded within the network. The latter, also known as resource consumption attack, targets to consume network bandwidth and node battery power - *the main reason of the network vulnerability to such attack is the lack of adequate packet freshness verification mechanism.*

Tempering attack: An attacker forges routing packets generated by legitimated nodes (e.g., tempering sequence number or metric of packets) and hence causes wrong routing decisions like redirection through suboptimal routes or route loops. This attack causes severe degradation in network performance - *the fundamental reason of the attacker's ability of tempering the routing information is the lack of packet integrity check.*

Wormhole attack: A pair of attackers, linked via a fast transmission path (tunnel), forward route requests more quickly than legitimate nodes. The tunneled packets can propagate faster than those through a normal multi-hop route. This causes victim nodes to always use the tunneled route to transmit their packets. The latter enables the attackers to gain information about specific communication traffic in the network or selectively forward packets. The attacker could even prevent the discovery of any routes other than through the wormhole - *the lack of authentication of transmissions between neighboring nodes in the route discovery is a main issue with respect to this attack.*

Thus, in order to combat the aforementioned attacks and to mitigate their risk to a large extent, a secure route discovery protocol has to fulfill the following security goals:

1) Anonymity
2) Message confidentiality
3) Message freshness and integrity
4) Neighbor transmissions authentication
5) Node authentication

The first goal is only necessary to combat location disclosure attack. We didn't consider this goal while designing PASER for the following reasons:

- The location and role of nodes in environments such as disaster rescue and relief operations are to a large extent known and hence protection against this attack is not really required.
- The performance cost of achieving the anonymity goal is very high and we are seeking a good trade-off between security and performance.

IV. The PASER Protocol

In this Section, we describe the main components of PASER.

A. *PASER Objectives*

PASER is an efficient secure route discovery protocol for wireless mesh networks. It is a mechanism that provides a route to a node (mesh router or gateway) wishing to send a packet to a destination. Hereby, PASER asserts that the discovered route is accurate in terms of metric and legitimized nodes in the presence of external attackers. Moreover, it keeps the consumption of network resources minimal. That is, PASER aims to ensure the reliability of the network and the availability of its services in an efficient manner.

From security point of view, PASER has to fulfill the following goals: Message confidentiality, message freshness and integrity, neighbor transmissions authentication and node authentication. Message confidentiality is only used where PASER is vulnerable against man-in-the-middle attacks. From performance point of view, PASER aims to strongly decrease the number of messages it exchanges over the network and to keep the cost of its security mechanisms minimal. To achieve these goals, we consider the following assumptions in the given priority:

1) Only legitimated nodes hold a valid certificate.
2) Nodes feature low mobility.
3) GPS signals are available at the application scene and nodes incorporate a secure GPS device, i.e., received GPS information is secure in terms of integrity and authenticity.

B. *PASER Cryptographic Primitives*

In this Subsection, we describe how the main security building blocks of PASER are applied.

1) Digital Signature Scheme: PASER specifies to apply a digital signature on its broadcast-messages. This signature is mainly necessary to guarantee the authenticity of these messages and thereby to establish trust between one hop neighbors. We recommend any of the standardized algorithms in [14]. The key pair used by the algorithm is the one bounded to the node identity in his certificate.

2) Symmetric Block Cipher: PASER prescribes the use of symmetric block cipher to encrypt its unicast-messages. This encryption is mainly necessary to protect these messages against man-in-the-middle attacks within a short time interval after sending them. The key used by the cipher is a group key distributed to the nodes during the setup phase of the network. The selection of the block cipher depends on the application of PASER and therefore it is left open. We recommend however the usage of the lightweight block cipher PRESENT [15]. PRESENT was specifically designed with constrained applications such as passive low-cost RFID-tags in mind. PRESENT is a simple substitution-permutation network with a block size of 64 bits and two different key sizes: 80 or 128 bits. We recommend the version with an 80 bit key for PASER since here we are seeking short-term security.

3) Authentication Tree: An authentication tree [16] is a complete binary-tree equipped with a hash function and an assignment function F such that for any interior node n_{parent} and two child nodes n_{left} and n_{right} the function F satisfies: $n_{parent} = F(n_{left}, n_{right}) = hash(n_{left}||n_{right})$, with $||$ denoting concatenation. The hash function to be used should be practically secure and efficient, such as SHA-256 or the winner of the SHA-3 competition [17]. We use authentication tree in PASER to build from hash functions an lightweight secure authentication scheme between one hop neighbors. Figure 1 illustrates an example of this approach.

Each node generates 2^n secrets, where n is a configuration parameter and is determined based on the application of PASER; these secrets are the leaf pre-images of the tree. Each leaf node is a hash of these secrets and each internal node is the hash of the concatenation of two child values. After computing root, a node (Alice) publishes that root to its one hop neighbors (Bob). A node can then authenticate itself to a neighbor by disclosing one secret, e.g., $Secret1$,

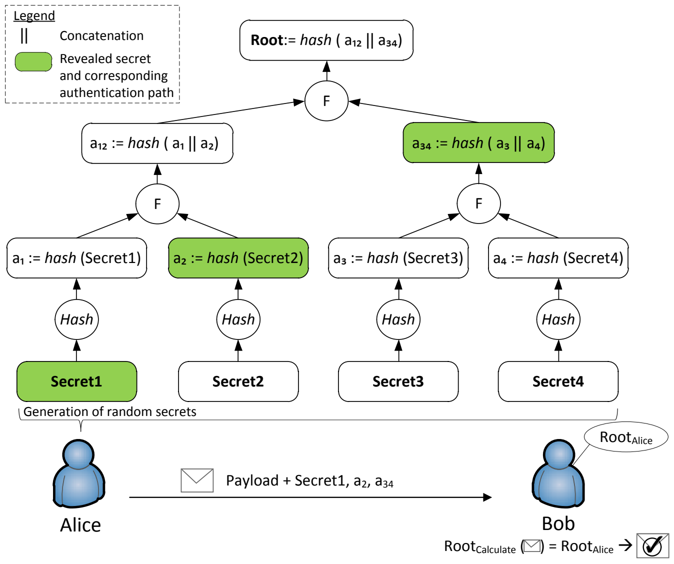


Figure 1. Authentication Tree Application

Table I
PASER MESSAGES

Name	Notation
Untrusted Broadcast Route Request	UB-RREQ
Untrusted Unicast Route Reply	UU-RREP
Trusted Unicast Route Request	TU-RREQ
Trusted Unicast Route Reply	TU-RREP
Trusted Unicast Route Reply-Acknowledge	TU-RREP-ACK

and sending it along with its authentication path, a_2 and a_{34}, see Figure 1. The authentication path of a secret consists of values of all the siblings of the secret corresponding leaf on the path between that leaf and the root. To verify the disclosed secret a receiver needs to compute the potential values of its ancestors by iteratively using of the F function. A secret is authenticated and accepted as correct if and only if the computed root value is equal to the already known root value of the node.

PASER tree secrets are l bits long, where l is a configuration parameter and $l > n$. A secret shall be constructed as specified in Figure 2.

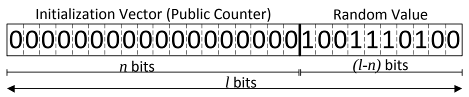

Initialization Vector (Public Counter)
Random Value
000000000000000000100111010O
n bits
(l-n) bits
l bits

Figure 2. Authentication Tree Secret Construction

The least significant $(l - n)$ bits are generated randomly for each secret. The most significant n bits constitute an initialization vector, the value of which is 0 for the first secret. The initialization vector is then incremented by one by each subsequent secret. When the maximum value $(2^n - 1)$ is reached, a node must generate a new root. The latter asserts the freshness of a secret. That is, a secret value can never be used twice for a given root. This technique is used to prevent replay attacks.

C. PASER Messages

PASER differs between messages destined to new neighbors and messages addressed to already known, trusted neighbors. From security perspective, messages addressed to new neighbors comprise identification fields that aim to establish a trusted relationship. These messages are always signed and their name is always prefixed with *U*, which stands for untrusted. Messages sent to trusted neighbors just include authentication fields to confirm the identity of the sender. These messages are always encrypted and their name is prefixed with the letter *T*, for trusted. PASER comprises five types of messages as depicted in Table I.

Table II (see next page) depicts the fields which constitutes these messages,

where * denotes fields that are included in a message if and only if the gateway flag *GFlag* is set. *Seq* is a concatenation of message type and node ID, where ID matches a sequence number in [4]. The address range list indicates all the addresses a node is responsible for. The latter is necessary in case of multiple interfaces. It allows the declaration of all node interfaces that participate in another routing domain. This is necessary in WMN since mesh routers mostly comprise at least two interfaces.

D. PASER States

In PASER, a node can be in two different states as illustrated in Figure 3.

At power-up the node enters the *UNREGISTERED* state. In this state the node is not known to the network. Before any communication can take place, it undergoes the following steps in the given order:

1) It generates empty routing and neighbor tables according to Table III. Hereby, a neighbor table comprises the position field if and only if the node is mesh router. In contrast, this field is included in the routing table by a gateway, because a gateway in PASER has knowledge of the position of all nodes. The neighbor Flag (*NeighFlag*) reflects the trust relation between a neighbor and that node.
2) It computes a hash tree root element and depending on the node type it executes the following:
   Gateway: It requests a random group key from a key distributing center (KDC). The physical location of the latter is less significant. For instance, in emergency and rescue operations it is reasonable to install the key distribution center as a web service at the CCS. Typically, gateways are placed near to the fire-fighting command and control vehicle and have a stable Internet link to the KDC, e.g., via satellite.
   Mesh router: It starts a route discovery for a gateway. To augment the security of this step, gateways may be assigned with the role "*gateway*" in their certificates.
3) It requests a certificate revocation list from a certificate authority and enters the registered state. We do neither restrict the choice of the protocol used to request the

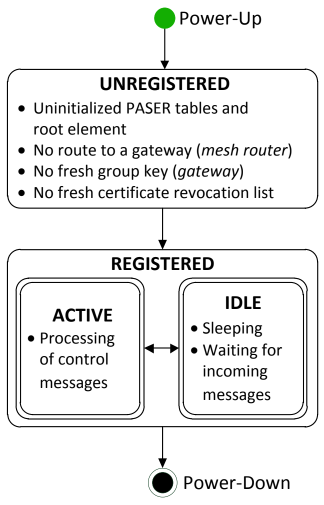

Power-Up
UNREGISTERED
• Uninitialized PASER tables and root element
• No route to a gateway (mesh router)
• No fresh group key (gateway)
• No fresh certificate revocation list
REGISTERED
ACTIVE
• Processing of control messages
IDLE
• Sleeping
• Waiting for incoming messages
Power-Down

Figure 3. Node Lifetime State Machine

Table II
MESSAGE CONTENT DECLARATION

Field	UB-RREQ	UU-RREP	TU-RREQ	TU-RREP	TU-RREP-ACK
			Basic fields		
Message type	✓	✓	✓	✓	✓
Querying node	✓	✓	✓	✓	✓
Destination node (*Dest*)	✓	✓	✓	✓	✓
Sequence number (*Seq*) of querying node	✓	✓	✓	✓	✓
Destination node gateway flag (*GFlag*)	✓	✓	✓	✓	
Address range list (*AddL*) of forwarding node	✓	✓		✓	
Route list from querying node to forwarding node	✓	✓	✓	✓	
Metric for the route between querying node and forwarding node	✓	✓	✓	✓	
Metric for the route between destination node and forwarding node		✓		✓	
			Neighbor (Neigh) identification fields		
Certificate (*Cert*) of querying node	*		*		
Certificate of forwarding node	✓	✓			
Root of forwarding node	✓	✓			
Initialization vector (*IV*) of forwarding node	✓	✓			
Geographical position (*Geo*) of querying node	✓	✓	*		
Geographical position of forwarding node	✓	✓			
Encrypted group transient key (*GTK*)		*		*	
Signature (*Sign*) of forwarding node	✓	✓			
			Neighbor authentication fields		
Secret (*Sec*) of forwarding node			✓	✓	✓
Authentication path (*Auth*) of forwarding node's secret			✓	✓	✓
Hash of message fields			✓	✓	✓

revocation list nor the location of the certification authority. At this stage of the network setup, it is assumed that also the mesh routers have a stable route to the CA/KDC, since they are typically turned on before disposing them (near to the gateways), thereby, they have a very good connection to the gateways. For the secure und fast communication between the mesh nodes and the CA/KDC we proposed in [18] an efficient single sign-on solution called Role integrated Certificate-based Single Sign-On (RC-SSO). This solution is based on the SSL/TLS communication procedure with certificates. Hereby, the certificates are integrated with roles, which reflect predefined mesh nodes' type (either router or gateway). Simulation and experimental results show that RC-SSO outperforms the widely spread Security Assertion Markup Language (SAML) by up to 80 %- *Implementing this solution makes PASER robust, among others, against malign attacks executed on the communication link to the CA/KDC.*

The *UNREGISTERED* state is mainly a state used at power up. Once the node has registered with the network, it is typically in one of the two sub-states, *ACTIVE* or *IDLE* of the *REGISTERED* state. *ACTIVE* is the sub-state where the node is active with transmitting and receiving PASER messages. *IDLE* is a low activity sub-state in which the node sleeps in order to reduce battery consumption. Note that a mesh router in *REGISTERED* state must always maintain a route to a gateway. That is, when the route to the gateway is not valid anymore; it has to restart a route discovery for the gateway.

Table III
ROUTING AND NEIGHBOR TABLE FORMAT

			Routing Table			
AddL	Dest	Seq	GFlag	Cert	NextHop	Metric
			Neighbor Table			
	Neigh	NeighFlag	Root	IV	Position*	

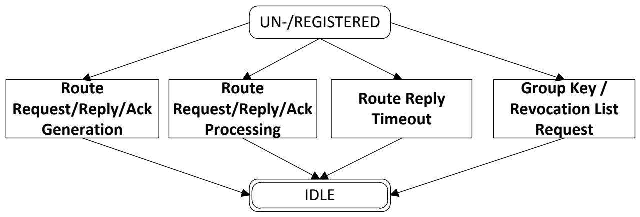

UN-/REGISTERED
Route Request/Reply/Ack Generation
Route Request/Reply/Ack Processing
Route Reply Timeout
Group Key / Revocation List Request
IDLE

Figure 4. Node Lifetime Operations

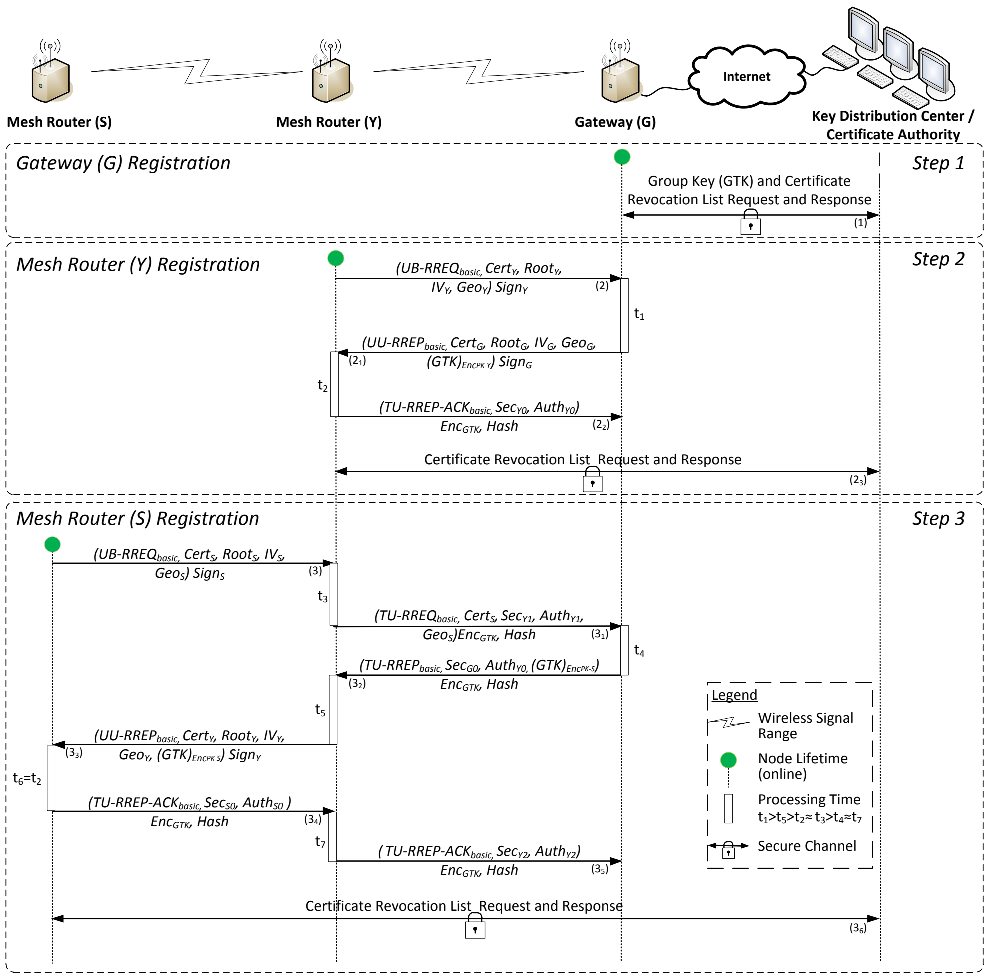

Mesh Router (S)
Mesh Router (Y)
Gateway (G)
Internet
Key Distribution Center / Certificate Authority
Gateway (G) Registration
Step 1
Group Key (GTK) and Certificate Revocation List Request and Response
(1)
Mesh Router (Y) Registration
Step 2
$(UB\text{-}RREQ_{basic,}\ Cert_Y,\ Root_Y,\ IV_Y,\ Geo_Y)\ Sign_Y$
(2)
t_1
$(UU\text{-}RREP_{basic,}\ Cert_G,\ Root_G,\ IV_G,\ Geo_G,\ (GTK)_{EncPK\text{-}Y})\ Sign_G$
(2_1)
t_2
$(TU\text{-}RREP\text{-}ACK_{basic,}\ Sec_{Y0},\ Auth_{Y0})\ Enc_{GTK},\ Hash$
(2_2)
Certificate Revocation List Request and Response
(2_3)
Mesh Router (S) Registration
Step 3
$(UB\text{-}RREQ_{basic,}\ Cert_S,\ Root_S,\ IV_S,\ Geo_S)\ Sign_S$
(3)
t_3
$(TU\text{-}RREQ_{basic,}\ Cert_S,\ Sec_{Y1},\ Auth_{Y1},\ Geo_S)Enc_{GTK},\ Hash$
(3_1)
t_4
$(TU\text{-}RREP_{basic,}\ Sec_{G0},\ Auth_{Y0,}\ (GTK)_{EncPK\text{-}S})\ Enc_{GTK},\ Hash$
(3_2)
t_5
$(UU\text{-}RREP_{basic,}\ Cert_Y,\ Root_Y,\ IV_Y,\ Geo_Y,\ (GTK)_{EncPK\text{-}S})\ Sign_Y$
(3_3)
$t_6=t_2$
$(TU\text{-}RREP\text{-}ACK_{basic,}\ Sec_{S0},\ Auth_{S0}\ )\ Enc_{GTK},\ Hash$
(3_4)
t_7
$(\ TU\text{-}RREP\text{-}ACK_{basic,}\ Sec_{Y2},\ Auth_{Y2})\ Enc_{GTK},\ Hash$
(3_5)
Certificate Revocation List Request and Response
(3_6)
Legend
Wireless Signal Range
Node Lifetime (online)
Processing Time $t_1>t_5>t_2\approx t_3>t_4\approx t_7$
Secure Channel

Figure 5. Node Registration Process

E. PASER Operations

In this Subsection, we elaborate all the operations a node undergoes when it executes PASER. These operations are depicted in Figure 4. To ease their understanding, we refer our explanation to a simple example given in Figure 5. The example illustrates three nodes, one gateway (G) and two mesh routers (Y) and (S). These nodes join the network in the order G-Y-S, which corresponds to the depicted steps 1, 2 and 3 respectively.

1) Route Request/Reply/Ack Generation:

- UB-RREQ: This message is generated if and only if a node has no route to the destination. The node creates a UB-RREQ message according to Table II. Hereby, it sets the gateway flag to 1 if the requested destination is a gateway. After creating the message, the node broadcasts it and initializes a RREQ-TIMEOUT timer. Messages (2) and (3) in Figure 5 provide an example of a UB-RREP.
- TU-RREQ: After receiving a UB-RREQ, an intermediate node, that has a route to the destination, generates this message and sends it to the next hop on that route, e.g., message number (3_1) in our example. Hereby, the querying identity remains the UB-RREQ originator identity, whereas *Seq* changes, since the message type has changed.
- UU/TU-RREP: Upon receiving a UB-RREQ or a TU-RREQ, a destination node generates a UU-RREP or a TU-RREP, respectively, e.g., messages (2_1) and (3_2). If the destination is a gateway and the *GFlag* in RREQ is set, the RREP message comprises the group key encrypted with the querying node public key.
- TU-RREQ-ACK: This message is generated by a querying node when it receives a route reply to a query identified by *Seq*. It creates the TU-RREQ-ACK message and sends it to the next hop on its route to the destination, as in messages (2_2) and (3_4).

2) Route Request/Reply/Ack Processing: Based on the querying node Identity and the message sequence parameter *Seq*, a node verifies the freshness of a received message. If it has been previously processed, the message is discarded (replay attack). Otherwise, it extracts the identity of its predecessor and executes the following verifications:

- UB-RREQ/UU-RREP:

  Is the predecessor the owner of the included certificate?
  Is the predecessor in my signal range? Is the difference of our geo-positions smaller than the maximum range of my WLAN device?
  Is the predecessor's message signature valid?

- TU-RREQ/RREP/RREP-ACK

  Is the predecessor a neighbor of mine?
  Is the predecessor's secret fresh?
  Is the predecessor's secret valid?

If one of these verifications fails, the node drops the message (impersonation attack, man-in-the-middle attack, tempering attack or wormhole attack). Otherwise, it updates its tables with the message information. Hereby, it sets the neighbor flag *NeighFlag* of its predecessor to 0 if the received message is a UB-RREQ and the predecessor hasn't been registered yet as a neighbor. Otherwise the *NeighFlag* of the predecessor is set to 1. Depending on the type of the received message, the node afterwards undergoes the following steps:

- UB-RREQ: It checks if it has a route to the destination, if not it updates the message with its own information (e.g., it adds its identity to the route list) and broadcasts it again. Otherwise, it generates a TU-RREQ as described above, e.g., message (3_1).
- TU-RREQ/RREP-ACK: The node updates the message with its own information and forwards it to the next hop on its route to the destination, e.g., message (3_5).
- UU/TU-RREP: It extracts the successor identity and verifies the value of its *NeighFlag*, if it is 0, it forwards a UU-RREP to that node, e.g., message (3_3)and otherwise it forwards a TU-RREP.

3) Route Reply Timeout: This operation occurs at the querying node when the RREQ-TIMEOUT timer expires. The latter happens in either of the following cases: First, no replies from destination, in response to the query, were received or accepted by the querying node, or, second, at least one reply was accepted. In the former case the route discovery is considered failed, while, in the latter case, the route discovery concludes, and the querying node ignores route replies that are further delayed.

Route Discovery Failure: The querying node initiates a new route discovery using a higher value for RREQ-TIMEOUT than the one previously used for the failed route discovery.

Route Discovery Conclusion: Upon accepting a RREP, the querying node considers the discovery concluded after RREQ-TIMEOUT elapses. From all incoming RREP, the querying node always chooses the best route based on the metric field and updates its tables with this route. If the querying node is in the UNREGISTERED state and the discovered route is a route to the gateway, it requests a certificate revocation list via the discovered route, as in messages (2_3) and (3_6). Based on that list, the node verifies if it has fraud routes and deletes them. Afterwards, the node switches to the REGISTERED state.

4) Group Key/Revocation List request: Both requests occur at the end of the node registration phase. It is assumed at this stage of the network setup, that all nodes have a stable route to the CA/KDC. The gateways are anyway provided by a reliable link, e.g., Long Term Evolution (LTE) or satellite, and the mesh routers are typically located during power up near to the fire-fighting command and control vehicle, i.e., they are in the best signal range of the gateways. While the group key request solely occurs at the gateway, message (1), revocation list request occurs at both types of nodes, mesh router and gateway, see messages (1), (2_3) and (3_6). PASER rather specifies the security goals that must be ensured by these requests than the mechanism used. These goals are authenticity and integrity by both requests in addition to confidentiality by the group key request.

V. PASER Analysis

Based on a hop-to-hop trusted relation, PASER promises to achieve the following goals:

Node authentication: This goal is guaranteed by the digital signature in untrusted messages (including revocation list messages) and by the hash tree authentication mechanism in trusted messages - *PASER is robust against impersonation and malign attacks.*

Message freshness and integrity: The freshness goal is provided by the sequence number included in each message. The integrity is achieved by the digital signature in untrusted messages and by the hash element in trusted messages - *PASER is robust against replay and tempering attacks.*

Messages confidentiality: It corresponds to the symmetric encryption of trusted messages, which is mainly applied to combat man-in-the-middle attack. Then, theoretically, an attacker located between two neighbors is able to eavesdrop on trusted messages and prevent the destination from receiving them. As a result, it uses the secrets of these messages to impersonate the messages' sender. Now, due to the encryption in trusted messages, the attacker is not able to reveal those secrets. Apart from that, message confidentiality of trusted messages strongly reduces traffic analysis in PASER. In untrusted messages man-in-the-middle attack is not possible due to the digital signature - *PASER is robust against man-in-the-middle attacks.*

Neighbor transmission authentication: Provided satellite GPS information is not falsified, PASER guarantees to a large extent that node's neighbors are always in that node transmission range. This goal is provided by the fault tolerant distance awareness between new neighbors combined with the achievement of the node authentication goal. - *PASER is robust against wormhole attacks.*

From efficiency perspective, PASER incorporates the following characteristics:

- Nodes always have a route to a gateway.
  - Nodes thereby detect all intermediate nodes on that route.
  - The route is found and maintained in a reactive way. Gateways do not flood the network with beacons.
- It is mainly based on unicast messages, strongly reducing the network overhead of control messages.
- Its security is essentially based on symmetric cryptography, keeping the cost of security mechanisms minimal.

VI. Conclusion and Future Work

In this paper we propose a novel secure and efficient position aware hierarchical route discovery protocol for wireless mesh networks. From a security perspective, the novelty of our approach is its hybrid scheme to secure the route discovery process. This novel combination of digital signature, hash tree authentication scheme and symmetric block cipher yields a huge performance gain while providing a high security level. Another key feature is the integration of nodes' geo-positions in the route discovery, allowing an advanced network management while mitigating a wider range of attacks. Apart from that, dealing with the network as a hierarchical network and building the route discovery process to a large extent upon unicast messages strongly decreases the overhead of this protocol.

In future work we intend to capture explicitly the inherently quantitative nature of security, via a concrete or exact treatment of security using practice-oriented provable security. This enables an exact assessment of how much security the protocol achieves rather than just being secure or non-secure. Furthermore, we designate to thoroughly investigate the performance of PASER in different scenarios experimentally as well as in the simulation to recognize its advantages and its limitations. Apart from that, we intend to analyze the energy consumption imposed by PASER especially by the GPS component it incorporates. Besides, we intend to extend PASER to a route maintenance part and thereby to design an efficient secure routing protocol for wireless mesh networks.

Acknowledgment

Our work has been conducted within the SPIDER project, which is part of the nationwide security research program funded by the German Federal Ministry of Education and Research (BMBF) (13N10238).

References

[1] (2011, May) Security System for Public Institutions in Disastrous Emergency scenaRios (SPIDER). [Online]. Available: http://www.spider-federation.org

[2] A. Wolff, S. Šubik, and C. Wietfeld, "Performance Analysis of Highly Available Ad hoc Surveillance Networks Based on Dropped Units," in *Proc. IEEE International Conference on Technologies for Homeland Security*, Boston, USA, May 2008, pp. 123–128.

[3] (2011, May) Highly Mobile Network Node (HiMoNN). IABG mbH. [Online]. Available: http://himonn.iabg.de/index.php?lang=en

[4] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc On-Demand Distance Vector (AODV) routing," RFC 3561, Jul. 2003.

[5] I. Chakeres and C. Perkins, "Dynamic MANET On-Demand (DYMO) Routing," draft-ietf-manet-dymo-21, Jul. 2010.

[6] T. Clausen and P. Jacquet, "Optimized Link State Routing (OLSR) Protocol," RFC 3626, Oct. 2003.

[7] K. Sanzgiri, D. Laflamme, B. Dahill, B. N. Levine, C. Shields, and E. Belding-Royer, "Authenticated Routing for Ad hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 23, pp. 598–610, Mar. 2005.

[8] Y. Hu, A. Perrig, and D. Johnson, "Ariadne: A Secure On-Demand Routing Protocol for Ad hoc Networks," *ACM Journal on Wireless Networks*, vol. 11, no. 1-2, pp. 21–38, Jan. 2005.

[9] L. Buttyán and I. Vajda, "Towards Provable Security for Ad hoc Routing Protocols," in *Proc. 2nd ACM Workshop on Security of Ad hoc and Sensor Networks (SASN)*, Washington DC, USA, Oct. 2004, pp. 94–105.

[10] M. Burmester and B. de Medeiros, "On the Security of Route Discovery in MANET," *IEEE Transactions on Mobile Computing*, vol. 8, no. 9, pp. 1180–1188, Feb. 2009.

[11] L. Abusalah, A. Khokhar, and M. Guizani, "A Survey of Secure Mobile Ad hoc Routing Protocols," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 78–93, 2008.

[12] *IEEE P802.11s*, IEEE Draft Amendment: Mesh Networking, Rev. 10.0, Mar. 2011.

[13] (2011, May) Better Approach To Mobile Ad hoc Networking (B.A.T.M.A.N.). Freifunk Community. [Online]. Available: http://www.open-mesh.org/

[14] *Digital Signature Standard (DSS)*, National Institute of Standards and Technology (NIST) Std. FIPS PUB 186-3, Jun. 2009.

[15] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An Ultra-Lightweight Block Cipher," in *Proc. 9th Workshop on Cryptographic Hardware and Embedded Systems (CHES) 2007*, ser. Lecture Notes in Computer Science (LNCS), no. 4727. Springer-Verlag, 2007, pp. 450–466.

[16] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996, ch. 13, p. 556.

[17] (2011, May) Cryptographic Hash Algorithm Competition. National Institute of Standards and Technology (NIST). [Online]. Available: http://csrc.nist.gov/groups/ST/hash/sha-3/index.html

[18] T. Tran, M. Sbeiti, and C. Wietfeld, "A novel Role- and Certificate-based Single Sign-On System for Emergency Rescue Operations," in *IEEE International Conference on Communications - ICC*, Jun. 2011, pp. 1–6.

Reliablity and Survivability of Wireless Sensor Network Using Security Adaptation Reference Monitor (SARM)

Tewfiq El Maliki
Information Technology department-hepia
University of Applied Sciences Western Switzerland
1202 Geneva - Switzerland
tewfiq.elmaliki@hesge.ch

Jean-Marc Seigneur
Advanced Systems Group
University of Geneva
1211 Geneva 4 – SWITZERLAND
Jean-Marc.Seigneur@trustcomp.org

Abstract* —Security has become a key issue for any huge deployment of Wireless Sensor Network (WSN). Moreover, data reliability combined with energy loss minimization is really a challenging task, particularly to maintain survivability of the WSN under attacks such as sinkhole. Therefore, new security mechnisms must be in accordance with energy consumption constraint. This paper proposes to address this task using our Security Adaptation Reference Monitor (SARM) which is an efficient Framework capable of trading-off between security and energy optimization. SARM is based on an autonomic computing security looped system, which fine-tunes security means based on the monitoring of the context including energy consumption aspects. We evaluate SARM in the context of WSN through a simulation tool to verify the performance of overall reliability and energy loss in the presence of sinkhole attackers. The results clearly show that SARM is efficient in terms of reliability, overall network utilization and power consumption.

***Keywords* – Framework, Autonomic, Security adaptation, Sinkhole, Sensor Network**

I. Introduction

Wireless sensor network (WSN) is a versatile network for supporting variety of important applications, consisting of a large number of low-power and multifunction sensor nodes that communicate as one hop, multi-hop or cluster-based models to send data to one or many base stations (BS)s through wireless links [1]. The BS is highly enriched system with a large amount of energy. This network is built by deploying the sensing nodes in the area of interest to form a self configured network and start acquiring the necessary information. The nodes in this network are battery operated and have limited lifetime to operate. Therefore, there is a need of energy aware security algorithm which should not perform heavy computation on the nodes since it shortens the network lifetime.

In general, many applications could not operate under significant packet loss. Thus, reliability is one of the most important criteria to evaluate the quality of wireless sensor networks. Unfortunately, packet loss is increased by two major factors: less coverage of sensors due to less power and high error rate of wireless links. Moreover, dynamic power attacks such as sinkhole are fatal to the survivability of the network. Therefore, the concept that must cope with this new security challenge has to be based on dynamic adaptation security system to satisfy an overall performance such as network reliability, being a key issue especially in sensor networks. We have already proposed a generic security adaptation reference monitor (SARM) as a compelling solution for such problems [2]. In this article, we will apply it for WSN in case of sinkhole attacks.

Please note: we use security in general term including availability, reliability and survivability.

In Section 2, we survey other related works. Section 3 gives the problem statement, highlighting the motivation of our work. Section 4 introduces SARM for WSN and explains its components and functionalities. Section 5 explains our experiments and simulation implementation to validate SARM in the case of sensor network. Our simulation results and performance analysis are presented in Section 6 and Section 7 concludes our paper.

II. Related Work

Many systems rated at the higher levels of security for data are implemented according to the reference monitor concept. First introduced by James Anderson [3], a reference monitor is a concept that has proven to be a useful tool for computer security experts. It is the only effective tool known for describing the abstract requirements of secure system design and implementation.

A suitable security service is provisioned in a progressive way to achieve the maximum overall security services against network performance services throughout the course of sensor networks operation. Security in sensor networks is complicated by the constrained capabilities of sensor node hardware and the properties of the deployment [4], [5] and [6].

We argue that the spare processing and transmission resources are wasted in sensor environments if security is over-provisioned. Hence the trade-off between security and performance is essential in the choice of security services. Adaptive security mechanisms are also found in flexible protocol stacks for wireless networks [7], context-aware access control systems [8] and security architectures [9]. This prompted us for the implementation of a completely reconfigurable architecture [10], which is fundamental to adapt the architecture to the terminal and network variability of the context and particularly in the security field [11]. J-M

Seigneur [12] has introduced autonomic security pattern in his security design but only at the authentication level.

III. Motivation for our Framework

Flexible security mechanisms are needed to respond to new types of attacks and to meet different network setting-specific protection requirements. The required flexible security assessment can be achieved by introducing a generic autonomic computing security framework.

In the case of sensor networks, the sensors usually forward their messages to a Base Station (BS) [13] in a hop-by-hop fashion because they are resource-constrained in terms of energy and the spending of energy dramatically increases with the range of transmission. It is quite easy for an attacker as a Sinkhole [14] to defeat the WSN purpose by dropping messages when received rather than forwarding them to consume energy of other sensors by requesting them to continuously send information.

It is highly critical to keep the overall security at the highest level due to the configuration complexity and the runtime changing context. In general, data transfer in WSNs is more susceptible to loss due to the nature of sensors (power, processing, etc) in addition to the high error rate of wireless links. Moreover, sinkhole attacks by means of dynamic changing behavior skyrocket the packet loss. Therefore, the most crucial constraint in WSN which is reliability is not at all guaranteed

Assuring reliable data delivery between the sensor nodes and the BS in Wireless Sensor Networks is a challenging task as it affects the ability to sense event. A reliable protocol in WSN is a protocol that allows reliably data transfer from source to BS with reasonable packet loss. The problem of achieving reliable communication between nodes is further aggravated by the presence of sinkhole attackers whenever they are changing dynamically their behavior.

In addition, most applications cannot operate in case of high packet loss. Thus, reliability, being a key issue especially in sensor networks, is definitely one of the important criteria to evaluate the quality of wireless sensor networks. Accordingly, the concept that must cope with this new security challenge in term of availability has to be based on dynamic adaptation security system to satisfy an overall performance such as network reliability and energy loss.

Briefly, to lengthen the lifetime of wireless sensor network, an efficient protocol needs to support reliable network combined with energy efficiency under sinkhole attacks.

We propose a generic Framework called Security Adaptation reference monitor (SARM) as a compelling solution for this problem, because it is looped system developed especially for highly dynamic wireless network. It is aimed to offer a global adaptation security scheme for any application instead of a classical layered security mechanism.

Implementing this security scheme at each application level is not feasible because the change will interfere in each communication program in each sensor. The best way to overcome this constraint is to implement it in the kernel that leads to an overall security control.

IV. SARM Description

We would like with SARM to fine-tune security means as best as possible taking into account the risk of the current application environment and the performance of the system especially regarding the optimization of its energy consumption. Thereby, our system differs from others by its [2]:

a) Autonomic computing security looped system

b) Dynamic and evolving security mechanisms related to context-monitoring

c) Explicit energy consumption management

The concept of isolating various functions and restricting their access to specific system can also be applied to security in wireless environment integrated in the operating system itself. The best way to overcome the non realistic constraint of implementing the framework in each communication program is to integrate it in the kernel and consequently having an overall security control. Thus, all communication programs go through SARM at some stage in order to gain access to communication resources.

The key challenge of SARM is the adaptation of Reference Monitor (RM) [3] concept for wireless communication and beyond data access control. The goal of a RM is to enforce security by forcing all processes and also to prevent applications from accessing any data but only through the reference itself. The security kernel is managed by security policies. We have also chosen to apply the autonomic computing security pattern [15] to design SARM by dividing it into a functional unit and a monitoring unit. In addition, localized trust [17] or distributed trust [18], [19] and [20] are good paths to explore because in some cases they generate low computing charge (less energy consumption) and give better results. Thereof, we are fitting perfectly the context of WSN.

In [2], we could find all information about SARM high-level components view.

A. WSN- SARM

To validate SARM, we have applied an adapted version of SARM, called WSN-SARM, to the application domain of wireless sensor network.

1) Application Domain Main Problem

In Wireless Sensor Networks (WSN), one of the main constraints is to minimize energy consumption in order to maximize the lifespan of the network.

We send messages to the BS in a hop-by-hop routing method. While this method searches to minimize the overall network utilization of energy, since the power cost is in function of distance to the power of a parameter ranged from 2 to 5.

This heavy load of traffic on nodes near the BS brings them to deplete their energy rapidly. Thereby, it is a bottleneck region for the network. Unfortunately, when too many of those nodes run out of energy, the BS becomes disconnected from the network, and putting the network down while there may be plenty of energy remaining in nodes away from the BS. Therefore, it seems that energy load balancing is a particularly promising way of maximizing the survivability of the network.

Another problem that challenges all proposed solution is sinkhole attack which is a node that does not retransmit any received packet.

The goal of this validation is to show that SARM adapts security as efficiently as possible by:

a) keeping an appropriate level of security depending on the context ;

b) whilst maximizing the overall reliability;

c) and minimizing the power consumption.

2) Metrics

Energy metrics are Packet loss ratio that affects energy loss per node and the whole network energy loss which is important to evaluate energy efficiency at transport protocol for any application.

Assuming dropped packets have a direct relation with energy depletion, the energy loss per node can be measured by [16]:

$$E(i) = \frac{\text{nbr of packets dropped by node}}{\text{total nbr packets received by node}}$$

Whereas the energy loss for the whole network can be calculated by total number of packet received by:

$$F_{\text{network}} = \frac{\text{nbr of packets dropped by the network}}{\text{total nbr of packets received by BS}}$$

Reliability of the entire network is defined as:

$$R_{\text{network}} = \frac{\text{nbr of packets received by BS}}{\text{total nbr packets Send by all nodes}}$$

We can show easily that $R_{\text{network}} = 1/(E_{\text{network}} + 1)$

3) WSN-SARM Description

In Fig. 1, we describe module by module, how SARM is applied to the application domain of our validation, becoming the WSN-SARM version.

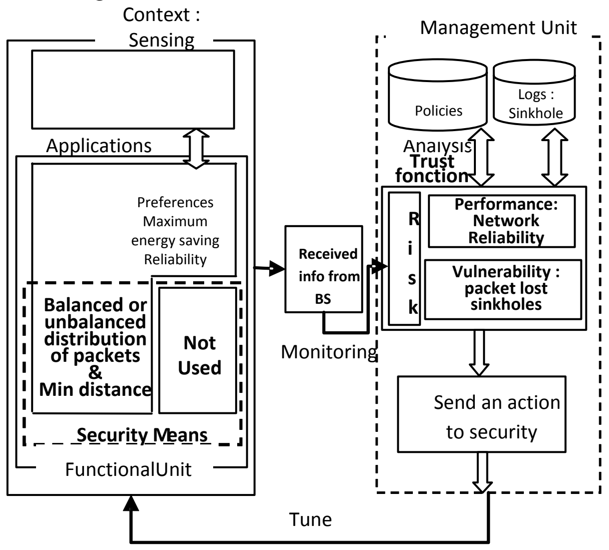


Figure 1. WSN-SARM Modules

First of all, the security means, which can be tuned by SARM, are uniform packet repartition or unbalanced neighbors packet repartition or a set of suboptimal routing paths. The application preference is to maximize the usage time whilst keeping enough security. The gathering context module is used to collect and distribute trust values between the Base Station and Nodes (sensors). These values represent the trust of a sensor about its neighbors. They are summarized in Table I.

TABLE I. BEHAVIOR AND RECOMMENDED VALUE SENT BY BASE STATION TO SENSOR UNDER SINKHOLE ATTACK

Sensor Behavior over neighbors	Recommended value to Sensor
Normal	The packet is received (1)
Sinkhole to neighbors' by not sending packet	The packet is lost (-1)

The values are sent to the management unit for analysis using a Trust Function (TF) that will assert the fact which algorithm has to be used. In addition, the performance is fixed as energy saving in accordance with Application Preference, which is lifespan maximizing.

Each Sensor sends packets uniformly to a number of Sensors within a define range according to thresholds used as policy. Thanks to its context gathering module the TF has all information to evaluate the trust.

The management unit will integrate the Trust Function TF that predicts whether or not to use uniform or unbalanced connections depending on the **output** of the TF depending on historical **values** $v_{i,j}$ (i packets) sent by the BS to sensor z about his neighbor sensor j within defined range.

- $T_j^z(v_{i,j}) = \frac{\sum_{i=1}^{N} v_{i,j}}{N}$ [$T_j^z(v_{i,j})$: trust of sensor z in sensor j and v_i are sent by BS as ACK, N : number of all packets sent by sensor z and received by BS]
- Threshold = rand()

```
For all j sensors
    if ($T_j^F(v_{i,j}) > 0$
        TF is the summation of all positive Trust over j
        neighbors
        if (TF ==0)
            then {we send uniformly}
        else{TF> **threshold**}
            then {we send the packet to sensor j}
End for
```

V. IMPLEMENTATION AND VALIDATION METHODOLOGY

We have implemented WSN-SARM and validated it in a Sensor wireless network simulation developed with AnyLogic, which is a simulation tool that supports all different simulation methodologies: System Dynamics, Process-centric, and Agent Based modeling. It is based on Real-time UML and Java object-oriented language.

A. Model Set-up

Setting up our security model using table 1, one can take advantage of state charts to control the behavior of Sensors. Using AnyLogic as implementation platform agents and especially state-charts can be programmed very conveniently. In particular modifications and/or extensions of the final model can be handled in a simple way.

In Fig. 2, each Agent (Sensor) starts simultaneously in a "Transmission" state in the "SensorStateR" and "Trust function" state-charts. The Agents are switched to their relative state (Sinkhole, Base Station, Sensors). They are then added to a list of the sensor whenever they are within his range.

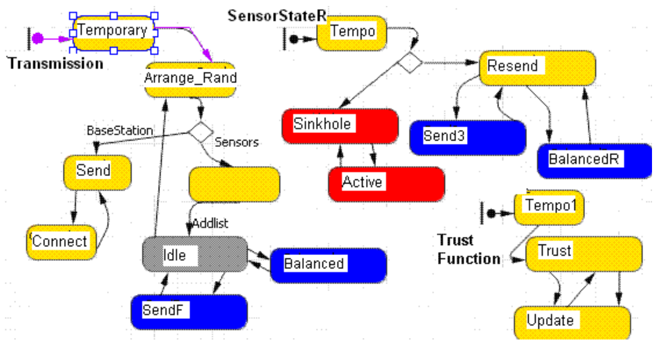


Figure 2. State-charts: "Transmission" of agent "SensorStateR" and "TF"

In Fig. 2, each Agent (Sensor) starts simultaneously in a "Transmission" state in the "SensorStateR" and "Trust Function" state-charts. The Agents are switched to their relative state (Sinkhole, Base Station, Sensors). They are then added to a list of the sensor whenever they are within his range.

We used Agents having one of the following behaviors:

a) Normal state and

b) Sinkhole

Each Agent is then processed depending on the decision of the monitor unit to choose a security mean or not. Therefore, the Agent transits to another state depending on the transition conditions or stand in the same state. When completing the transfer, the Agent returns to its initial state and so on. The state-chart Trust update the trust each time the Base Station sends an Ack. Of course, the BS is not limited in energy and thus is not subject to any sinkhole attack.

B. Validation Methodology

In our experiments, we have validated our proposed solution and analyzed the extended performance under a range of various scenarios.

We have carried out simulations under 0%, 20% and 50% sinkhole attackers. Furthermore, the network topology was set to random spreading or arranged uniform spreading of sensors. We have taken as a reference uniform packet distribution over the neighbors. In addition, a Time-To-Live TTL counter is used to avoid that a packet stay forever in the network and to guarantee that the consumed energy is limited to a maximum value when a packet is sent from the farthest sensor to the BS.

To minimize the transit delay and the energy consumption, we have also introduced suboptimal routing paths as paths that have the shortest Euclidian distance to the BS. Indeed, if the topology of sensors is uniformly distributed and the sensors aren't in the border of the square, there are 3 possible sensors that have the shortest distance to the BS.

Normally, the BS is in the middle of the network to minimize the distance to the farthest sensor. Additionally, 90 degree sector antennas are used to cover each of four squares to lengthen the BS range and minimizing the energy consumption. Sector directional antennas can be also added to sensors to take advantage of this technique in term of energy consumption [21] Therefore, we do not lose any generality if we put the BS in the upper left side of the square; rather we gain in survivability of WSN.

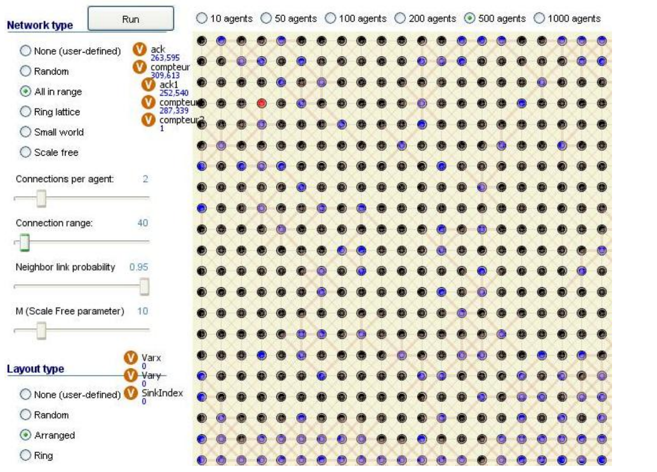

Network type: [✓ All in range]
ack 263,595
compteur 309,613
ack1 252,540
compteur 287,339
compteur2 1
Connections per agent: 2
Connection range: 40
Neighbor link probability 0.95
M (Scale Free parameter) 10
Varx 0
Vary 0
SinkIndex 0
Layout type: [✓ Arranged]
[✓ 500 agents]

Figure 3. Animation interface of Arranged WSN-SARM

Fig. 3 shows a very powerful animation interface using AnyLogic. The BS is placed in the upper left side of the square.

Arranged sensors means that they are placed in an equidistant manner as depicted in Fig. 3. Random sensors repartition means that the sensors are physically placed in a random manner.

All sensors are over spread over a square topology of 520m side length, and operating over one day of simulation time. In our simulations, we considered that the Base station was taken at the origin. The coverage of the Base Station is over the entire network. We fix the connection number of neighbors from 1 to 7. Indeed, depending on the topology of the network (arranged or random distributed sensors positions), each sensor was configured to have a maximum communication range equal to 50 meters. We deployed the Sensors in an incremental mode, from S_1 to S_n. The number of sensors can be selected from 10 to 1000 and their arrangement can be selected between arranged uniformly or randomly.

VI. RESULTS ANALYSIS

During our analysis, we firstly studied the performance of WSN-SARM in the defined scenarii where sensors were arranged uniformly or at random. The performance metrics are network Reliability Ratio and overall network Energy loss within the constraints:

a. Thanks to TTL almost the same average energy consumption for any packet and
b. Balancing overall traffic over all the neighbors to guaranteed the network survivability.

Secondly, we studied long-run convergence of TF used in WSN-SARM.

We have depicted in Fig.4 and Fig. 5 the results of the simulation of WSN-SARM and uniform traffic balancing under respectively 0%, 20% and 50% of sinkhole attackers. We can easily conclude that SARM is largely better than uniform balancing. A ratio of 10 is reached within short time. Indeed, we have the obtained the desired effect of the feedback mechanism and Trust Function implemented in WSN-SARM.

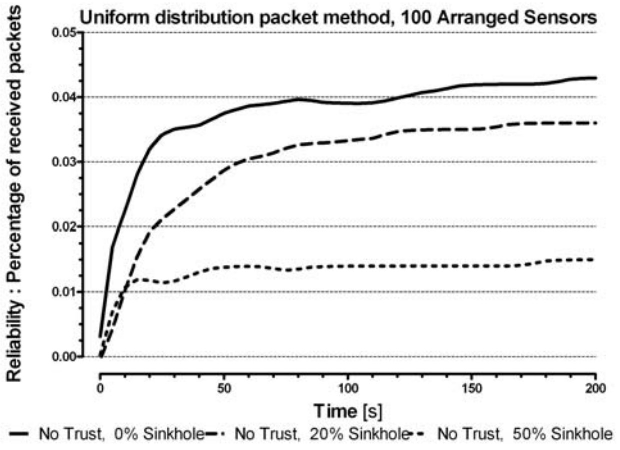


Figure 4. Reliability of WSN-SARM under different sinkhole attacks.

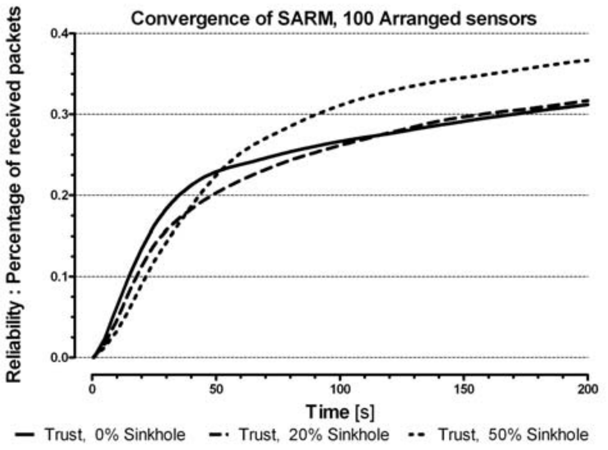


Figure 5. Reliability of WSN-SARM under different sinkhole attacks.

For comparison purpose, we plotted the WSN-SARM under 20% of sinkhole attackers using our Trust Function and without trust (No Trust) in Fig. 6. We have used all suboptimal routing paths to the Base Station. The results clearly demonstrate that the convergence is boosted to reach 100% of Network Reliability.

Remark: WSN-SARM constitutes a good algorithm to detect any sinkhole with the help of the Base Station and eliminates it from its connections. We can see that when the Sinkhole attackers are detected and inhibited by the message sent by BS to all sensors, the reliability of the network is raised and especially in case of 50% sinkhole attackers (many attackers) get a significant step for its convergence. Therefore, simulation shows that our Framework is efficient in this context and is tuning to achieve the best trade-off between security in one side and, energy loss and reliability in other side.

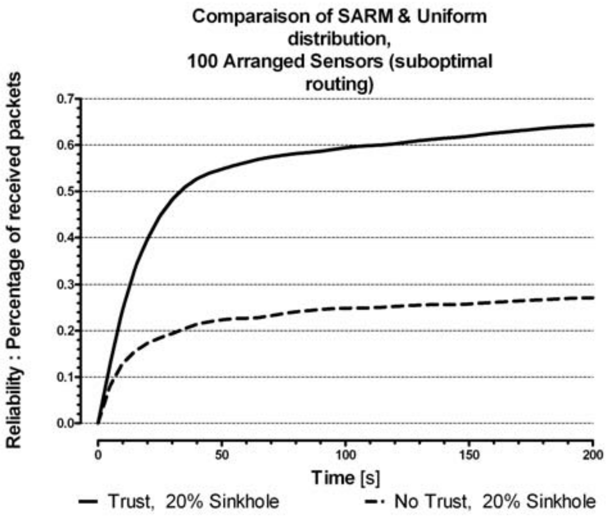


Figure 6. Reliability of WSN-SARM and reference.

We have noticed that there are significant differences between Trust Function used by WSN-SARM and uniform packet distribution reference in the case of arranged Sensors.

We have an average ratio of 2.4 between the two cases. The convergence is also boosted for WSN-SARM.

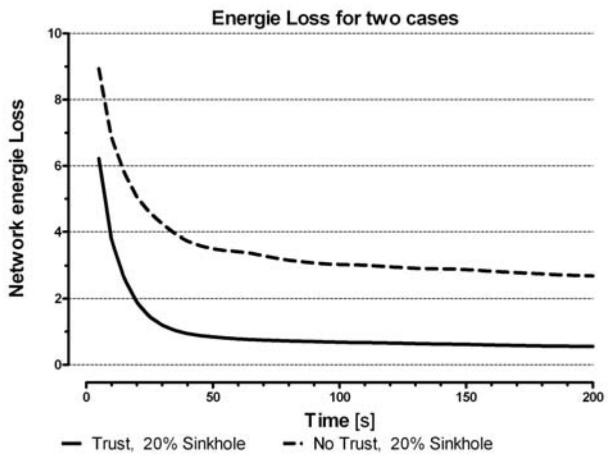


Figure 7. Network Energy Loss for WSN-SARM and uniform balancing

We have depicted in Fig. 7 energy loss of WSN-SARM using Trust Function and a reference case without trust under 20% of sinkhole attackers. The convergence is rapid and the overall Energy loss is very rapidly minimized within WSN-SARM.

Since long-run simulation has a Network reliability of 1 (estimated with an error of less than 0.1%), the system convergence is guaranteed.

All the results show clear advantages of WSN-SARM under sinkhole attackers thanks to the looped system and the Trust Function efficiently.

VII. CONCLUSION AND FUTURE WORK

We have proposed a Security Adaptation Reference Monitor (SARM) based on the Reference Monitor concept and the Autonomic Computing Security pattern to support both context monitor and behavior control. The results show that WSN-SARM copes with reliability and network Energy loss under sinkhole attack even at 50% of attackers. Indeed, WSN-SARM constitutes a good Platform to detect within the Base Station any sinkhole and eliminates it from its connections. The results clearly show that our platform copes with reliability and security of the network under sinkhole attack, by efficiently tuning the adequate means whilst minimizing energy loss.

These results encourage us to further research on other strategies that could automatically optimize the trade-off between security and energy consumption in other important application domains, such as mobile wireless sensor networks under other attacks.

REFERENCES

[1] J. Ibriq and I. Mahgoub, “Cluster-based routing in wireless sensor networks: Issues and challenges” Proceedings of the International Symposium on Performance Evaluation of Computer and Telecommunication Systems, July 25-29, 2004, San Jose, California, USA

[2] T. El Maliki and J.-M. Seigneur “A Security Adaptation Reference Monitor (SARM) for Highly Dynamic Wireless Environments” The International Conference on Emerging Security Information, Systems, and Technologies SECURWARE July 2010

[3] J. Anderson, “Computer Security Technology Planning,” http://seclab.cs.ucdavis.edu/projects/history/papers/ande72.pdf, 1972.

[4] Hiren Kumar Deva Sarma, Avijit Kar, “Security Threats in Wireless Sensor Networks”, IEEE 2006

[5] E. Shi and A. Perrig, “Designing Secure Sensor Networks,” Wireless Commun. Mag., vol. 11, no. 6, pp. 38–43, Dec. 2004

[6] Yun Zhou; Yuguang Fang; Yanchao Zhang, “Securing Wireless Sensor Networks: A Survey”, IEEE Communications Surveys & Tutorials, Vol:10, Issue 3, PP: 6 –28, Third Quarter 2008.

[7] C. Hager, “Context Aware and Adaptive Security for Wireless Networks,” PhD thesis, Virginia Polytechnic Institute and State University, 2004.

[8] M. Lacoste, G. Privat, and F. Ramparany. “Evaluating Confidence in Context for Context-Aware Security,”. European Conference on Ambient Intelligence (AmI'07), 2007.

[9] J. Al-Muhtadi, D. Mickunas, and R. Campbell, “A Lightweight Reconfigurable Security Mechanism for 3G/4G Mobile Devices,” IEEE Wireless Communications, 9(2):60–65, 2002.

[10] E2R Deliverable D2.2., “Equipment Management Framework for Reconfiguration: Architecture, Interfaces, and Functions,” Dec. 2005.

[11] T. Jarboui, M. Lacoste, and P. Wadier, “A Component-Based Policy-Neutral Authorization Architecture,” French Conference on Operating Systems (CFSE), 2006.

[12] J.-M. Seigneur, “Trust, Security and Privacy in Global Computing,” PhD Thesis, 2005.

[13] Olivier Powell, Jean-Marc Seigneur and Luminita Moraru,“Trustworthy Forwarding Sensor Networks Information to the Internet” The International Conference on Emerging Security Information, Systems, and Technologies (SECURWARE 2007).

[14] Asad Amir Pirzada and Chris McDonald “Circumventing Sinholes and Wormholes in Wireless Sensor Networks” the Int. Workshop on Wireless Ad-hoc Networks, 2005.

[15] D.M. Chess, C.C. Palmer, and S.R. White, “Security in an autonomic computing Environment,” IBM Systems Journal, VOL 42, N1, 2003.

[16] Anylogic 38M. A. Rahman, A. E. Saddik and W. Gueaieb, “Wireless Sensor Network Transport Layer: State of the Art”, Sensors, Springer-Verlang Berlin Heidelberg, 2008.

[17] C. Davis, “A localized trust management scheme for ad-hoc networks,” Proc. 3rd International Conference on Networking (ICN'04), Mar. 2004.

[18] L. Eschenauer, V. Gligor, and J. Baras, “On Trust Establishment in Mobile Ad-Hoc Networks,” Proc. 10th International Workshop of Security Protocols, Springer Lecture Notes in Computer Science (LNCS), Apr. 2002.

[19] A. Rahman and A. Hailes, “A Distributed Trust Model,” New Security Paradigms Workshop 1997, ACM, 1997.

[20] X. Titi, Tewfiq EL MALIKI, Jean-Marc Seigneur, “Trust-Based Hotspot Selection” IADIS International Jounal onComputer Science and Information System. V V,2 , 2010

[21] E. Felemban, S. Vural, R. Murawski, E. Ekici, K Lee, Y. Moon, and S. Park, ”SAMAC: A Cross-Layer Communication Protocol for Sensor Networks with Sectored Antennas” Mobile Computing, IEEE Transactions on, August 2010

Federated Identity Management for Android

Anders Fongen
Norwegian Defence Research Establishment
Norway
anders.fongen@ffi.no

***Abstract*—A federated identity management system (IdM) must include mobile units and must provide mutual authentication for client-server connections. Existing frameworks for identity management like SAML are unlikely to apply well to resource constrained mobile terminals like Android. The contribution of this paper is an IdM with simpler data representation and protocols for identity management and authentication, which can be deployed with fewer code lines, consume less bandwidth and require less connectivity than traditional protocols, e.g., those based on SAML and WSSec. The related service invocation mechanisms is designed to support mobile services, where object methods in mobile units can be invoked from other nodes in the network, regardless the use of NAT units and firewalls.**

***Keywords*-Identity management, Android, Authentication**

I. INTRODUCTION

The XML protocol was once proposed as a simple replacement for SGML, which had grown very complicated and required large software stacks for processing. XML on the other hand, was simple, human readable and could be processed even with simple string operations (in case a parser was unavailable). Current XML-based standards are often seen to have lost this virtue, and are heavily dependent upon a dedicated software stack for processing, as well as they are hard to read by a human eye and even harder to verify.

Complicated protocols with many optional properties offer less interoperability than simple protocols. In the case of Identity Management (IdM) and authentication based on SAML [1] and WSSec [2] standards there is little interoperability to see, implementations only talk to themselves. This observation is based on unpublished experiments conducted by the author in order to make a Java stack and a .NET stack cooperate over a SOAP header containing WSSec and SAML data. The large number of variables (key length, key algorithm, key presentation, addressing formats etc.) were not sufficiently coordinated, the implementations were immature and software bugs added to the problem.

No one-size-fits-all solution to identity management exists, but yet it does not make sense to duplicate "stovepipe" systems in order to accommodate the different operating environments. What makes sense, however, is to differentiate in the *presentation* of the data structures involved, including the credentials used for authentication. Different presentation layers retrieve their information from the same storage of keys, roles and attributes, and put their trust upon the same identification and revocation procedures (since these procedures are costly to operate and should not be duplicated). Different presentations could be used to improve interoperability of an identity management system as well as to offer services to disadvantaged equipment.

For the purpose of offering identity management services to mobile units based on the Android platform an "extension" to an existing IdM has been built, using a presentation layer better suited for those units.

The IdM builds on existing Public Key Infrastructures (PKI) technology and storage services for user roles and attributes, and offers identity management services with related services for authenticated and encrypted service invocation. The services employ simple protocols and a thin presentation layer. The participant of the IdM (clients and service providers) exchange serialized Java objects and must therefore run on a platform with support for the Java serializing API (like Java VM, Android Dalvik, Scala etc.)

The presented IdM is built on the principles and prototype presented in [3], [4]. The contribution of this paper is to show how disadvantaged nodes and networks may be included in existing IdM systems through the provision of an adapted presentation layer. An investigation of Message Oriented Middleware like XMPP for IdM-related communication will also be presented.

The remainder of this paper is organized as follows: The next section will give a general background on Identity Management. Section III will present an outline of the GISMO IdM system. Section IV will discuss specific mechanisms related to operation across *Communitites of Interest* (COI), and Section V will present the GISMO IdMs framework for service invocation based on PDU (Protocol Data Units) with serialized Java objects. The use of a messaging protocol for improved reliability and connectivity will be discussed in Section VI, and interoperability issues related to the dual stack situation in the GISMO IdM will be discussed in Section VII. The paper provides a summary and some conclusive remarks in Section VIII.

II. MOTIVATIONAL BACKGROUND

Identity Management (IdM) are collection of services and procedures for maintaining subject information (key pair, roles) and to issue credentials for the purpose of authentication, message protection and access control. From the client perspective, the credentials issued by the IdM services enables it to access many services inside a community under the protection of mutual authentication and encryption. From the server perspective, IdM enables it to offer credentials to clients in order to provide mutual authentication.

A. Federated Identity Management

Several federated IdM schemes have been developed, some of which offer single sign on (SSO) for web clients [5], [6], [7]. The SSO protocols exploits the redirection mechanism of HTTP in combination with cookies and POST-data so that an Identity Provider (IdP) can authenticate the client once and then repeatedly issue credentials for services within the federation. This arrangement requires IdP invocation for each "login" operation, and does not offer mutual authentication, i.e., service authentication.

In the situation where the client is an application program (rather than a web browser), there are more opportunities for the client to take actively part in the protocol operations, e.g., by checking service credentials, contacting the IdP for the retrieval of own credentials, caching those credentials etc. The research efforts presented in this paper assume that the clients enjoy the freedom of custom programming.

The usual meaning of the word "federated" is that several servers share their trust in a common IdP for subject management and authentication. It does not necessarily imply any trust relationship between independent IdPs so that they can authenticate each others' clients. For the following discussion, we will call the group of clients and services which put their trust in the same IdP as a *community of interest*. A trust relation between independent IdPs is called a *cross-COI relation*.

B. Mobile and Federated IdM requirements

An essential property of an IdM is its ability to integrate with other components for management of personnel and equipment.

- An IdM should be able to use resources from the existing PKI (keys, certificates, revocation info) and offer its services to different platforms, with different presentation syntax and for different use cases.
- An IdM should also be able to tie trust relations with other IdMs in order to provide accommodation for guests and roaming clients.
- An IdM should support protocol operations for mutual authentication.

For IdM used in mobile systems, there are requirements related to the resource constraints found in these systems:

- A IdM for mobile operation must use the minimum number of protocol operation, small PDU sizes and must allow the use of caches.

C. The relation between IdM and Access Control

Services can enforce access control on the basis of the *identity* of an authenticated client, or based on *roles* or *attributes* associated with the client. For the purpose of the accommodation of roaming users, it is absolutely necessary to make access control decisions based on roles/attributes, not identity. Identity based access control requires that all roaming clients are registered into the guest IdM, which is an unscalable solution.

The principles of *Role/Attribute Based Access Control* (RBAC/ABAC) are well investigated [8]. The names and meaning of the roles/attributes that are used to make access decisions must be coordinated as a part of an IdM trust relationship. For that reasons, the number of roles/attributes used for access control needs to be kept low.

It is the obvious responsibility of an IdM to manage the roles/attributes of a subject, some of which may enter into access control decisions, others be used by the service to adapt the user interface etc. The presence of subject attributes is the main functional difference between IdM credentials and X.509 public key certificates.

III. The GISMO IdM architecture

For the purpose of authenticated service provisioning in military tactical networks (meaning wireless, mobile, multi-hop, multicarrier networks), an Identity Management system has been developed under the project name "GISMO" (General Information Security for Mobile Operation). The system has been previously presented in [3], [4], so its properties are only briefly listed here:

- It uses short lived *Identity Statements* containing the subject's public key and subject attributes. No revocation scheme is necessary. Identity Statements are issued by an Identity Provider (IdP).
- Cross COI relations are represented by ordinary identity statement issued from one IdP to another.
- IdPs can issue *Guest Identity Statements* when presented with a Identity Statement issued by an IdP with with whis it has a Cross COI relation. A guest identity statement contains the same information, but is signed by the different IdP.
- Authentication takes place either through a signature in the service request, or through the encryption of the service response.
- Supports Role/Attribute Based Access Control (RBAC/ABAC) through the subject attributes.
- Employs, but encapsulates an existing PKI. Clients never see X.509 certificates or revocation info.
- Identity Statements are cached and re-used during its lifetime. An IdP is invoked to issue Identity Statements, not to verify authenticity.
- Loose coupling between IdP and services/clients, and between COIs. Very little redundant registration is necessary.

Subject Distinguished Name
Subject Public Key
Subject Attributes
Valid from–to
Issuer Distinguished Name
Issuer Public Key
Issuer’s Signature

Figure 2. The structure of the Identity Statement

The main contribution of this manuscript is an IdM for mobile systems and the related discussion on how a common IdM can accommodate different presentation layers.

Figure 1 illustrates the concepts and components of the GISMO IdM. Identity establishment, key generation and key certification happens in the (existing) PKI. Related to a CA (Certificate Authority) domain there are several Communities of Interest (COI) with one IdP common to all members of that community.

The IdP issues signed *Identity Statements*. The structure of the Identity Statement is shown in Figure 2.

Members of a COI only trust the signature of their IdP, so an Identity Statement (signed by the IdP) is not valid outside the COI unless there exists a *cross-COI Identity Statement* which links the signature of the foreign IdP to the trusted IdP. More on that later.

A. Presentation layer issues

The GISMO IdM was first developed over existing SOAP standards like SAML, WSSec, WS-addressing etc. There are libraries for Java that supports the processing of the structures, although somewhat incomplete and buggy. There are also .NET components available, but we were unsuccessful in building the IdM services on .NET. Interoperability were therefore apparently limited to Java code based on the same class library (Sun XWSS 2.0).

The second version of the GISMO IdM was built with a different presentation layer. The choice was to use *serialized Java objects*. Java serialization is a mature and well proven technology, which is available for all Java platforms except J2ME, and is also supported by the Android Dalvik virtual machine.

Even though Java is a proprietary platform as opposed to SAML/WSSec, the interoperability property of the IdM actually has improved, since it now accommodates not only J2SE platforms, but also Android Dalvik VM and programming languages that use the same serialization engine, e.g., Scala. Besides, the serialization engine consumes less resources than the XWSS library and the serialization API is straightforward and well understood.

Consequently, the second phase of the GISMO IdM uses native Java objects (POJO) for representing identity statements and service invocation PDUs. These objects are serialized during network transport. The transport protocol of choice has been HTTP and XMPP, although any reliable transport protocol (or messaging middleware) will do.

To summarize: the reasons for the use of native Java objects for PDU presentation rather than XML based standards are interoperability, network efficiency and ease of programming. The last property is of importance since this is a prototype system for experimental study.

TABLE I
ABBREVIATIONS USED IN THE FIGURES

Client X_a	Client X of COI a
IdP_a	Identity provider of COI a
PKI_a	Validation services in domain a
Server F_b	Server F in COI b
$(Id_x)_a$	Identity statement for identity x, issued by IdP_a
$(msg)S_x$	Message *msg* signed with private key of x
$(msg)E_x$	Message *msg* encrypted with public key of x

IV. Cross COI relationships

Any client will likely be a member of several COIs, reflecting the diverse tasks and responsibilities of a worker or a soldier. It is not convenient to manage the client’s key pairs, attributes etc. in every COI. Most of them will naturally belong to one COI, e.g., their national military unit or the employing department, and could be regarded as “guests” in other COIs.

The ability to authenticate across COI borders is believed to be an essential requirement for a modern IdM. In the GISMO IdM, this problem has been solved by the use of *Guest Identity Statements*. One IdP can issue a Guest Identity Statement if presented for an Identity Statement issued by an IdP with which it has a trust relationship. The trust relationship is represented by a pair of *cross-COI Identity Statements* issued from one IdP to the other.

During invocation of a service in the foreign COI, the client presents the Guest Identity Statement as a part of the authentication process. The service trust the Guest Identity Statement since it is issued by “its” IdP. In order for the client to authenticate the service response, it needs the cross-COI identity statement issued by *its own IdP* to the foreign IdP so that a signature path back to its own IdP can be made. The service signs the response, includes its identity statement (signed by the foreign IdP), which together with the aforementioned cross-COI identity statement forms a signature path back to the trust anchor of the client.

Figure 3 shows the interaction between the client and the IdPs during the issuance of identity statements. Please observe that the cross-COI identity statements are issued asynchronously with regard to the client operations, but handed back to the client during issuance of a guest identity statement. Abbreviations used in the figure are explained in Table I.

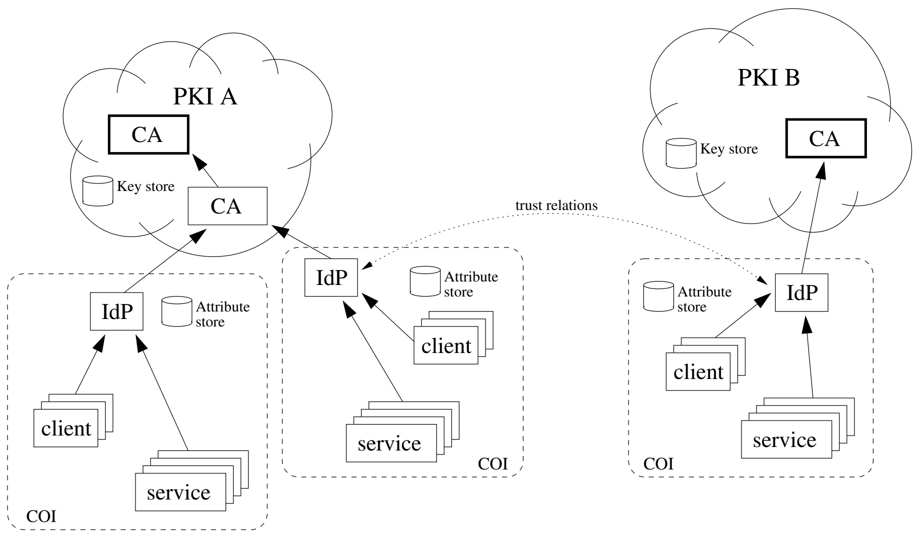


Figure 1. The functional components of a federated IdM. Observe that the IdP serves one single COI, and the trust relations are formed between COIs, not domains. Key management is handled by the PKI whereas the attribute management is done by the IdPs on the COI level

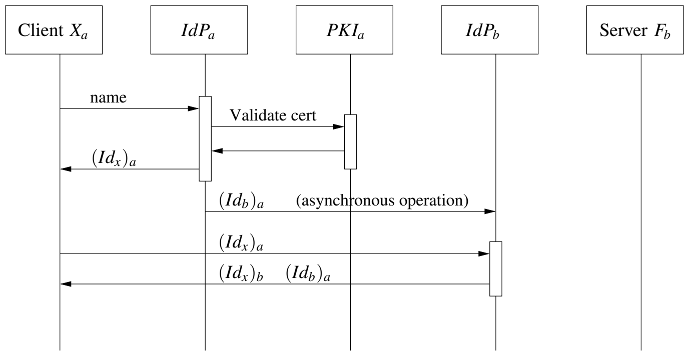


Figure 3. The identity statement issuing protocol. The IdP of COI A, termed IdP_a, issues a "native" identity statement to the client, which is given to IdP_b, which in turn issues a guest identity statement. The term PKI_a denotes a set of certificate validation services in COI a.

V. Service invocation

For service invocation using serialized POJOs as PDUs a number of interesting opportunities knock: The client may simply send a parameter object to the server containing the parameter values, and the *class* of the object identifies the service method. This arrangement eliminates the need for a separate scheme for service addressing and also eliminates the need for separate stub/skeleton compilation.

In the server, a single Java servlet hosts all services. This is possible since we do not address the service in a URL, but through association with the parameter class. The URL addresses a servlet "dispatcher" service, and the serialized parameter object included in the POST operation controls the dispatching process. The services are loaded dynamically from a JAR file repository at servlet startup and deployed through class introspection, no configuration file editing is necessary. Consequently, the deployment of services requires less configuration than a Java servlet.

One could argue that Remote Method Invocation (RMI) could have been chosen rather than a home made invocation scheme. The answer is that RMI is a full size distributed object system, whilst what is needed here is invocation of remote *procedures*. RMI is not very firewall friendly, requires distributed garbage collection and separate stub compilation and is over-specified for this particular purpose.

A. Authentication dependent on server state space

The authentication mechanisms assure the identity of the client and service during service invocation. Many different authentication protocols can be incorporated into GISMO IdM as long as they employ a public key pair corresponding to the information in the Identity Statement. It is also a requirement that the authentication can be piggybacked on the service request and should not generate separate PDUs. Two protocols have been implemented in GISMO IdM:

1) In those cases where the request must be authenticated *before* the service execution a replay protection must be in place. Replay protection requires the server to remember past requests (by their Nonce) for a while, so a clock synchronization scheme and a non-volatile stable storage must be in place (since past requests must be be remembered also across server incarnations). These requirements are rather costly.
2) In the case of a *stateless* service, where the execution of a service request does not alter the state of the service, replay protection is not necessary. A request should be signed by the client in order to protect the integrity of the message, but no Nonce for request replay protection is included. The response is *en-*

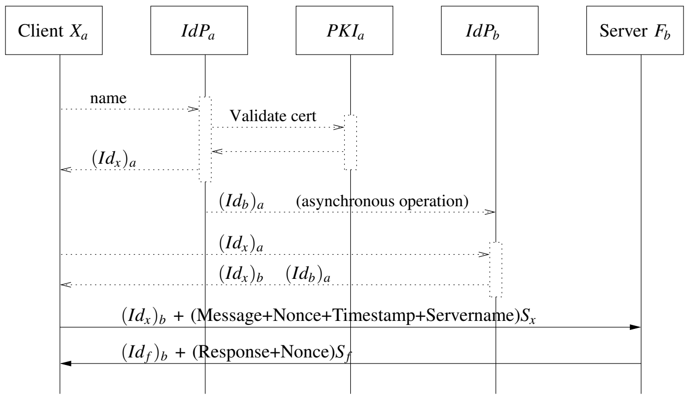

Client X_a
IdP_a
PKI_a
IdP_b
Server F_b
name
Validate cert
$(Id_x)_a$
$(Id_b)_a$ (asynchronous operation)
$(Id_x)_a$
$(Id_x)_b$ $(Id_b)_a$
$(Id_x)_b$ + (Message+Nonce+Timestamp+Servername)S_x
$(Id_f)_b$ + (Response+Nonce)S_f

Figure 4. The authentication protocol for the stateful service. Both the request and response are signed with the sender's private key as a part of authentication process. A timestamp, a nonce and the server's name is included for replay protection.

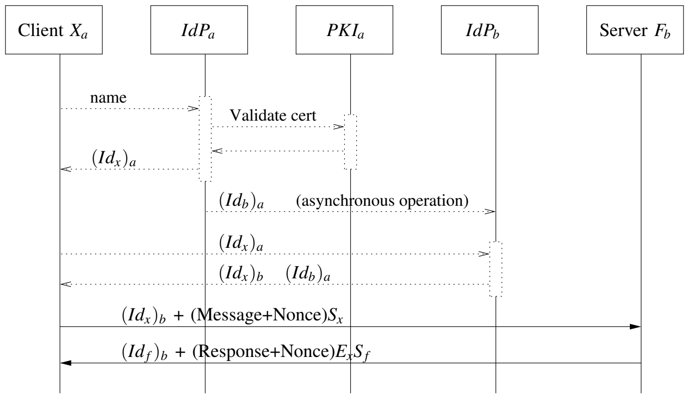

Client X_a
IdP_a
PKI_a
IdP_b
Server F_b
name
Validate cert
$(Id_x)_a$
$(Id_b)_a$ (asynchronous operation)
$(Id_x)_a$
$(Id_x)_b$ $(Id_b)_a$
$(Id_x)_b$ + (Message+Nonce)S_x
$(Id_f)_b$ + (Response+Nonce)E_xS_f

Figure 5. The authentication protocol for the stateless service. Requests are not reply protected since this is not considered as a threat, but the response need to be protected for reasons of response replay and information compromise. For the sake of integrity protection, the request is signed. The encryption of the response is a part of the authentication scheme, not a privacy measure.

crypted with the client's public key, making it useless for everyone but the holder of the private key. To a stateless server, replayed requests are not a threat and protection is not needed. Requests still need a Nonce for reasons of response replay protection, but that does not increase the state space in the client.

Figures 4 and 5 shows the two variants as an interaction diagram. The interactions shown with dotted lines are related to IdP operations and discussed in more detail in Figure 3.

B. Authentication during Identity Statement Issuance

Authentication also takes place during Identity Statement issue operations. The client simply signs the request with its private key. If the requested Identity Statement contains the corresponding public key the client is regarded as authenticated.

The Identity Statement is generally a public document and the need for authenticated requests does not always seem apparent. It is, however, likely that some subject attributes are sensitive since they reveal information about the subject's authorizations. For that reason, only authenticated requests are given the full attribute set in the Identity Statement, others receive a subset of the attributes. The selection takes place over a simple attribute name prefix convention.

VI. MESSAGING PROTOCOLS

In a wired private network where capacity and reliability suffice, and there exist IP routes between the nodes that wish to communicate, the HTTP protocol works just fine for IdP operations and service invocations. For mobile networks this is not necessarily the case: they are slow, unreliable and consists of several partitions connected with application level gateways (from reasons of security and traffic control).

In the context of the experimental study of the GISMO IdM, an XMPP (eXtensible Messaging and Presence Protocol) network was already in place for chat communication. Through the XMPP routers (working as application gateways) otherwise isolated networks (where no IP route exists between them) can exchange chat messages. The XMPP system provides reliable and "persistent" communication in the sense that messages are stored in XMPP routers if they are undeliverable for the moment.

A. Service provision by mobile units

A messaging system creates reachable endpoints for nodes, which are disconnected at the IP layer. Nodes which reside behind a NAT unit or a firewall are unreachable from the outside world at the IP layer, yet a messaging system can send them messages. Through the XMPP protocol a mobile node can receive service requests as any other service provider. The prototype system uses a very simple service container (not a servlet), which is easily portable to a mobile Android based unit.

B. Synchronization and message persistence

The use of an "persistent" communication layer underlying an RPC system poses interesting problems related to recovery and resynchronization. In those cases where a client does not receive a timely response it simply aborts the operation and sends a similar request later. The server may have processed the first request and the response may simply be delayed. The client may now receive the delayed response as the apparent response to the repeated operation, which will be discarded. A "forward synchronization" scheme solved that problem. Under some circumstances, the XMPP nodes can be instructed not to store messages if they are of a "headline" type.

VII. SOAP VS. POJO INTEROPERABILITY

The GISMO IdM contains nodes which use different presentations for Identity Statements and service invocations. In order for two nodes to communicate, they must use the same communication stack, including the presentation layer. A client using serialized POJOs can therefore not communicate with an IdP or a service requiring SOAP message syntax and vice versa. For a client to reach the services it needs regardless its choice of presentation syntax three approaches can be taken:

1) Make services (and the IdP) dual-stack.
2) Make a general proxy for automatic conversion between the presentation forms (POJO and SOAP), e.g., based on JAXB.
3) Make a specific proxy for each service

Option 1 is a possible solution, but do carry a rather high cost in terms of software footprint and deployment configuration. Since SOAP services cannot employ the parameter class association scheme explained in Section V, the automatic deployment mechanism must be replaced with a manual configuration procedure.

Option 2 has not been studied in detail, but requires a combination of WSDL-compilation and JAXB-assisted conversion. It is not likely to be possible to convert on-the-fly any POJO to a SOAP message which conform to the WSDL-file of a particular web service.

Option 3 has been studied and tested, and represents an attractive approach. A service which takes the parameter values and passes them to a precompiled web services stub (generated by the WSDL compiler). The return value from the stub is passed back to the caller of the POJO service. Example code lines required for this function are shown below:

```
public class MainClass {
  public Serializable service(WeatherRequest wr,
                              Properties props) {
    try {
      Weather w = new Weather();
      String result = w.getWeatherSoap()
         .getWeather(wr.town);
      return result;
    } catch (Exception e) { return e; }
  }
}
```

Option 3 is also attractive since it gives the developer control over service aggregation and orchestration. One service call to a POJO service need not be passed on as one single web service invocation. Many individual calls may be made, and they may be sequenced or tested in any manner. Aggregated operations are useful because they potentially reduce the network traffic to and from the mobile unit, which is likely to be connected through a disadvantaged link. The proxy can even cache results for subsequent service calls.

For options 2 and 3 there is a problem related to signature values. Equivalent POJO and SOAP messages will have different signature values, and the integrity of the message is broken during a conversion. The proxy can sign the converted object using its own private key, which would require that the service accepts that the proxy vouches for the original client in the authentication phase.

VIII. Conclusions

A number of problems related to identity management of mobile units have been presented and discussed in this paper. Rather than the deployment of a separate IdM with a presentation layer and protocols adapted to a mobile environment, the addition of a separate presentation layer to an existing IdM has been proposed. The architecture of this IdM, including technical details of the use of serialized Java objects (POJO) has been described. Also, a simpler authentication protocol with fewer round trips and smaller PDUs have been proposed.

The use of serialized POJOs in the service invocation opens up interesting opportunities for easier construction and deployment of services. These aspects have been studied and described in the paper.

Future research in this field is planned to be targeted on concept demonstration in military exercises. The frameworks and suggested programming patterns will be tested in a medium scale mobile networks where military technology from several NATO countries will be tested with cooperation and interoperability in mind. During these experiment the protocols' ability to sustain service in disadvantaged networks with low bandwidth and episodic connectivity will be tested under realistic conditions.

References

[1] N. Ragouzis, J. Hughes, R. Philpott, E. Maler, P. Madsen, and T. Scavo, *Security Assertion Markup Language (SAML) V2.0 Technical Overview*, OASIS Committee Draft, March 2008.

[2] K. Lawrence and C. Kaler, *Web Services Security: SOAP Message Security 1.1*, OASIS Standard Specification, 2004.

[3] A. Fongen, "Identity management without revocation," in *SECURWARE 2010.* Mestre, Italy: IARIA, July 2010.

[4] ——, "Architecture patterns for a ubiquitous identity management system," in *ICONS 2011.* Saint Maartens: IARIA, Jan. 2011.

[5] "Shibboleth." [Online]. Available: http://shibboleth.internet2.edu/ [retrieved November 9, 2010]

[6] "OpenID." [Online]. Available: http://openid.net/ [retrieved November 9, 2010]

[7] "The Libery Alliance." [Online]. Available: http://www.projectliberty.org/ [retrieved November 9, 2010]

[8] R. Sandhu, D. Ferraiolo, and R. Kuhn, "The NIST model for role-based access control: towards a unified standard," in *RBAC '00: Proceedings of the fifth ACM workshop on Role-based access control.* New York, NY, USA: ACM, 2000, pp. 47–63.

A Scalable Architecture for Countering Network-Centric Insider Threats

Faisal M. Sibai
Volgenau School of Engineering
George Mason University
Fairfax, VA 22030, USA
Email: fsibai@gmu.edu

Daniel A. Menascé
Dept. of Computer Science, MS 4A5,
George Mason University
Fairfax, VA 22030, USA
Email: menasce@gmu.edu

***Abstract*—Dealing with the insider threat in networked environments poses many challenges. Privileged users have great power over the systems they own in organizations. To mitigate the potential threat posed by insiders, we introduced in previous work a preliminary architecture for the Autonomic Violation Prevention System (AVPS), which is designed to self-protect applications from disgruntled privileged users via the network. This paper extends the architecture of the AVPS so that it can provide scalable protection in production environments. We conducted a series of experiments to asses the performance of the AVPS system on three different application environments: FTP, database, and Web servers. Our experimental results indicate that the AVPS introduces a very low overhead despite the fact that it is deployed in-line. We also developed an analytic queuing model to analyze the scalability of the AVPS framework as a function of the workload intensity.**

Keywords-insider threat; scalability; network security.

I. Introduction

Defeating the insider threat is a very challenging problem in general. An insider is a trusted person that has escalated privileges typically assigned to system, network, and database administrators; these users usually have full access and can do almost anything to the systems and applications they own. Users with escalated privileges within an organization are trusted to deal with and operate applications under their control. This trust might be misplaced and incorrectly given to such users. It is extremely difficult to control, track or validate administrators and privileged user actions once these users are given full ownership of a system. The recent disclosure by Wikileaks of U.S. classified embassy foreign policy cable records provides a perfect example of an insider attack [1]. In this disclosure, an insider with unfettered access to data at his classification level was able to access data over a secure network using laptops that had functional DVD writers. Our approach to mitigate the insider threat allows for users or groups of users to be treated differently despite having the same classification level [2]. The approach limits and controls network access through an in-line component that checks access to specific applications based on policies that can be as specific or granular as needed.

In our prior work, we introduced a framework that self-protects networks in order to mitigate the insider threat [2]. The framework, called AVPS (Autonomic Violation Prevention System), controls and limits the capabilities provided to administrators and privileged users in organizations. AVPS concentrates entirely on detecting and preventing usage policy violations instead of dealing with viruses, malware, exploits, and well-known intrusions. In our implementation, the AVPS monitors events and takes actions for conditions that occur, as specified by Event-Condition-Action (ECA) commonly used in security-centric systems and autonomic computing [3]. Our prior work does not address scalability though.

The design of the AVPS architecture must consider scalability, manageability, application integration, ease of use, and the enforcement of separation of duties. There has been prior work in this area at the application, host, and network levels [4], [5], [6], [7], [8]. The previous methods have applied self-protecting capabilities by either considering single applications on the host or more towards vulnerabilities, malware, exploits and traditional threats.

This paper significantly extends our previous work ([2]) in that it presents a scalable AVPS architecture and supports its design with experimental results and theoretical queuing modeling. We present here the results of experimental evaluations of the AVPS architectures as well as the analysis of its performance overhead on three different types of application servers: FTP server, database server, and web server. We specifically measured the average throughput, average transfer time, average CPU utilization, and provided 95% confidence intervals for all three measurements. We also used a queuing theoretic analytic model to predict the scalability of AVPS for different workload intensity values for these three types of applications

The rest of the paper is organized as follows. Section II presents some of the major challenges and requirements faced in the design of AVPS. The next section presents a scalable architecture for the AVPS framework. Section V presents an experimental evaluation and a full performance and scalability analysis of AVPS. Finally, Section VI presents the conclusion, final remarks, and future work.

II. Challenges and Requirements

The following major challenges play a primary role in the success of the AVPS framework: scalability in production environments, support for encrypted network traffic, integration with multiple types of application servers on the network, and ease of deployment in large production environments. This paper mainly addresses scalability and performance issues and sheds some light on all four challenges.

Scalability is an absolute requirement for production environments. The AVPS solution is an in-line solution that intercepts every single packet that traverses the local area network that is destined to an application server. Therefore, it could become a focal point and a possible bottleneck. The primary goal of our solution is to scale with growing network and application demands. The AVPS architecture should allow for horizontal scaling to cope with high-volume environments. This requirement is further discussed in more detail in the following sections.

Encryption is another important challenge in the design of our solution. SSH and SSL are widely used in local area networks for information retrieval and administration of applications and devices. The AVPS performs packet inspection on some or all (depending on the application) packets that pass through it. This poses a challenge that is handled in our solution through one of the following methods: (1) decrypting the traffic that passes through the AVPS and then re-encrypting it for delivery to its destination using viewSSLd [9] or netintercept [10] for example, (2) completely off-loading the encryption/decryption requirements to external hardware-based devices that sit before and after the AVPS, or (3) decrypt the traffic by having a legitimate man-in-the-middle host that decrypts and re-encrypts the traffic and delivers it to the destination [11]. This paper does not discuss encryption in any further detail.

Application server integration is also extremely important. With the wide range of applications deployed in production environments, the AVPS framework must be capable of interpreting and understanding requests and responses that it intercepts. The AVPS is based on intercepting, not necessarily inspecting, every single packet initiated by a host that is delivered from and to an application. This makes application integration completely possible and achievable. Policies deployed on the AVPS are customizable to the desired granularity level and types of attributes (e.g., from very generic, such as IP or user level, to very specific, such as IP, user, application type, request, and response). Thus, it is completely up to the AVPS owner to specify the granularity of what should be inspected and what should be ignored.

Finally, the successful deployment of AVPS in large environments is crucial. The AVPS solution should be easy to deploy and maintain and should be capable of handling heavy traffic loads. Current environments have hundreds if not thousands of servers with networks that are capable of handling and processing 100 to 1000 Mbps of traffic. A solution that handles thousands of servers through a handful of clustered AVPS compute nodes is part of the architecture discussed in the remaining sections of this paper.

III. Scalable AVPS Architecture

For the AVPS to achieve its goal of solving the insider threat problem, it must be placed in-line between clients and internal application servers. This way, the AVPS is capable of intercepting every single packet that flows from clients to applications and back in order to take the correct actions when a rule in a policy is matched.

Figure 1 depicts the architecture of the AVPS framework. Performance and high availability are extremely important since the AVPS is located between the clients and the application servers. Traffic coming from a pool of M clients goes through a load balancer that handles incoming requests. The load balancer forwards the traffic to one of N AVPS engines that process and inspect the incoming traffic. The AVPS engines compare traffic policies that contain rules and actions on how to handle traffic. The policies are stored on a database local to the AVPS engine or on an external database shared by all AVPS engines. Events are stored on a centralized database. Actions are taken on traffic once a rule in a policy has been matched. Examples of possible AVPS actions include dropping, blocking, or replacing traffic as it traverses the engine on its way to application servers. Let there be K different types of applications servers (e.g., FTP server, database server, Web server).

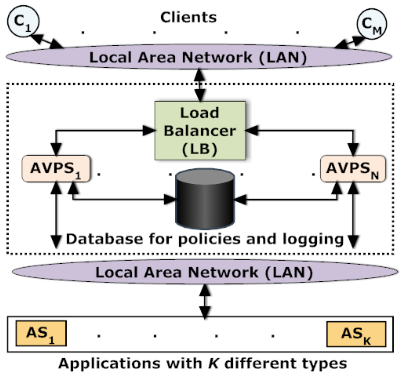


Fig. 1: Architecture of the AVPS framework.

Figure 2 depicts the steps taken by the AVPS engine. Traffic is first received by a layer 2 bridge that is responsible for handling incoming and outgoing traffic. Traffic is then forwarded to the normalization and processing module where packets are broken down into pieces that can be matched against rules. Traffic is then matched against policies and rules that are pre-loaded into memory. If there is a rule match, an event or action is generated. Finally, if an event or action occurred, it is logged into a database.

As an example of the advantage of using the AVPS architecture, consider a scenario with multiple database servers scattered over a large geographically distributed network. Assume that a *top secret* table is replicated in every database server and that we want to have fine access control to this table. Using conventional access control methods, we would be able to limit specific users or roles from accessing the table. This would

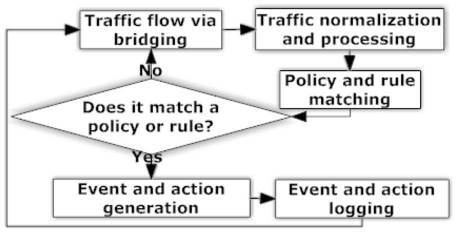

Traffic flow via bridging
Traffic normalization and processing
Policy and rule matching
Does it match a policy or rule?
No
Yes
Event and action generation
Event and action logging

Fig. 2: Steps of the AVPS engine.

require manually setting these controls on every database server. This approach has several drawbacks: (1) Manually setting access controls into each server is time consuming and might have a high error rate. (2) This method requires an administrator to know all of the DB servers that live on the network; newly installed DB servers or even covert ones may be missed. (3) The DB owner actually does the changes with no oversight, which contradicts the separation-of-duties concepts. (4) Last but not least, it would be almost impossible with traditional access control methods to limit access for a specific population of administrators or privileged users, coming from a specific location on the network, accessing the information at a specific time and targeting a specific table.

The AVPS is designed to block detected violations that match specific rules in a policy. Therefore, the AVPS reduces or possibly completely eliminates the drawbacks listed above. The AVPS is also tamper resistant. It enforces a separation-of-duties policy, i.e., the primary application system owner has no control over the AVPS policies [2]. The AVPS can be deployed to carry insider and regular user traffic or to only carry insider traffic. The proper deployment depends on how the network is setup and on how the network is segmented.

Emerging technologies, such as new network TAPs (e.g., Network Critical V-line TAP [12]), that can handle 1/10 Gpbs traffic and allow in-line functionality without introducing a single point of failure, make systems such as AVPS possible to implement without fault-tolerance concerns.

IV. AVPS vs. Other Solutions

Our prior work [2] distinguishes the AVPS from other systems such as IPS, Firewalls, Host based IPS and Network Admission Control/Network Access Control (NAC). We use Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS) in this paper interchangeability. The only difference between the two is that IDS is considered a passive network monitoring system and IPS is considered an active\inline network monitoring system. Traditional IDS/IPS systems tend to concentrate on users that do not have access to the system and try to exploit, hack, or crack into it. Other enhanced IPS/Firewall systems such as IBM Proventia [13] or Cisco ASA [14] do have enhanced context-aware security but lack insider threat defeating capabilities. The AVPS, on the other hand, is designed with the insider threat in mind. Our current AVPS implementation relies on well-known methods used in traditional IDS/IPS for the detection of insider attacks. In our in-progress work we are working on enhancing the detection capabilities of the engine to incorporate self-learning/self-adaptation rule learning, enhance application integration and interaction, include user roles and responsibilities and have better session management and detection capabilities which current IDS/IPS systems either lack or have weak functionality.

V. Performance Assessment of AVPS

This section presents an experimental evaluation of the AVPS in a controlled environment. We describe the experimental testbed, analyze the results, and present a scalability analytical model based on the M/M/N//M queuing model.

A. Experimental Environment

We based our experiments on three different applications: FTP, database, and Web server. The specification of the environment and the experimental testbed is shown in Fig. 3.

In this environment, the client requests services from application servers, which respond to the requests. All traffic between client and server is monitored and inspected by the AVPS. A controlling host controls the environment and collects the results of the experiments (see Fig. 3).

Apache JMeter 2.4 [15] was used on the client to conduct both FTP and Web experiments. We measured the average throughput and average transfer time in both cases. For the database experiment, mysqlslap [16] was used to measure the average response time.

On the AVPS we used Snort-inline 2.8.6.1 [17]. Snort is highly used in academic IDS/IPS research experiments. Other tools are also used in academic research (e.g., Bro [18] and EMERALD [19]). We used Linux iptables [20], a firewall package installed under RedHat, Fedora, and Ubuntu Linux, in conjunction with Snort in-line to filter packets as they come into the AVPS and leave. We used MySQL 5.1 [21] to store events and event packet captures. We used BASE [22] to query the DB and display the events in the browser.

We configured three different application servers: (1) vsftpd 2.3.2 FTP server [23], (2) MySQL 5.1 DB [21], and (3) Apache 2 Web server [24].

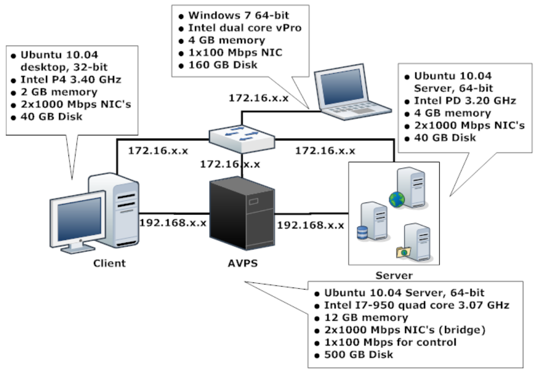


Fig. 3: Experimental environment.

We customized the Snort configuration file to meet the AVPS requirements. All default rules that come with Snort were disabled and our own policies were added inside *local.rules*. We configured Snort to output events into a MySQL database.

The client and server are connected directly to the AVPS as shown in Fig. 3. All three machines are also connected via a second network card to a switch. The controlling host is also connected to the switch to control and collect the results from all three machines.

B. Experimental Results

In this and the following section we show the very little overhead that the AVPS adds in the worst case when traffic passes and is processed by it and alerts are generated. An ideal situation would block the violating traffic without further processing. This causes very little CPU overhead. In this section, we provide measurements for average transfer times and throughputs with 95% confidence intervals, and average and maximum CPU utilization values.

For each application server type, we conducted two types of experiments. The first consisted of manually submitting 10 requests to the application server. This was used to measure the average transfer time, query response time, and throughput. The second consisted of automatically submitting 30 requests to the application server, in sequence with no think time. This process was used to measure the average and maximum CPU utilization of the AVPS engine.

The manual experiments considered the following four scenarios: (1) No AVPS, client and application servers are connected to a 1000-Mbps switch. (2) Client and server are connected to the AVPS but the engine is disabled, traffic is only being bridged. (3) The AVPS is enabled and no rules match the traffic (either because no policies are loaded or because the loaded policies do not trigger a violation). (4) The AVPS is enabled, detects a violation on all rules checked, and generates an alert, which is stored in a database. However, the AVPS is configured not to block the traffic. It should be noted that case (4) above is the one that generates the largest possible overhead because all rules generate a violation, an unlikely event in practice, and traffic flowing through the AVPS is not decreased due to matching offending requests. Thus, all results presented in what follows for scenario (4) represent a worst-case performance scenario.

The automated experiments were used to measure average and maximum CPU utilization of the AVPS engine and consider the following four scenarios: (1) Same as (2) above. (2) Same as (3) above. (3) Same as (4) above. (4) Same as (4) above but the AVPS is configured to block the traffic. Case (3) above is also a worst-case performance scenario for the reasons outlined above. Case (4), the blocking case, is the ideal operational situation. In that case, blocked traffic does not contribute to network and application server load.

The FTP results are discussed in what follows. Table I shows, the measured results for the average throughput (in KB/sec) and average transfer time (in msec) for 10 manually submitted requests using JMeter for four different file sizes: 100 KB, 1 MB, 10 MB and 100 MB. Results include 95% confidence intervals for all file sizes.

File Size →	**100 KB**	**1 MB**	**10 MB**	**100 MB**
Average throughput (KB/Sec) with 95% confidence interval				
No AVPS, switching	327.0 ± 7.01	2811.9 ± 48.00	9834.2 ± 38.48	13395.3 ± 90.12
AVPS, process not on	331.1 ± 5.30	2754.7 ± 35.57	9984.4 ± 56.32	13539.5 ± 94.95
AVPS, process on but not matching	330.0 ± 5.92	2754.9 ± 45.45	9756.8 ± 29.41	13257.5 ± 57.54
AVPS, matching and policy applied	332.6 ± 4.71	2746.4 ± 34.38	9841.2 ± 77.32	13300.8 ± 78.88
Average transfer time (msec) with 95% confidence interval				
No AVPS, switching	307.2 ± 6.87	365.3 ± 7.16	1043.2 ± 4.17	7647.6 ± 51.59
AVPS, process not on	302.8 ± 5.05	372.3 ± 4.81	1025.9 ± 5.97	7566.4 ± 52.39
AVPS, process on but not matching	304 ± 5.77	372.7 ± 6.5	1049.6 ± 3.16	7725.2 ± 33.3
AVPS, matching and policy applied	301.3 ± 4.44	373.4 ± 4.74	1041.1 ± 8.10	7701.2 ± 44.95
Average CPU utilization (%) with 95% confidence interval				
AVPS - bridging only	0.02 ± 0.01	0.04 ± 0.03	0.05 ± 0.01	0.05 ± 0
AVPS enabled, not matching	0.02 ± 0.01	0.3 ± 0.06	1.44 ± 0.20	2.11 ± 0.06
AVPS enabled, matching, not blocking	0.20 ± 0.06	0.79 ± 0.17	3.90 ± 0.51	6.12 ± 0.17
AVPS enabled, matching, blocking	0.09 ± 0.02	0.12 ± 0.02	0.10 ± 0.02	0.12 ± 0.03

TABLE I: FTP results

In the case where we check against a rule (case (4) in the manual experiments), we loaded into memory the following rule that alerts when user "appserver" tries to log into a specific FTP server.

alert tcp any any → FTPserver any (classtype:attempted-user; msg:"Snortinline Autonomic FTP event";content: " appserver";nocase;sid:2;)

From Table I, we see that the differences, respectively, in average throughput and average transfer time for any of the various file sizes are either statistically insignificant at the 95% confidence level (e.g., for 100 KB and 1 MB files) or are very small (e.g., less than 1.8% different for 10 MB and 100 MB files). This means that there is little or no difference between the case when the AVPS process is disabled (case (2)) and the case where the AVPS engine is enabled and all rules checked generate a violation, but traffic is not blocked (case (4)). This is expected behavior since the AVPS does not inspect packets that contain file data being transferred. It only inspects the initial administration and request commands. Thus, the AVPS has no or very little impact on throughput and transfer time.

For the CPU measurements discussed below, we used the automated submission scenario. We load into memory the following rule that blocks a user when he/she tries to access a specific FTP server using "appserver" by replacing it with "*********".

*alert tcp any any → FTPserver any (classtype:attempted-user; msg:"Snortinline Autonomic FTP block"; content: " appserver"; nocase;replace:"*********";sid:2;)*

Table I shows the measured average CPU utilization of the AVPS engine for 30 automated requests with zero think time using JMeter for four different file sizes: 100 KB, 1 MB, 10 MB and 100 MB. The figure also shows 95% confidence intervals.

Table I shows that the CPU utilization grows linearly with the file size. For large files (e.g., 100 MB) we see an average 6.12% utilization when the AVPS is matching but not blocking. This is considered the worst case but is still considered very small and almost has no effect on the traffic traversing or being processed. If we consider the blocking situation (the default action in an ideal AVPS deployment), we see an average of 0.11% utilization, a negligible overhead. This is expected because in this case, data packets are blocked and are not processed any further.

For the database server experiments we built a database of customers, orders, and order items and developed three different queries. Query Q1 returns the list of all items of all orders submitted by all customers for a total of 51,740 records. Query Q2 returns one record with the number of customers in a geographical region. This query needs to scan 50 customer records. Finally, query Q3 returns the dollar amount of all orders placed by customers in a given geographical region. While this query returns only a number, it needs to do significant work on the database to obtain the result.

Table II shows the measured average response time (in sec) for 10 manually submitted queries using mysqlslap for the three different queries and for the four scenarios described above. The table also shows the 95% confidence intervals for all queries.

For the case in which rules generate a violation alert but no traffic is blocked, we loaded into memory the following rule that alerts when a user tries to access "companyxyz" database located at a specific DB server.

alert tcp any any → DBserver any (classtype:attempted-user; msg:"Snortinline Autonomic DB event";content: " companyxyz";nocase;sid:2;)

We can see from Table II, that the worst case appears in Q1, which returns 51740 records. For Q1 the differences between no AVPS and AVPS matching is almost 5 msec, or 13% additional overhead. We consider the extra time to be small given the large number of records returned. In fact, the overhead is approximately 0.08 μsec per record returned. For queries Q2 and Q3 we can see almost no overhead given that both only return one record. In fact, for Q3, there is no statistically significant difference at the 95% confidence level between the no AVPS and AVPS matching cases. For Q2, the difference in response time is small and equal to 1.2 msec.

It is important to note that the largest component of the response time is the transfer time over the network and not processing time at the DB server. We measured Q1, Q2, and Q3 directly at the server and we found that Q1 takes14 msec to execute, and Q2 and Q3 take virtually zero seconds to execute. The difference in execution time between Q1 and the other two queries lies on the fact Q1 has to output a very large number of records. Thus, the average transfer time for case (4) for query Q1 is 22 msec obtained by subtracting the average response time at the client (i.e., 36 msec) from the server execution time of 14 msec.

Query →	**Q1**	**Q2**	**Q3**
Average response time (msec) with 95% confidence interval			
No AVPS, switching	31.6 ± 0.24	10 ± 0.31	10.6 ± 0.39
AVPS, process not on	32.4 ± 0.24	10.2 ± 0.2	10.8 ± 0.57
AVPS, process on but not matching	36.4 ± 0.24	11 ± 0.31	10.6 ± 0.24
AVPS, matching and policy applied	36.2 ± 0.57	11.2 ± 0.2	11.2 ± 0.37
Average/Maximum CPU utilization (%)			
AVPS - bridging only	0.024/0.15	0.045/0.23	0.007/0.04
AVPS enabled, not matching	0.43/1.51	0.01/0.05	0.058/0.3
AVPS enabled, matching, not blocking	1.57/4.75	0.152/0.43	0.23/0.71
AVPS enabled, matching, blocking	0.220/1.49	0.262/1.14	0.221/1.05

TABLE II: DB results

As before, the CPU utilization experiments use the automated submission process. In the cases where we block against a rule, we load into memory the following rule that blocks a user when he/she tries to access the "companyxyz" database located at a specific database server by replacing it with "**********".

*alert tcp any any → DBserver any (classtype:attempted-user; msg:"Snortinline Autonomic DB block"; content:" companyxyz"; nocase;eplace:"**********";sid:2;)*

Table II shows the measured average and maximum (after the "/") CPU utilization of the AVPS engine for 30 automated requests with zero think time using JMeter for queries Q1, Q2, and Q3. The minimum CPU utilization was zero in all cases.

In Table II, we notice that the average CPU utilization does not fully reflect the actual CPU utilization due to the very low amount of time that it takes to process a request over the network. The maximum CPU utilization provides a better view of the actual utilization encountered. We can see again that the worst case occurs with a maximum CPU utilization of 4.75% for Q1 when the AVPS is matching but not blocking. This overhead is considered very small and almost negligible given the number of records returned. The other queries have a maximum of 1.14% utilization, which is extremely low and can almost be completely ignored. In the case of blocking (last row), we see extremely low overhead for the worst case (Q1) that has a maximum of 1.49% utilization. Again, in an ideal environment a blocking policy would be in place.

The results of the experiments in a Web server environment

File Size $\rightarrow$	**518 KB**
Average throughput (KB/sec) with 95% confidence interval	
No AVPS, switching	43038 $\pm$ 1675.01
AVPS, process not on	33861 $\pm$ 902.16
AVPS, process on but not matching	23385 $\pm$ 372.36
AVPS, matching and policy applied	17938 $\pm$ 676.78
Average transfer time (msec) with 95% confidence interval	
No AVPS, switching	6.1 $\pm$ 0.23
AVPS, process not on	7.7 $\pm$ 0.21
AVPS, process on but not matching	11.1 $\pm$ 0.18
AVPS, matching and policy applied	14.6 $\pm$ 0.47
Average CPU utilization (%) with 95% confidence interval	
AVPS - bridging only	0.03 $\pm$ 0.04
AVPS enabled, not matching	0.24 $\pm$ 0.45
AVPS enabled, matching, not blocking	0.54 $\pm$ 1.04
AVPS enabled, matching, blocking	0.15 $\pm$ 0.11

TABLE III: Web results

are shown in Table III, which presents the average throughput (in KB/sec) and the average transfer time (in msec), with 95% confidence intervals, for 10 manually submitted requests using JMeter for a Web page of 518 KB. In the cases where we check against a rule but do not block, we loaded into memory the following rule that alerts when a user tries to access the page "notallow.html" located at a specific webserver.

alert tcp any any → Webserver any (classtype:attempted-user; msg:"Snortinline Autonomic web event"; content:"notallow.html";nocase;sid:2;)

Table III indicates that the average throughput is reduced by 56% when the AVPS is running, matching, and not blocking as compared with the case of no AVPS. The response time difference in that case (see Table III) increases 2.28 times. However, the increase in time units is only 8.2 msec for a large web page (i.e., 518 KB). This increase in response time is hardly noticeable by a human being. It should be noted that in the Web case, the AVPS has to inspect every single packet of a Web page.

Table III indicates that the CPU utilization results for the web case are equally low as in the previous cases.

C. Scalability Analysis

The experiments reported in previous sections allow us to determine the execution time and overhead of running applications protected by the AVPS system. This section uses a queuing-theoretic model to explore the scalability of our proposed approach. We assume that there are M clients that submit requests that are initially processed by one of N AVPS engines, which then send the requests to an application server (AS) (e.g., FTP server, database server, Web server). Each client pauses for an exponentially distributed time interval, called *think time*, before submitting a new request after a reply to the previous request has been received. The average think time is denoted by Z. See Fig. 4 for a depiction of the model.

We also assume that the average time to process a request, not counting time waiting to use resources at the AVPS and the

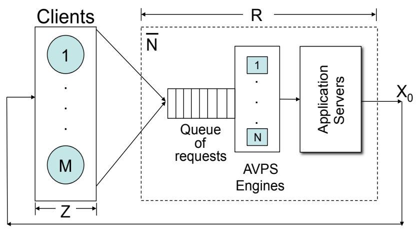


Fig. 4: AVPS analytic model.

application server, is exponentially distributed with an average equal to $\bar{x}$.

We can use the results of the M/M/N//M queue (see [25]) to obtain the probabilities p_k of having k requests being processed or waiting by either the AVPS or the application server. The M/M/N//M queue models a variable service rate finite-population of M request generators that alternate between two states: (1) waiting for a reply to a submitted request and (2) thinking before submitting a new request after receiving a reply to the previous request.

The probabilities p_k are then given by

$$p_k = \begin{cases} p_0 \; (\bar{x}/Z)^k \frac{M!}{(M-k)! \; k!} & 0 \leq k \leq N \\ p_0 \; [\bar{x}/(N \; Z)]^k \frac{M! \; N^N}{(M-k)! \; N!} & N < k \leq M \end{cases} \quad (1)$$

where

$$p_0 = \left[\sum_{k=0}^{N} (\frac{\bar{x}}{Z})^k \frac{M!}{(M-k)!k!} + \sum_{k=N+1}^{M} (\frac{\bar{x}}{N \; Z})^k \frac{M! \; N^N}{(M-k)!N!} \right]^{-1} \quad (2)$$

We can now compute the average number, $\bar{N}$, of requests being processed or waiting to be processed by the AVPS + application server system as

$$\bar{N} = \sum_{k=1}^{M} k \; p_k \quad (3)$$

and the average throughput X_0 as

$$X_0 = \sum_{k=1}^{N} \frac{k}{\bar{x}} \; p_k + \sum_{k=N+1}^{M} \frac{N}{\bar{x}} \; p_k \quad (4)$$

The average response time, R, can be computed using Little's Law [26] as $R = \bar{N}/X_0$.

The workload intensity of such a system is given by the pair (M, Z). An increase in the number of clients M or a decrease in the think time Z imply in an increase in the rate at which new requests are generated from the set of clients. As the processing time $\bar{x}$ increases, contention within the system increases and requests tend to spend more time in the system instead of at the client. In the extreme case, $p_M \approx 1$ and $p_k \approx 0$ for $k = 0, \cdots, M-1$. When that happens, $\bar{N} \rightarrow M$, $X_0 \rightarrow N/\bar{x}$, and $R = \bar{N}/X_0 \rightarrow M \; \bar{x}/N$. In other words, the response time grows linearly with M at very high workload intensities.

We now use the $\bar{x}$ values obtained in our measurements from Section V-B to analyze the scalability of the AVPS for an FTP server, database server and web server under the same conditions shown in the previous sections and for a single AVPS engine (i.e., $N = 1$). Note that the values of $\bar{x}$ used here correspond to the worst-case scenario in the automated tests, i.e., case (3) in which all rules generate a violation and an alert but traffic is not blocked.

Server type		$\bar{x}$
FTP Server	100 KB	0.360 sec
	1 MB	0.513 sec
	10 MB	1.050 sec
	100 MB	8.100 sec
DB Server	Q1	41.6 msec
	Q2	14.8 msec
	Q3	15.6 msec
Web Server	518 KB	12.1 msec

TABLE IV: Average Service Time $\bar{x}$ for the FTP Server, DB Server and Web Server Applications.

Figure 5 shows the average file transfer time R when the number of clients varies from 5 to 30 for an average think time equal to 10 sec. The AVPS is enabled, matching packets against the policy, but not blocking bad transfers. If blocking were enabled the transfer time would be reduced since some files would not be transferred. As expected, for each file size, the average transfer time increases with the file size. For large files (e.g., 100 MB) and for this value of the think time, the system is close to saturation and the average transfer time increases almost linearly with the number of clients, as discussed above. For example, $R = 233$ sec for $M = 30$. This value is very close to $30 \times \bar{x} = 30 \times 8.1 = 243$ sec. For half the number of clients, R is 111.5 sec, which is almost half the value for $M = 30$. But, even in this worst case, the FTP server with the AVPS system scales linearly with the number of clients.

Before saturation is reached, the increase in average transfer time is more than linear, as can be seen for example in the 10 MB file size case. For example, the value of R for $M = 30$ is about 3.4 times higher than for $M = 15$. However, as M increases way past $M = 30$ for 10-MB files, the system will saturate and the transfer time will increase linearly with M.

Figure 6 shows the average response time, R, for the result of queries Q1, Q2, and Q3 defined in Section V-B for an average think time equal to 0.1 sec. As before, the number of clients varies from 5 to 30. The number of records returned by queries Q1-Q3 are 51740, 1, and 1, respectively. Q3 is a much more complex query and requires more database processing time. Thus, its average response time is slightly higher than that for Q2, even though both queries return the same amount of data. The graph indicates that for 30 clients and for Q1, the system is very close to saturation and the average transfer time is very close to be proportional to M. In fact, $R = 1.148$ sec $\approx 30 \times \bar{x} = 30 \times 0.0416 = 1.248$ sec. Queries Q2 and Q3 do not return enough records to push the system to saturation and therefore we see a more than linear increase in transfer time as a function of M for the values shown in the graph.

Figure 7 shows the average transfer time R for a 518-KB Web page and for an average think time equal to 1 sec. As before, the number of clients varies from 5 to 30. The graph indicates that the increase in transfer time is negligible between 5 and 30 clients. While R increases linearly with the number of clients, the rate of increase is mainly due to increased congestion at the Web server and not to AVPS overhead, which is small (8.2 msec) and hardly noticeable by a human being.

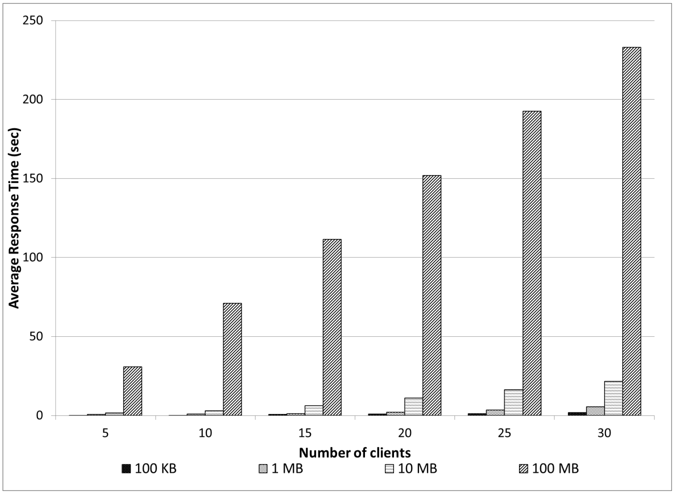


Fig. 5: Average file transfer time vs. number of clients for various file sizes. The average think time is 10 sec.

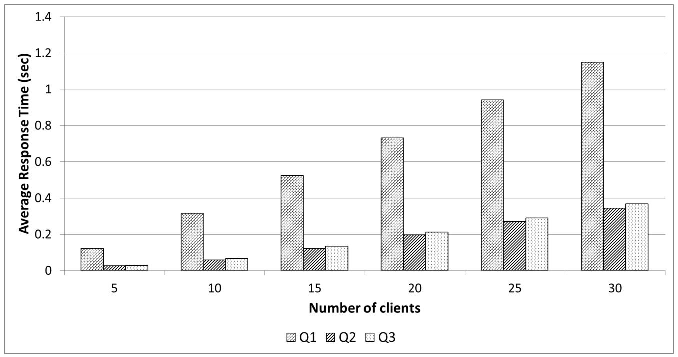


Fig. 6: Average database query result transfer time vs. number of clients for three different queries. The average think time is 0.1 sec.

The clustered architecture for AVPS allows for horizontal scaling. Using the model presented above and the FTP server example, we can see the effect of increasing the number of AVPS engines from 1 to 5. This is illustrated in Table V shows the average file transfer time for 100 MB files, 20 clients, and an average think time of 10 sec. As it can be seen, increasing the number of AVPS engines from 1 to n reduces the average transfer time by a factor larger than n.

VI. Concluding Remarks

This paper presented a scalable AVPS framework to defeat the insider threat. It also presented a performance evaluation assessment for three different application servers. In the

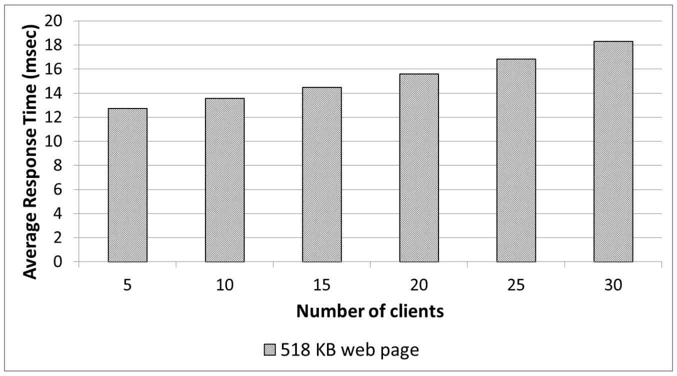


Fig. 7: Average web transfer time vs. number of clients for a 518-KB Web page. The average think time is 1 sec.

N	1	2	3	4	5
R	152	71	44	31	22

TABLE V: Average file transfer time, R, (in sec) for various values of the number of AVPS engines, N. Other parameters: $M = 20$ and $Z = 10$ sec.

performance assessment we measured average transfer times, average throughput, and CPU utilization. We also provided 95% confidence intervals for all three measurements.

The experiments showed that: (1) The impact on the average transfer time and throughput for FTP transfers is either negligible at the 95% confidence level or very small (i.e., less than 1.8%). (2) The response time impact on database queries is heavily dependent on the number of records returned by the queries. For queries that return a very large number of records (e.g., over 51,000), the response time increase is 13% on average. However, this amounts to only 0.08 μsec on average per record returned. (3) When a Web server is accessed through the AVPS system, the response time for a large Web page (e.g., 518 Kbytes) increases by 8.2 msec, an amount hardly noticeable by a human being. (4) The average and maximum CPU utilization of the AVPS engine are very small in all cases tested, not exceeding 7%.

We also presented an M/M/N//M queuing analytical scalability model and generated expected response times for all three application servers. The goal of our scalability and performance evaluation was to show that there is very low overhead incurred when the AVPS is in-line between the clients and the application servers. We used worst-case scenarios in our analysis by considering situations in which all rules checked trigger a violation and generate an alert, but do not block incoming traffic. Blocking traffic in violation situations, which is the normal operational situation, reduces the load on the network and on the AVPS engine and improves performance.

The AVPS is based on Event-Condition-Action (ECA) autonomic policies. If a condition occurs, an event/action is triggered. Rules are entered into policies manually. The use of ECA might be difficult to maintain and manage if the number of rules is very large. For this reason, we are currently looking into self-learning and self-adapting approaches to lower human involvement in rule writing. We are also looking at model based architectures, typically used in self-optimizing systems, and the effects of rule complexity on the overall performance of the system.

References

[1] "zdnet," 2010. [Online]. Available: http://www.zdnet.com/blog/perlow/wikileaks-how-our-government-it-failed-us/14988
[2] F. Sibai and D. Menascé, "Defeating the Insider Threat via Autonomic Network Capabilities," in *Proc. Third Intl. Conf. Communication Systems and Networks, 2011.*, 2011.
[3] M. Huebscher and J. McCann, "A survey of autonomic computing - degrees, models, and applications," *ACM Comp. Surveys, Vol. 40, Issue 3*, pp. 1–28, 2008.
[4] G. Jabbour and D. Menascé, "Policy-Based Enforcement of Database Security Configuration through Autonomic Capabilities," in *Proc. Fourth Intl. Conf. Autonomic and Autonomous Systems*. IEEE Computer Society, 2008, pp. 188–197.
[5] ——, "The Insider Threat Security Architecture: A Framework for an Integrated, Inseparable, and Uninterrupted Self-Protection Mechanism," in *Proc. 2009 Intl. Conf. Computational Science and Engineering-Volume 03*. IEEE Computer Society, 2009, pp. 244–251.
[6] M. Engel and B. Freisleben, "Supporting autonomic computing functionality via dynamic operating system kernel aspects," in *Proc. 4th Intl. Conf. Aspect-oriented Software Development*. ACM, 2005, p. 62.
[7] Y. Al-Nashif, A. Kumar, S. Hariri, G. Qu, Y. Luo, and F. Szidarovsky, "Multi-Level Intrusion Detection System (ML-IDS)," in *Intl. Conf. Autonomic Computing, 2008.* IEEE, 2008, pp. 131–140.
[8] R. He, M. Lacoste, and J. Leneutre, "A Policy Management Framework for Self-Protection of Pervasive Systems," in *2010 Sixth Intl. Conf. Autonomic and Autonomous Systems*. IEEE, 2010, pp. 104–109.
[9] "viewSSLd," 2010. [Online]. Available: http://sourceforge.net/projects/viewssld/
[10] "Netintercept, Sandstorm Enterprises, Inc." 2010. [Online]. Available: http://www.sandstorm.net/products/netintercept/
[11] "Ettercap, Sourceforge," 2010. [Online]. Available: http://ettercap.sourceforge.net/index.php
[12] "Network Critical V-Line TAP, Network Critical Solutions Limited," 2008. [Online]. Available: http://www.networkcritical.com/Products/Bypass.aspx
[13] "IBM Proventia Network Intrusion Prevention System , IBM," 2011. [Online]. Available: http://www-01.ibm.com/software/tivoli/products/network-multifunction-security/
[14] "Cisco ASA, Cisco Systems," 2011. [Online]. Available: http://www.cisco.com/en/US/products/ps6120/index.html
[15] "JMeter," 2010. [Online]. Available: http://jakarta.apache.org/jmeter/
[16] "MySQL Slap, Oracle Corporation," 2010. [Online]. Available: http://dev.mysql.com/doc/refman/5.1/en/mysqlslap.html
[17] "Snort, Sourcefire, Inc ," 2010. [Online]. Available: http://www.snort.org/snort
[18] "Bro Intrusion Detection System, Lawrence Berkeley National Laboratory," 2010. [Online]. Available: http://www.bro-ids.org/
[19] "Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD), SRI Intl." 2010. [Online]. Available: http://www.csl.sri.com/projects/emerald/
[20] "Iptables, netfilter," 2010. [Online]. Available: http://www.netfilter.org/
[21] "MySQL DB, Oracle Corporation," 2010. [Online]. Available: http://www.mysql.com/
[22] "BASE Project, Basic Analysis and Security Engine," 2010. [Online]. Available: http://base.secureideas.net/
[23] "Vsftpd," 2010. [Online]. Available: http://vsftpd.beasts.org/
[24] "Apache 2, The Apache Software Foundation," 2010. [Online]. Available: http://httpd.apache.org/
[25] D. Menascé and V. Almeida, *Capacity Planning for Web Services*. Prentice Hall, 2002.
[26] L. Kleinrock, *Queueing systems, volume 1: theory*. John Wiley & Sons, 1975.

A Framework for Protocol Vulnerability Condition Detection

Yuxin Meng
Computer Science department
City University of Hong Kong
Hong Kong, China
ymeng8@student.cityu.edu.hk

Lam-for Kwok
Computer Science department
City University of Hong Kong
Hong Kong, China
cslfkwok@cityu.edu.hk

***Abstract*—Intrusion detection system (IDS) detects an intrusion by comparing with its attack signatures. The generation of IDS signatures is based on the analysis of attack traffic, which is a result of exploiting vulnerabilities in a network protocol. Thus, the protocol analysis becomes an effective method to find out protocol vulnerabilities with regard to IDS. But the problem of protocol analysis in IDS is that how to detect all protocol vulnerability conditions in protocols. In this paper, we propose a novel framework to identify protocol vulnerability conditions by utilizing existing protocol analysis techniques. In particular, there are three major analysis steps in our framework: protocol semantic analysis, protocol implementation analysis and protocol state transition sub-condition analysis. In the final step of our framework, we illustrate the use of deletion, addition and modification operations with the purpose of generating all potential protocol vulnerability conditions from the normal protocol transition conditions. Experimental results show that this framework is encouraging and feasible.**

Keywords-intrusion detection; vulnerability analysis

I. INTRODUCTION

Rule-based intrusion detection and prevention systems (RIDS/RIPS) [1, 3] are mainly based on attack signatures to detect an attack. The attack signatures are stored in a rule database and updated to the latest version periodically. What is more, the generation of these attack signatures heavily depends on an exploit of vulnerability in a protocol. Take Snort [2] as an example, this lightweight RIDS monitors and analyzes the protocol packets (e.g., UDP, TCP, IP) according to its rules to alert and prevent intrusions. The common rule format of Snort is as blow:

Action-type protocol-type Source-ip Source-port -> Destination-ip Destination-port (content:"|attack signature|"; msg:"attack msg";)

For an ICMP DDoS attack by using tfn2k tool, the attack rule or signature can be produced in terms of the detected characteristic as below:

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"DDOS tfn2k icmp possible communication"; icmp_id:0; itype:0; content:"AAAAAAAAAA"; rev:5;)

The content "AAAAAAAAAA" in this rule is the attack signature (also called *characteristic*) for this exploit.

What is more, the vulnerabilities in a protocol can appear in different forms, e.g., the change of bit values or the change of packet sequence. In common cases, an attacker usually utilizes these forms of protocol vulnerabilities to do harm to the network security.

To identify protocol vulnerabilities, protocol analysis is a prevalent and effective method used in intrusion detection. The advantages of protocol analysis are listed below:

- Strong capability of vulnerability detection: protocol analysis does not only assist IDS to analyze network traffic in terms of protocol specification, but also has the ability to identify vulnerabilities during protocol implementation. For example, the input length and special characters checking and filtering.
- Target detection space reduction: protocol analysis lightens the analysis workload by cutting down the target number of protocol fields, e.g., searching for specific parts of packet rather than entire payload.

Problems. The coverage of signatures is the key problem for rule-based IDS in reducing the detection accuracy. The IDS signatures usually are easy for an attacker to evade by making some small modifications of the original message in a packet. For example, changing the size of variable-length fields or changing the field values in a packet with the purpose of mismatching the IDS signatures.

We argue that the coverage problem of IDS signatures stems primarily from the variants of vulnerabilities in a protocol. In particular, different forms of a vulnerability in a network protocol usually are caused by some minor changes of respective protocol vulnerability conditions. As a result, the ideal solution to this problem is finding out all potential protocol vulnerability conditions that lead the protocol state from a normal to an abnormal state.

Related work. The concept of modifying the software testing paths has been implemented in detecting software vulnerability conditions [17, 6, 8] and then help identifying software threats. Saxena et al. [18] introduced loop-extended symbolic execution that broadens the coverage of symbolic results with loops to find out the vulnerability conditions in programs. Our work attempts to make use of this concept of detecting software vulnerability conditions through creating various software testing paths into the detection of protocol vulnerability conditions by analyzing a protocol specification. We aim to make progress towards systematic detection of possible vulnerability conditions in network protocols by applying three operations to normal protocol state conditions.

Contribution. In this paper, we propose a framework to detect protocol vulnerability conditions by utilizing existing

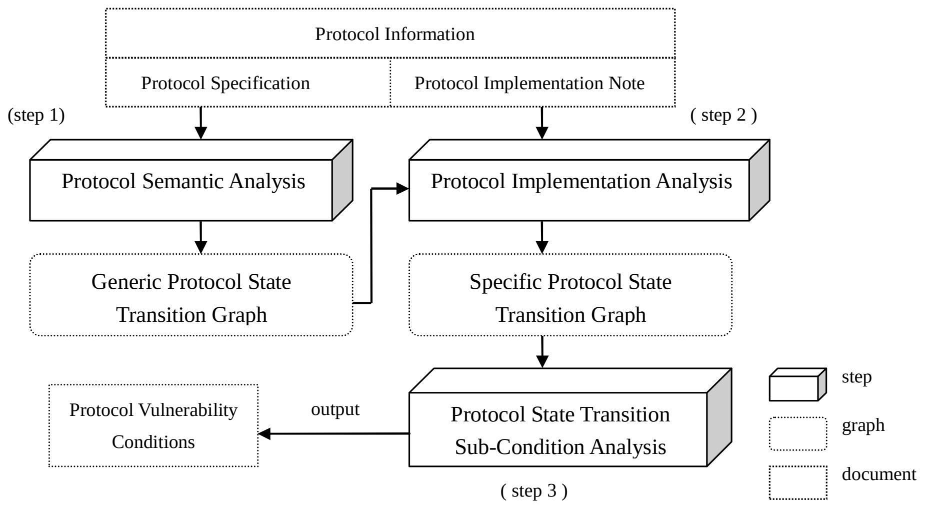

Protocol Information
Protocol Specification
Protocol Implementation Note
(step 1)
(step 2)
Protocol Semantic Analysis
Protocol Implementation Analysis
Generic Protocol State Transition Graph
Specific Protocol State Transition Graph
Protocol Vulnerability Conditions
output
Protocol State Transition Sub-Condition Analysis
(step 3)
step
graph
document

Figure 1. The framework for identification of protocol vulnerability conditions.

protocol analysis techniques with the goal of identifying all vulnerability conditions in network protocols. In particular, our framework has the ability to detect protocol vulnerability conditions without a real attack by applying the operations (deletion, addition and modification) onto normal transition conditions of protocol states.

To demonstrate the feasibility of our framework, we developed and evaluated the framework in an experimental environment which is constructed by Snort [2], Wireshark [8] and a packet generator [9]. Furthermore, we verified our framework by comparing our identified ICMP protocol vulnerability conditions with a set of ICMP Snort rules.

The rest of this paper is organized as follows: in Section 2, we introduce the steps in our framework that how to detect potential protocol vulnerability conditions and then give an in-depth description of operations in protocol state transition sub-condition analysis; Section 3 presents our experimental methodology and an experimental result; Section 4 states the future work; at last, Section 5 gives our conclusion.

II. Our Framework

In this section, we first give the definition of protocol vulnerability condition in our framework, and then introduce the representation format of protocol vulnerability condition.

Protocol vulnerability: a point that causes the execution of a protocol to be out of normal function and make errors.

Protocol vulnerability condition: A specific and certain condition that leads the protocol state to reach the protocol vulnerability which results in causing the protocol execution to an abnormal state.

In addition, the specific forms of protocol vulnerability conditions could consist of particular triggers (e.g., network parameters change, packet flag values reset) that cause a protocol vulnerability exploited, and abnormal points (e.g., implementation errors, coding ignorance etc.) that possibly compromise the function of a protocol.

What is more, we use the *IF/THEN* format to represent these protocol vulnerability conditions as the output in our framework. For example, as for Destination fragmentation vulnerability in ICMP (this vulnerability in ICMP packet due to the packet size is larger than 65536 octets, but the DF/Don’t fragmentation bit is set to 1), the representation of this protocol vulnerability condition could be (according to captured attack traffic):

IF {Datagram greater than 65536 octets and DF=1}
THEN {alert msg: ping of death attack}

This representation is compact and at protocol level. The merits of this representation (*IF/THEN*) are:

- Easy for understanding, the *IF* part describes the details of vulnerability conditions in protocols, the *THEN* part gives the description and information of this exploit.
- In favor of signature generation, it is comfortable for rule-based intrusion detection/prevention systems to produce attack rules and signatures according to the *IF/THEN* representation. In general, attack signature can be extracted from the *IF* part, and alert message is corresponding to the *THEN* part.

In the next two subsections, we first give details of the steps in our framework to account for the general procedure that how to detect protocol vulnerability conditions with high coverage by making use of current protocol analysis techniques. We then give an in-depth description on the use of operations (deletion, addition and modification) that how to identify potential protocol vulnerability conditions from known protocol state transition sub-conditions.

A. Framework Design

In Fig. 1, the framework illustrates that how to identify potential protocol vulnerability conditions by using protocol analysis techniques. The framework consists of three major steps: protocol semantic analysis, protocol implementation analysis and protocol state transition sub-condition analysis. The first step aims to produce a *generic protocol state transition graph* in terms of protocol specification (e.g., RFC [12], or use other intermediate language [10]); the second step considers the protocol implementation note to enrich and extend *generic protocol state transition graph* to a *specific protocol state transition graph*; and the goal of the last step is to produce potential protocol vulnerability conditions with operations (such as deletion, addition and modification) onto normal transition conditions of protocol states according to *specific protocol state transition graph* and then analyze the generated results to identify real protocol vulnerability conditions.

Protocol Information. According to the Fig. 1, this is the data source in our framework. The protocol information contains both essential details of Protocol Specification [4] and Protocol Implementation note [5].

- *Protocol Specification:* All semantic information of network protocols can be found in RFC (Request for Comments) [12], in which a series of documents that collect Internet information, UNIX and Software documents of Internet community. The extraction of protocol specification can be referred to previous work for details [4, 6, 7, 11]. In addition, it is useful and effective to find out lots of public specification in Vulnerability Database (e.g., NVD [13], CVE [14], OSVDB [16]).
- *Protocol Implementation Note*: This note contains the implementation details of network protocols (e.g., according to RFC documents, how to program the protocol). Moreover, we need to notice that the real implementation of a protocol may be changed a bit from the standard document due to the specific network environment and demands. We refer the reader to [5, 15, 17] for details of the extraction of protocol implementation note.

In practice, the network protocol implementations are usually distinct from the RFC documents due to practical environment. In this case, we could retrieve the essential protocol information with the method of protocol reverse engineering [6, 11, 19, 20], which is an effective method to find out the principles of a protocol by analyzing the relative structure, function, operation etc.

Step1: Protocol Semantic Analysis. The purpose of this step is to draw a generic protocol state transition graph by using protocol specification. State transition graph [20] (also called state transition diagram) is a graph that indicates the relationship between two states indicating that an object will take certain actions from the first state to the second state. Obviously, protocol state transition graph (PSTG) is a particular instance of the state transition graph in network protocols.

Generic protocol state transition graph (GPSTG). The term of *generic* means that this graph only contains indispensable protocol states and fewer transition conditions which shows the generic transition relationship among protocol states. This generic protocol state transition graph is easier to be drawn according to protocol specification, since there is no need to identify all specific state transition conditions in practical scenarios.

Step2: Protocol Implementation Analysis. This step aims for generating a specific protocol state transition graph by using protocol implementation note and relevant generic protocol state transition graph. Actually, the generation of the specific protocol state transition graph heavily depends on the information in protocol implementation note from which the specific protocol state transition conditions can be indicated.

Specific protocol state transition graph (SPSTG). The term of *specific* means that this graph contains all protocol states, as well all specific protocol state transition conditions. From another view, the SPSTG is an extended graph that emerges from GPSTG by utilizing much more information provided by the protocol implementation note.

Step3: Protocol State Transition Sub-Condition Analysis. The main purpose of this step is to analyze sub-conditions in the specific protocol state transition graph and generate potential protocol vulnerability conditions.

To facilitate the illustration of this analysis work, we give definitions of atom condition and compound condition.

Atom Condition: a certain kind of condition that cannot be decomposed further in semantic level (e.g., {DF=1}, {Datagram greater than 65535 octets}).

Compound condition: a kind of condition that consists of more than one atom condition (e.g., {Datagram greater than 65536 octets and DF=1}).

In this step, the analysis work falls into three types. The first type is to analyze the specific protocol state transition graph and divide the protocol state transition conditions into sub-conditions until all atom conditions are identified within each protocol state transition.

The second type is to pick up overlap atom conditions since these conditions affects more than one state transition. As a result, these overlap conditions are more likely to be utilized to create protocol vulnerability conditions.

The last type of analysis work is to operate (delete, add and modify) on these atom conditions (not only the overlap atom conditions but also the other atom conditions) to form potential protocol vulnerability conditions. To delete, add or modify an atom condition has the possibility to change the contents of relevant compound condition and cause a flaw in the state transition. In this case, these changed conditions (which may cause a flaw in protocol state transitions) are protocol vulnerability conditions that an attacker can utilize.

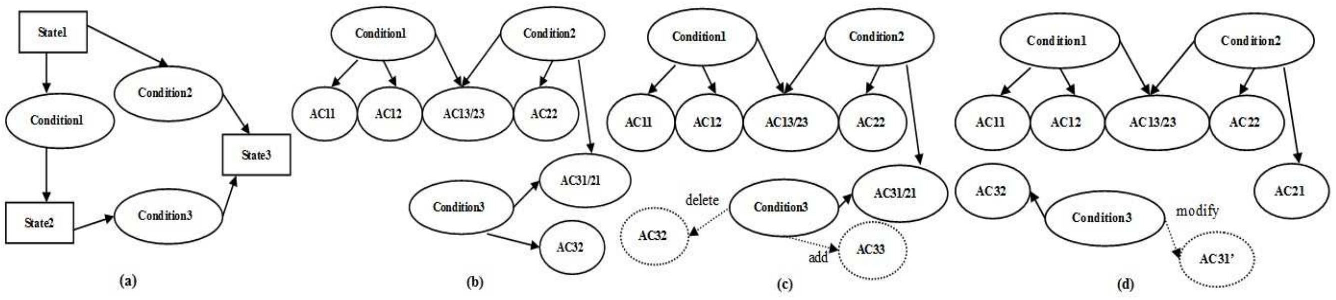


Figure 2. Operations of deletion, addition and modification.

B. *Operations in Protocol State Transition Sub-Condition Analysis*

In Fig. 2, we assume a specific protocol state transition graph and then illustrate the operations (deletion, addition and modification). In Fig. 2 (a), a specific protocol state transition graph is assumed that State1 can change to State2 and State3 if Condition1 and Condition2 are satisfied respectively. State2 converts to State3 as long as Condition3 is fulfilled. In Fig. 2 (b), we assume that Condition1 could be divided into three atom conditions {AC11, AC12, AC13}. Similarly, Condition2 and Condition3 have atom conditions {AC21, AC22, AC23} and {AC31, AC32} respectively.

Look for overlap atom conditions. As shown in Fig. 2 (b), AC13 is the same as AC23, and AC31 is equal to AC21. Thus, AC13/AC23 and AC31/AC21 are the overlap atom conditions. The advantage of finding out these overlap atom conditions is that these overlap atom conditions have more chances to affect the state transitions among State1, State2 and State3. Moreover, the overlap atom conditions are used in the addition operation in our framework. We then define the three operations of deletion, addition and modification.

Deletion of atom conditions. According to a specific protocol state transition graph, deleting or omitting an atom condition in a relevant compound condition could change the original contents of this compound condition and result in a fault or error during the state transition. As shown in Fig. 2 (c), if an atom condition AC32 is deleted, some errors may be occurred in the protocol state transition between State2 and State3.

Addition of atom conditions. Similar to deletion, adding an atom condition to a state transition condition may cause the state to an abnormal state as well. In Fig. 2 (c), adding an atom condition AC33, the State2 could do not know how to deal with the additional condition and thus produces a flaw during the protocol state transitions. In our framework, we use the overlap atom conditions for addition operation.

Empirically, the overlap atom conditions are much more vulnerable in the protocol state transitions. To ensure the feasibility and effectiveness of this operation, we thus utilize the overlap atom conditions to generate potential protocol vulnerability conditions for the operation of addition in our framework. For example, the AC13/23 and AC31/21 are overlap atom conditions according to Fig. 2 (b), we then can attempt to add AC13/23 to Condition3 instead of the AC33 (which is only a generic atom condition) to guarantee that our produced conditions are limited. Like this, we can add AC31/21 to Condition1 in evaluating the effects of these changes as well.

Modification of atom conditions. To change the original content of an atom condition is an effective way to cause some errors in the protocol state transitions. In Fig. 2 (d), if we modify AC31 to AC31' in Condition3, the errors may be caused between the protocol state transition between State2 and State3. The major modification skill is to set the packet bit values to an opposite value or another different value. For example, if DF=0 in original ICMP packet, the DF bit will be set to 1 (DF=1) during the modification.

If deleting, adding or modifying an atom condition can cause a normal protocol state to an abnormal state, this atom condition or the relevant compound condition is regarded to a real protocol vulnerability condition.

The merits of these operations are: 1) the effects and results of applying these operations onto atom conditions could be able to cover all possible forms of protocol vulnerability conditions (which may consist of an atom condition or more than one atom condition). Namely, these operations have the ability to generate all potential protocol vulnerability conditions; 2) it is more likely to identify the variants of known exploits and has the chance to discover unknown vulnerability conditions by applying these three operations and analyzing the produced conditions.

III. EVALUATION

In this section, we evaluated our framework in a constructed environment by using existing tools such as Snort [2], Wireshark [8] and a packet generator [9]. The environment is shown in Fig. 3.

In the following parts, we begin by discussing our experimental methodology to explain that how to launch our evaluation and achieve the results. Then we show the results

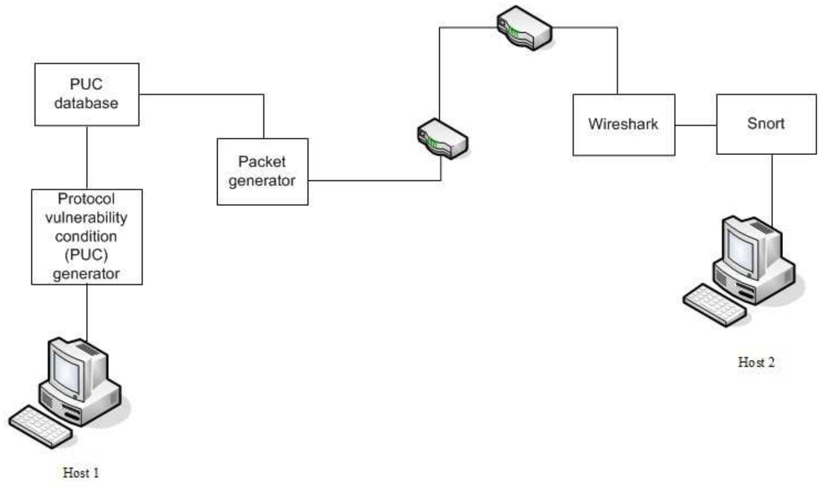


Figure 3. Experimental environment and deployment.

of our experiment on the protocol of ICMP to demonstrate the feasibility of our framework.

A. *Experimental Methodology*

In Fig. 3, we developed a protocol vulnerability condition generator (the input is protocol atom condition, thus we should draw SPSTG in advance) in Host 1 that generates a majority of potential protocol vulnerability conditions with deletion and addition operations. But as for the modification operation, it is more laborious to obtain the results. The PUC database provides a passive storage space for the generated potential protocol vulnerability conditions. Then, we used a packet generator to create packets according to the potential vulnerability conditions in the PUC database (e.g., setting the bit value in the packet to an opposite or different value, changing the sequence of bits). At last, the crafted packets are sent to the Host 2 through routers.

The Host 2 is the target for the experiment. Therefore, we deployed two open source tools Wireshark and Snort into this host with the purpose of monitoring network traffic and detecting abnormal packets that come from Host 1. The Wireshark is a powerful tool to capture network traffic and perform packet analysis while the objective of Snort is to detect abnormal packets according to its rules and signatures.

In this experiment, we used Snort rule database (version 2.8) to verify our identified potential protocol vulnerability conditions. In practice, we evaluate our framework on ICMP by comparing our identified protocol vulnerability conditions with the ICMP Snort rules. Based on these conditions, we can create specific ICMP packets to challenge the Snort. In this case, if the number and contents of our detected potential protocol vulnerability conditions can cover all of the ICMP Snort rules, we can show that our framework is feasible.

B. *Analysis with an Example on the detection of protocol vulnerability conditions*

Based on our experimental methodology, we give an example to illustrate the experiment on ICMP. According to the *description* part of page 5 in RFC 792 [12], we produce normal transition conditions (a compound condition) manually that trigger State1 (send packet) to State2 (wait for response). The conditions have then been divided into three sub-conditions: SubC1 {the distance to the network is finite}, SubC2 {indicated protocol module or process port is active} and SubC3 {datagram need fragmented and DF=0}.

Furthermore, we identify atom conditions as follows: {distance finite}, {protocol module is active}, {process port is active}, {datagram must be fragmented} and {DF=0}. To better understanding, all these atom conditions are presented at semantic level. There are no overlap atom conditions in this example since there are only two protocol states.

Subsequently, we apply operations (deletion, addition and modification) into these atom conditions to affect related transition conditions.

Deletion: delete any one or more above atom conditions. For instance, deleting {distance finite}, the remaining conditions will be {SubC2 and SubC3}. If deleting two atom conditions such as {distance finite} and {process port is active}, the result is {protocol module is active and SubC3}.

Addition: this operation is based on expert knowledge to some degrees. In our method, the atom condition for addition comes from the overlap atom conditions. Since there is no overlap condition in this example, we can skip this operation.

Modification: this action aims to change the contents of atom conditions. For example, atom condition {distance finite}, {DF=0} could be modified to {distance infinite}, {DF=1}. The purpose of these changes is to reverse the meaning of atom conditions or to set condition values to another different value.

We use the *IF/THEN* to represent the generated potential protocol vulnerability conditions. For instance, during the experiment, we can detect a protocol vulnerability condition through modifying the atom condition {DF=0} to {DF=1} in affecting Sub3. Thus, the representation of this protocol vulnerability condition in our framework is:

IF {datagram must be fragmented and DF=1},
THEN {alert msg: host crack down}.

In the experiment, these oversized packets can cause the host to be crashed or rebooted since the host does not know how to deal with these oversized packets. What is more, we find out the relevant rule in Snort rule database to verify this is a real protocol vulnerability condition. The snort rule is:

alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"ICMP Destination Unreachable Fragmentation Needed and DF bit was set"; icode:4; itype:3; reference:cve,2004-0790;reference:cve,2005-0068; classtype:misc-activity; sid:396; rev:7;)

The meaning of this rule is to alert that ICMP message needs fragmentation but the Don't Fragment flag is on, which is the same to our detected protocol vulnerability condition.

Performance Analysis. In the experiment, we evaluated our framework by the use of 115 ICMP Snort rules (in the *icmp.rules* and *icmp-info.rules* folders). In particular, we classified the ICMP Snort rules into 7 types such as ICMP fragmentation, ICMP ping, ICMP redirect, ICMP parameter problem, ICMP unreachable problem, ICMP TTL and ICMP conversion error.

The experimental results on the protocol of ICMP show that our detected ICMP vulnerability conditions cover 100% of the ICMP Snort rules except for those software oriented ICMP rules. We discovered that one protocol vulnerability condition at protocol level could cover more than one Snort rule since the Snort rules are very specific at byte-level. That is, our approach can generate some new exploits based on the variations of the protocol vulnerability conditions at byte level. By analyzing attack patterns of these new exploits, we might be able to discover more new attack signatures, and thus the Snort rules, before the new attacks actually arrive.

IV. FUTURE WORK

While the evaluation demonstrates the analysis steps in our framework, it does reflect some limitations which we could work in the future. First of all, the number of protocol vulnerability conditions increases exponentially by using the three operations. The benefits are high vulnerability coverage and unknown protocol vulnerability condition detection, but it does need much more storage space and is hard to avoid redundant conditions that have the same effects as well. To overcome this issue, we plan to develop a reference engine to correlate these potential protocol vulnerability conditions with the goal of reducing the unreasonable conditions. A second area for future work is the design of a system to generate byte-level IDS rules from the detected protocol vulnerability conditions automatically. In addition, we plan on applying our framework into other network protocols to further evaluate its feasibility.

V. CONCLUSION

In this paper, we proposed a framework aiming to detect all protocol vulnerability conditions in a network protocol. There are three steps in our framework: protocol semantic analysis, protocol implementation analysis and protocol state transition sub-condition analysis. In particular, we describe the relationship among these steps of the framework and define three operations of deletion, addition and modification in the final step that are used to generate potential protocol vulnerability conditions from normal transition conditions of protocol states. In the experiment, we develop and evaluate our framework on the protocol of ICMP in a constructed network environment by using Snort, Wirewark and packet generator. The experimental results show that our framework is feasible and encouraging.

REFERENCES

[1] Paxson, V.: Bro: A System for Detecting Network Intruders in Real-Time. Computer Networks 31(23-24), pp. 2435–2463 (1999).

[2] Snort, http://www.snort.org/. (access on 1 Apr. 2011)

[3] Scarfone, K. and Mell, P.: Guide to Intrusion Detection and Prevention Systems (IDPS), NIST Special Publication 800-94, pp. 1-127, (Feb 2007).

[4] Comparetti, P.M., Wondracek, G., Kruegel, C., and Kirda, E.: Prospex: Protocol Specification Extraction. In: IEEE Symposium on Security and Privacy, pp. 110-125, Oakland, CA (2009).

[5] Yating H., Guoqiang S., and Lee, D.: A model-based approach to security flaw detection of network protocol implementations. In: IEEE International Conference on Network Protocols, pp. 114-123, Orlando, Florida, USA (2008).

[6] Cui, W., Peinado, M., Chen, K., Wang, H.J., and Irun-Briz, L.: Tupni: Automatic reverse engineering of input formats. In: ACM Conference on Computer and Communications Security, pp. 1-12, VA (2008).

[7] Wondracek, G., Comparetti, P.M., Kruegel, C., and Kirda, E.: Automatic network protocol analysis. In: Network and Distributed System Security Symposium, pp. 1-14, San Diego, CA (2008).

[8] Wireshark, http://www.wireshark.org. (access on 21 Mar. 2011)

[9] Colasoft Packet Builder, http://www.colasoft.com/packet_builder/. (access on 21 Mar. 2011)

[10] Borisov, N., Brumley, D., Wang, H.J., Dunagan, J., Joshi, P., and Guo, C.: Generic application-level protocol analyzer and its language. In: Network and Distributed System Security Symposium, pp. 1-15, San Diego, CA (2007).

[11] Lin, Z., Jiang, X., Xu, D., and Zhang, X.: Automatic protocol format reverse engineering through context-aware monitored execution. In: Network and Distributed System Security Symposium, San Diego, CA (2008).

[12] Request for Comments (RFC), http://www.ietf.org/rfc.html. (access on 1 Apr. 2011)

[13] National Vulnerability Database (nvd): http://nvd.nist.gov. (access on 1 Apr. 2011)

[14] Common vulnerabilities and exposures (cve): http://cve.mitre.org. (access on 1 Apr. 2011)

[15] Shu, G. and Lee, D.: Testing Security Properties of protocol implementations: a machine learning based approach. In: Proceedings of IEEE ICDCS, pp. 25-32, Toronto, Canada (2007).

[16] Open Source Vulnerability Database (osvdb), http://osvdb.org/. (access on 1 Apr. 2011)

[17] Peled, D., Vardi, M. Y., and Yannakakis, M.: Black-box checking. In: Proceedings of IFIP FORTE/PSTV, pp. 225-240 (1999).

[18] Saxena, P., Poosankam, P., McCamant, S., and Song, D.: Loop-extended symbolic execution on binary programs. In: International Symposium on Software Testing and Analysis, pp. 225-236, Chicago, IL (2009)

[19] Cui, W., Kannan, J., and Wang, H.: Discoverer: Automatic Protocol Reverse Engineering from Network Traces. In: 16th Usenix Security Symposium, pp. 199-212, Boston, MA (2007).

[20] Guha, B. and Mukherjee, B.: Network security via reverse engineering of TCP code: vulnerability analysis and proposed solutions. IEEE Network, pp. 40-48 (Jul/Aug 1997).

Access Control in BitTorrent P2P Networks Using the Enhanced Closed Swarms Protocol

Vladimir Jovanovikj, Dušan Gabrijelčič, Tomaž Klobučar
Laboratory for Open Systems and Networks
Jožef Stefan Institute
Ljubljana, Slovenia
E-mail: vladimir@e5.ijs.si

Abstract— **The future content delivery platforms are predicted to be efficient, user-centric, low-cost and participatory systems, with social and collaborative connotation. The peer-to-peer (P2P) architectures, especially ones based on BitTorrent protocol, give a solid basis for provision of such future systems. However, current BitTorrent P2P networks lack flexible access control mechanisms. In this paper enhancements to existing access control mechanism for BitTorrent systems – the Closed Swarms protocol are presented, providing additional flexibility in access control mechanism, enabling fine grained security policies specification and enforcement. The enhancements fulfill a number of content providers' requirements and promise efficient, flexible and secure content delivery in future content delivery scenarios.**

Keywords-access control, P2P, BitTorrent, flexible policy, Closed Swarms

I. Introduction

It is envisaged that in the future people will consume 3D content enriched with additional media types and technologies that will engage more of our senses and will provide us immersive experience. People will have the ability to create virtual and personalized environments that will correctly simulate the real world and could have a variety of everyday applications. The virtual environments together with enriched 3D content will bring the communication between people to a higher level, and at the same time will enhance the users' entertainment. Moreover, they will foster human creativity even more and the current trend of people to be not only consumers but also producers of media content is expected to grow [1].

Future content delivery platforms will need to be able to provide efficient delivery of such high quality media content (streaming and stored), on-demand or live to the consumers, with an excellent quality of service. They are predicted to be user-centric and capable of considering the social aspects of the users, as well as the data being delivered. In order to become economically successful, the future content delivery platforms will have to be suitable for large and small size content providers and to be low-cost. This can be achieved if they are participatory and collaborative systems, in which all customers will become actively involved in the content delivery process. Because of its characteristics the peer-to-peer (P2P) architectures gives a solid basis for future provision of such systems. Indeed, one of its most prominent representative [2] – the BitTorrent protocol [3] has already proved to be scalable, robust and efficient in delivery of large audio and video data, and suitable for live streaming and social interaction between its users [4][5][6]. Thus, BitTorrent promises to be a suitable P2P protocol for future P2P-based content delivery platforms.

In short, with BitTorrent peers exchange small and fixed size pieces of the content file. A group of peers sharing the same file is called a swarm. A peer needs to acquire a so called torrent file in order to start downloading. The torrent file contains the needed information for the protocol initiation. The sharing process is coordinated either by a central server – the tracker or by the participants themselves – using the DHT [7] protocol. BitTorrent uses tit-for-tat policies to provide fairness in the delivery process [8]. Peers that continue to upload after they have downloaded the whole content file (seeds) improve the downloading process of the other peers (leeches).

The future P2P-based content delivery platforms need to be secure and trusted in order to be widely accepted and used. The importance of security as well as the main security requirements for P2P networks have already been emphasized in [9][10]. Among them access control is considered basic and standard, especially by content providers. The access control in the P2P-based content delivery systems is quite difficult to accomplish because of the basic properties of the system: 1) the content consumers are directly involved in the process of content distribution, i.e. peers exchange the content among themselves; and 2) the system tends towards full decentralization, without even a single central party for administration.

The main goal of this paper is to propose several enhancements of an existing access control mechanism for BitTorrent P2P networks – the Closed Swarms protocol [11], that we believe will provide a flexible access control mechanism for future P2P-based content delivery platforms applicable in various scenarios. First, we give an overview of the existing approaches for access control in BitTorrent P2P networks in Section II. Then, we describe the motivation for enhancing the Closed Swarms protocol in Section III. We present our proposed enhancements in Section IV and furthermore discuss them in Section V. Finally, we conclude the paper and present our future work in Section VI.

II. RELATED WORK

Access control in P2P content delivery systems can be achieved either directly protecting the content being delivered or controlling the content delivery process.

An access control mechanism directly protecting the content is proposed by Zhang et al. [12]. It is basically a digital rights management (DRM) mechanism for BitTorrent, based on using trusted tracker and initial seed, as well as using trusted content viewer on the client side. The main idea behind their schema is existence of a single plaintext copy of the content being delivered, the one at the trusted initial seed, while all the other copies of the content, resting at the peers being part of the content delivery system are uniquely encrypted for every peer, piece by piece. The peers consume the content only with a trusted content viewer, which is responsible for decrypting the content according to the purchased license from the content provider. This scheme is highly dependable on the tracker, which is far from full decentralization and is an obvious security risk – a single point of failure. Moreover, it doesn't provide means for applying flexible content usage policies, even though it is possible to define copyright related usage policies into the license. All this makes this scheme not appropriate for the future P2P-based content delivery platforms. In addition, the encryption and increased communication with the tracker certainly have impact on the performance of the content delivery. It is worth mentioning that providing copyright protection in a fully decentralized environment that favours open source software is a task very difficult to fulfill.

Another mechanism for direct protection of the content is described by Jimenez et al. [13]. In their scheme, the provider first encrypts the content before it is being distributed among the peers. Only peers that commit a payment and satisfy the provider's policies are authorized to receive the decryption key and consequently are able to decrypt the content. Although this mechanism is capable for implementing a certain access control policies (for example based on geolocation), it depends only on one cryptographic key, which makes it to be easily compromised.

Private tracker [14] extension of the BitTorrent protocol is an access control mechanism for controlling the delivery process. It restricts access in the system by simply not giving information about the participants to unauthorized users, i.e. users that do not meet a certain criteria, such as minimum upload-to-download ratio. This mechanism is not appropriate for future P2P-based content delivery platforms as it is highly centralized. Also, it depends on peers using only one private tracker at a time as a peer discovery mechanism, which makes it be easily subverted.

Closed swarms (CS) protocol [11] is an access control mechanism for controlling the delivery process that acts on peer level. It enables peers to recognize the authorized peers and to avoid communication with the non-authorized ones. The distinction between authorized and non-authorized peers in the swarm is made based on possession of an authorization credential called proof-of-access (PoA). The peers exchange their credentials right after they establish connection, in a challenge-response messages exchange. In most severe case, with the CS protocol only the authorized peers receive service (content). Nevertheless, it is possible to design a system in which both users would receive service (content), but graded – the authorized users would receive additional or better service than the non-authorized ones, for example access to high speed seeds for better performance. The CS protocol can provide access control in an innovative business content delivery system, but only under the same conditions for all authorized users. Moreover, this protocol is vulnerable to man-in-the-middle attacks.

Another access control mechanism for controlling the delivery process that acts on peer level is Lockr [15]. It is a privacy preserving access control mechanism for social networks in general. It is also applicable in BitTorrent P2P networks, for people to control the delivery of their personal content via them. The content owner issues digitally signed social attestations to all persons it has a social interaction with. A social attestation certifies the social relationship between two persons. In order to start exchanging pieces of the content, two peers need to verify their attestations during a social handshake, a form of zero-knowledge protocol. This access control mechanism is fine example of improved privacy in content delivery and in social networks in general. However, it still lacks support of flexible access control policies for the future P2P-based content delivery platforms.

III. MOTIVATION

To motivate our work we describe the following scenario. An international TV broadcaster (a content provider) wants to distribute live TV program to its clients (authorized users) using a P2P-based content delivery platform, based on the BitTorrent protocol. The TV broadcaster aims at achieving fine grained load balancing and optimization of its program delivery process, and restricting its program's availability only in one country (e.g., only in Slovenia) because of the digital rights issues, although it is broadcasting other programs in several countries. Furthermore, the TV broadcaster decides to deliver a service to clients under different conditions. Premium clients, for example, would receive higher content quality (e.g., HD video) for a certain amount of money, whereas basic clients would receive lower content quality (e.g., SD video) for free. This is beneficial from business perspective, as it can increase indirect earnings, and from technical perspective, since it can improve content delivery. Moreover, clients should be able to purchase certain service packages in which they will receive high content quality only during certain time periods, e.g., every day from 18 till 20 hours, during the most popular show. Analysis of the scenario elicited the following requirements.

Requirement 1: Fine grained load balancing and optimization of the delivery process: In BitTorrent live streaming swarm, none of the peers, except the content injector, has the whole content in advance, as seeds in regular swarms do. Instead, seeds in live streaming swarm are special peers with outstanding properties (e.g., high bandwidth), which are always unchoked by the content injector and have the same role in content delivery as the regular seeds – they improve the other peers' download performance. The seeds are often purposely set by the content provider and behave as a small Content Delivery Network (CDN) [6].

In order to achieve fine grained load balancing and optimization of the delivery process, the content provider (TV broadcaster) can create and maintain a hierarchical structure of seeds in the live streaming swarm, analogical to a hierarchical CDN [16]. This structure is formed by separation of the seeds into layers (levels) according to the priority assigned to them by the content provider (Fig. 1) and placing the seeds at strategic locations. The greater the priority of the seeds a layer contains is – the higher it appears in the structure. The value of the priority defines the level of precedence a seed has among the other peers in the live streaming swarm (seeds and leeches). Normally, the content injector and the seeds establish a connection to any peer in the swarm regardless of its priority, as long as they have a free connection. However, when a lack of free connection occurs, the connections with seeds having lower priorities or with leeches will be terminated in favour of seeds having greater priorities.

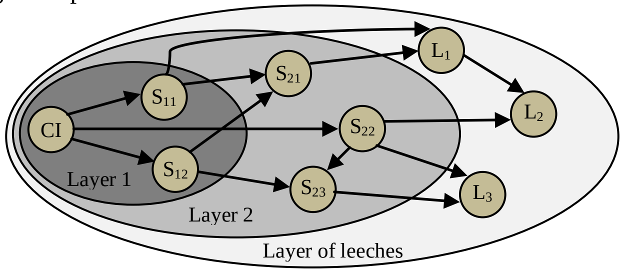


Figure 1. Hierarchical structure of a live streaming swarm: the content injector (CI) is not part of any layer; the seeds (S) from layer 1 have a priority (e.g., 20) greater than the seeds from layer 2 (e.g., 10); the leeches (L) are all placed in one layer and do not have any priority.

Two mechanisms are needed for the process of creation and maintenance of a hierarchical structure of seeds in a live streaming swarm.

Sub-requirement 1.1: Automatic introduction of a seed: Seeds download content only from the content injector or other seeds, which explicitly know them by maintaining lists of their identifiers (e.g., IP address and port number). However, these lists are maintained manually – something that becomes impractical for a large swarm (like in the scenario above) and very difficult for creation and maintenance of a hierarchical layered structure. Therefore, a mechanism for automatic introduction of a seed in the live streaming swarm is needed, that will also place the seed in a specific layer of the hierarchical structure.

Sub-requirement 1.2: Suitable peer discovery: This mechanism is needed to enable quick transport of the content from the content injector towards the lowest level of the structure of seeds, and consequently to the regular peers (clients). Currently, none of the peer discovery mechanisms the BitTorrent protocol supports (e.g., the tracker [3] or the DHT [7] protocol), takes into consideration a hierarchical structure of a live streaming swarm.

Requirement 2: Restriction of the content delivery based on peer location: According to this requirement, only peers inside one country are allowed to receive an authorization credential and join the swarm. The physical location of a peer on country level can be determined by using the Internet geolocation technology. Although tactics for evasion of this technology do exist, it is considered sufficient for compliance with the legal regulations [17]. The CS protocol needs to be properly extended in order to take into consideration the output of the Internet geolocation technology in the access control decision.

Requirement 3: Provision of different content quality in the same swarm: The content provider needs to create only one content stream by using a scalable video coding technique, but encoded in several layers [18]. Then, by specifying in the authorization credentials which layers the holders are allowed to receive, peers can easily determine which content pieces should provide to them. For example, premium clients would be authorized to receive content pieces from all the encoding layers, while other clients – only from fewer layers.

Requirement 4: Temporal constraints: In addition to the previous requirement, the authorization credential can also specify temporal constraints, for example, when the allowed content layers would be provided to the clients. This can even provide a basis for business models in the content delivery process by creating different service packages for the clients.

IV. THE ENHANCED CLOSED SWARMS PROTOCOL

We believe that after enhancement, the Closed Swarms protocol fulfills the requirements from Section III and becomes resistant to man-in-the-middle attacks. Before presenting the proposed enhancements of the CS protocol, in short we describe the format of the authorization credential and the message exchange process in the CS protocol, explained in detail in [11].

The authorization credential (called Proof of Access) of an arbitrary peer A (1) contains information about: the specific swarm – its identifier (SwarmID) and public key (K_S); the credential holder, defined by its public key (K_A); and the expiry time of the credential (ET). The credential issuer, usually the content provider in correlation with a payment system[1], digitally signs this information with the private key of the swarm (K_S^{-1}). The authorization credential is valid only when all the fields and the digital signature are correct.

$$SwarmID, K_S, K_A, ET, \{SwarmID, K_S, K_A, ET\}_{K_S^{-1}} \quad (1)$$

Two peers, an initiator – peer A, and a swarm member – peer B, exchange their credentials in a challenge-response message exchange process:

$$A \rightarrow B: SwarmID, N_A \quad (2)$$

$$A \leftarrow B: SwarmID, N_B \quad (3)$$

$$A \rightarrow B: PoA_A, \{N_A, N_B, PoA_A\}_{K_A^{-1}} \quad (4)$$

$$A \leftarrow B: PoA_B, \{N_A, N_B, PoA_B\}_{K_B^{-1}} \quad (5)$$

[1] The credential issuer signs credential for all swarms it is responsible for, by using their private keys. Although there is no specific protocol of issuing the credentials, the process is explained in detail in [11].

First, with (2) and (3) they exchange the identifier of the swarm they want to join/are part of and randomly generated nonces (N_A/N_B). Then, with (4) and (5) they exchange their credentials (PoA_A/PoA_B) followed by a concatenation of the previously exchanged nonces and the credential, digital signed with their private keys (K_A^{-1}/K_B^{-1}).

The requirements from Section III can be satisfied by using an access control based on flexible authorization framework and proper policy enforcement. A number of distributed frameworks have already been proposed in the past [19]. Here, we aim at integrating such distributed authorization framework in the CS protocol. Furthermore, although the protocol uses authorization credentials containing public key for owner identification, random nonces for message freshness and digital signatures for message authentication, it still remains vulnerable to man-in-the-middle attacks. An attacker can interfere in the communication between two authorized peers by simply relaying the messages between them. After the authorized peers successfully finish the protocol and start the content delivery, the attacker will be able to read all the exchanged content pieces, since they are not encrypted. We propose encryption of the content pieces with a shared secret key as a countermeasure for this attack.

The format of the extended authorization credential is as follows:

$$SwarmID, K_S, K_A, ET, Rules_A, \{SwarmID, K_S, K_A, ET, Rules_A\}_{K_S^{-1}} \quad (6)$$

The newly introduced field – Rules contains conditions under which the credential holder is authorized by the credential issuer to join the swarm and receive the requested service (e.g., content quality, level of prioritized treatment). The format of this field, described with the ABNF notation [20], is given below:

```
Rules = [General] [Per-piece]                                   (7)
General = conditions                                            (8)
Per-piece = conditions                                          (9)
conditions = condition [log-operator conditions]  /
             "(" conditions ")"                                 (10)
log-operator = "and" / "or"                                     (11)
condition = variable operator value /
            variable operator variable                          (12)
operator = "=" / "!=" / "<" / "<=" / ">" / ">="                 (13)
variable = 1ALPHA *99(ALPHA / DIGIT)                            (14)
value = 1*10DIGIT / 1*10DIGIT "." DIGIT /
        """ 1*10ALPHA """                                       (15)
```

The Rules field contains two groups of conditions: a general group and per-piece group. The former contains conditions evaluated every time the credential holder connects to another peer, as well as at specific time (in case of time conditions), whereas the latter contains conditions evaluated on every piece request from the credential holder. In each condition a value of a specific environment variable is compared to other variable or a predefined value. The values of the environment variables are dynamically assigned from the environment of the evaluating peer or from another field, as described later. Each group of conditions is positively evaluated only if the compound logical sentence produces a truth value.

Having on mind the roles of peers A and B, the extended and modified message exchange process goes as follows:

$$A \rightarrow B: Version_A, SwarmID, N_A \quad (16)$$

$$A \leftarrow B: Version_B, SwarmID, N_B \quad (17)$$

$$A \rightarrow B: PoA_A, ReqService_A, \{N_A, N_B, PoA_A, ReqService_A\}_{K_A^{-1}} \quad (18)$$

$$A \leftarrow B: PoA_B, Info_B, Peers_B, \{K_{AB}\}_{K_A}, \left\{N_A, N_B, PoA_B, Info_B, Peers_B, \{K_{AB}\}_{K_A}\right\}_{K_B^{-1}} \quad (19)$$

$$A \leftarrow B: Info_B, \{N_A, N_B, Info_B\}_{K_B^{-1}} \quad (20)$$

First, the peers exchange the latest version of the protocol they support (Version), together with the swarm identifier and the randomly generated nonce, with (16) and (17). Then, peer A sends its authorization credential and specifies the service properties (ReqService) it wants to receive with (18). Next, peer B evaluates the service request. If it is according to peer A's authorizations and if peer B can provide the requested service (for example it has an available connection – a free or one to a peer with lower priority that can be terminated), it will enable upload to peer A. Otherwise, upload will be disabled. In both cases, it will first send (19) in order to clarify the process outcome (Info) and to recommend other swarm members for contacting (Peers) to peer A. In positive case (19) will also contain a symmetric key (K_{AB}), generated by peer B and encrypted with peer A's public key, which will be used for encryption of the provided service – the content pieces. On the other hand, in negative case this field will be empty. After a positive (19), peer B starts to upload content to peer A. It also continues to verify the validity of the peer A's credential and to evaluate every piece request according to its authorizations. When a violation occurs, it will send (20) as notification and it will stop uploading. In addition, the protocol will be aborted when one of the peers sends an invalid credential, an incorrect digital signature or a different swarm identifier.

The positive outcome of the message exchange process is one way upload, from peer B to peer A. If peer B is also interested in downloading content from peer A while uploading, it needs to start the same exchange process, but now acting as an initiator.

The formats of the newly introduced fields are as follows. First, the Version field is two bytes and states the protocol version. For example, 02_{HEX} denotes the enhanced CS protocol version. Since this or any future protocol extension or modification results in a new version, peers need to be aware of the versions they support in order to have successful communication. In this way, means for backward compatibility between CS protocol versions can be possible. Next, the description of the ReqService field format, by using the ABNF notation, is:

```
ReqService = ["(" assignment ")"] ["," ReqService] (21)
assignment = variable "," value                    (22)
variable = 1ALPHA *99(ALPHA / DIGIT)               (23)
value = 1*10DIGIT / 1*10DIGIT "." DIGIT /
        """ 1*10ALPHA """                          (24)
```

It contains pairs that actually define an assignment of a certain value to a specific environment variable at the evaluating peer. These values must be assigned to the environment variables before evaluation of the conditions in the Rules field, since they influence the evaluation of the policies from the Rules field. The ReqService field can contain information such as requested content quality or level of prioritized treatment. Furthermore, the Info field is two bytes and specifies the identifier of predefined information that clarifies the protocol outcome. This information can confirm that the upload is enabled or state the reason why it is disabled. For example 01_{HEX} means unauthorized service properties requested. Finally, the Peers field is a set of maximum 5 pairs, each denoting a swarm member. A pair contains either IP address (IPv4 or IPv6) or DNS name of the member, together with its port number.

In conclusion, the enhanced CS protocol is an access control mechanism that acts on a peer level that enables peers to exchange the authorization credential and requested service properties between them in a secure manner.

V. DISCUSSION

Together with our proposed enhancements, the CS protocol fulfills the requirements from Section III, and becomes resistant to man-in-the-middle attacks.

To begin with, the introduction of Rules and ReqService fields fully satisfies the desired requirements 2-4, as well as the sub-requirement 1.1. The Rules field provides creation of expressive and flexible access control policies. These policies are contained in the authorization credential itself which makes their modification easy and dynamic. The policies can be tailored to several groups of peers in the swarm, distinguishable on the basis of various criteria, such as role in a swarm (seed or leech), priority, location and allowed content quality (number of stream layers), during different time periods. Now, seeds can automatically join the hierarchical swarm only by receiving appropriate authorization credentials. However, every peer must first explicitly request the properties of the service it wants to receive by specifying them in the ReqService field, in order to have its policy evaluated correctly. In this way, together with the help of the notifications in the Info field, they can even negotiate (to some extent) the service properties they want to receive.

The access control diagram of a request for service is illustrated in Figure 2 (based on [21]). The initiator – peer A sends its authorization credential and specifies the service properties it wants to receive to the swarm member – peer B with message (18). The Rules field is passed to the peer B’s Access Control Decision Function (ADF), where the embodied polices are evaluated. On the other hand, the values from the ReqService field are assigned to specific environment variables and together with other environment variables are taken into account during evaluation of the policies. The peer B’s Access Control Enforcement Function (AEF) grants or denies the access to the requested service according to the evaluation of the specified policies.

An example of information provided to the evaluating peer’s ADF, i.e. the contents of the Rules and ReqService fields sent by a seed, together with needed environment variables, applicable in the described scenario above is given in Figure 3. The Rules field denotes that the seed is authorized by the credential issuer to join the swarm only when it is located in Slovenia and requests high content quality (e.g., HD video) and prioritized treatment appropriate for level 2 seed. According to the explicitly requested service properties in the ReqService field by the seed and the values of the specific environment variables at the evaluating peer, this policy is positively evaluated at the ADF and the seed is granted access to the swarm.

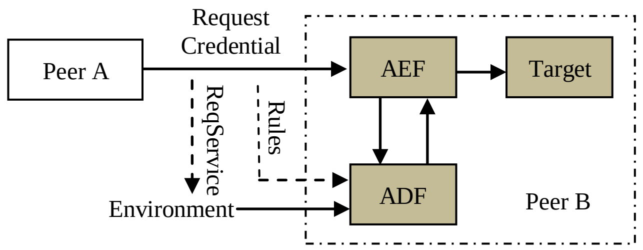


Figure 2. Access control diagram of request for service with the enhanced CS protocol (based on [21]).

Rules:
General:
GEOLOCATION = 'SI' and
PRIORITY <= 10 and
CONTENT_QUALITY <= 3
Per piece:
ReqService:
(PRIORITY, 10),
(CONTENT_QUALITY, 3)
Environment:
GEOLOCATION = 'SI'

Figure 3. Contents of the *Rules* and *ReqService* fields sent to a closed swarm member by a level 2 seed (Fig. 1) and the values of the environment variables at the swarm member.

Furthermore, the newly introduced Peers field provides a peer discovery mechanism applicable in hierarchically structured live streaming swarm, which satisfies sub-requirement 1.2 from Section III. The peer discovery mechanism goes as follows. Every peer first contacts the content injector using the CS protocol. If it is authorized to enter the swarm, it will receive by the content injector a list of swarm members from the layer with the highest priority. Then it continues to contact the returned members and to receive information about other members from the swarm, until it creates the number of connections it needs. Peers return information about members from the same layer or the layer with one level lower priority, as long as this priority is greater than or equal to the initiator’s priority. This guarantees that peers will always download content only from peers with the same or higher priority in the structure.

In addition, the Version field provides means for backward compatibility. After two peers exchange the protocol version they support, the peer supporting the higher version can adapt and send appropriate messages to the version the other peer supports. However, this is applicable in specific cases and only to those peers that are not directly concerned with the higher version changes. For example, if the original CS protocol did contain the Version field, the basic clients from the described scenario could use the original protocol, but only when the requirement for restriction of the content delivery based on peer location is not mandatory.

Finally, by encrypting the exchanged content with a secret key we prevent a malicious peer to read it in an unauthorized manner. However, the purpose of the encryption is not to provide confidentiality of the provided service, but only to fight man-in-the-middle attacks. Also, it does not prevent explicit leakage of content to unauthorized peers, which still depends on the good behavior of the authorized peers.

In conclusion, all the desired requirements from Section III can be satisfied with our proposed enhancements, and means for backward compatibility can be achieved. Also, the vulnerability of the protocol to man-in-the-middle attack is fixed.

VI. Conclusion and Future Work

In this paper we have proposed several enhancements of an existing access control mechanism for BitTorrent P2P networks – the Closed Swarms protocol. The enhancements provide additional flexibility in access control mechanism, enabling fine grained security policies specification and enforcement. The enhancements fulfill a number of content providers' requirements and promise efficient, flexible and secure content delivery in future content delivery scenarios. Our future work includes integration of the proposed enhancements into the P2P-Next delivery platform (http://p2p-next.org) and their evaluation.

References

[1] Future Media Internet Task Force: Research on Future Media Internet. A white paper, (2009). Obtained on April 5, 2011 from: ftp://ftp.cordis.europa.eu/pub/fp7/ict/docs/netmedia/research-on-future-media-internet-2009-4072_en.pdf

[2] Schulze, H., Mochalski, K.: Internet Study 2008/2009 (2009). Obtained on April 5, 2011 from: http://www.ipoque.com/userfiles/file/ipoque-Internet-Study-08-09.pdf

[3] Cohen, B.: BitTorrent protocol specification, (2008). Obtained on April 5, 2011 from: http://www.bittorrent.org/beps/bep_0003.html

[4] Pouwelse, J. A., Garbacki, P., Wang, J., Bakker, A., Yang, J., Iosup, A., Epema, D. H. J., Reinders, M., van Steen M. R., Sips H. J.: Tribler: A social-based based peer to peer system. 5th Int'l Workshop on Peer-to-Peer Systems (IPTPS), Santa Barbara, USA (2006)

[5] Pandey, R.R., Patil KK.: Study of BitTorrent based Video on Demand Systems. International Journal of Computer Applications, Vol.1, No.11, pp. 29-33. Foundation of Computer Science, (2010)

[6] Mol, J.J.D., Bakker, A., Pouwelse, J.A., Epema, D.H.J., Sips, H.J.: The Design and Deployment of a BitTorrent Live Video Streaming Solution. In: 11th IEEE International Symposium on Multimedia, ISM '09, pp. 342-349. IEEE Computer Society, Los Alamitos (2009)

[7] Loewenstern, A.: DHT Protocol, (2008). Obtained on April 5, 2011 from: http://bittorrent.org/beps/bep_0005.html

[8] Cohen, B.: Incentives Build Robustness in BitTorrent, In: Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley, USA, (2003)

[9] International Telecommunication Union – Telecommunication Standardization Sector (ITU-T): Data networks, open system communications and security, Framework for secure peer-to-peer communications. ITU-T Recommendation X.1161, (2008)

[10] Gheorghe, G., Lo Cigno, R., Montresor, A.: Security and privacy issues in P2P streaming systems: A survey. Peer-to-Peer Networking and Applications. Springer, New York (2010)

[11] Borch N.T., Arntzen, I., Mitchell K., Gabrijelčič D.: Access control to BitTorrent swarms using Closed Swarms. In: Proceedings of the 2010 ACM Workshop on Advanced Video Streaming Techniques for Peer-to-Peer Networks and Social Networking, AVSTP2P '10, pp. 25-30. ACM, New York (2010)

[12] Zhang, X., Liu, D., Chen, S., Zhang, Z., Sandhu, R.: Towards digital rights protection in BitTorrent-like P2P systems. In: Proceedings of the 15th ACM/SPIE Multimedia Computing and Networking, San Jose, USA (2008)

[13] Jimenez, R., Eriksson, L, Knutsson, B.: P2P-Next: Technical and Legal Challenges. The Sixth Swedish National Computer Networking Workshop and Ninth Scandinavian Workshop on Wireless Adhoc Networks, Uppsala, Sweden (2009)

[14] Harrison, D.: Private Torrents, (2008). Obtained on April 5, 2011 from: http://bittorrent.org/beps/bep_0027.html

[15] Tootoonchian, A., Saroiu, S., Ganjali, Y., Wolman, A.: Lockr: better privacy for social networks. In: Proceedings of the 5th international conference on Emerging networking experiments and technologies, CoNEXT '09, pp. 169-180. ACM, New York (2009)

[16] Andreev, K., Maggs, B.M., Meyerson, A., Sitaraman, R.K.: Designing overlay multicast networks for streaming, In: Proceedings of the fifteenth annual ACM symposium on Parallel algorithms and architectures, SPAA '03, pp. 149-158. ACM, New York (2003)

[17] Muir, J.A., van Oorschot, P.C.: Internet geolocation and evasion. Technical Report TR-06-05, Carleton University, School of Computer Science (2006)

[18] Asioli, S., Ramzan, N., Izquierdo, E.: Efficient Scalable Video Streaming over P2P Network. User Centric Media, LNICST, vol. 40, pp. 153-160. Springer, Heidelberg (2010)

[19] Chapin, P., Skalka, C., Wang, X.S.: Authorization in trust management: Features and foundations. ACM Computing Surveys, Vol. 40, pp.1-48. ACM, New York (2008)

[20] Crocker, D., Overell, P.: Augmented BNF for Syntax Specifications: ABNF. Request for Comments (RFC) 5234, (2008)

[21] International Telecommunication Union – Telecommunication Standardization Sector (ITU-T): Data networks, open system communications and security, Information technology – Open systems interconnection – Security frameworks for open systems: Access control framework. ITU-T Recommendation X.812, (1995)

Enhancing System-Called-Based Intrusion Detection with Protocol Context

Anyi Liu*, Xuxian Jiang†, Jing Jin*, Feng Mao‡, and Jim X. Chen*
*Department of Computer Science
George Mason University, Fairfax VA 22030
Email: aliu1, jjin3, jchen@gmu.edu
†North Carolina State University
Raleigh, NC 27695
Email: jiang@cs.ncsu.edu
‡EMC
Santa Clara, CA 95054
Email: fengmao@acm.org

***Abstract*—Building an accurate program model is challenging but vital for the development of an effective host-based intrusion detection system (IDS). The model should be designed to precisely reveal the intrinsic semantic logic of a program, which not only contains control-flows (e.g., system call sequences), but also data-flows as well as their inter-dependency. However, most existing intrusion detection models consider either control-flows or data-flows, but *not* both or their interweaved dependency, leading to inaccurate or incomplete program modeling. In this paper, we present a semantic flow-based model that seamlessly integrates control-flows, data-flows, as well as their inter-dependency, thus greatly improving the precision and completeness when modeling program behavior. More specifically, the semantic flow model describes program behavior in terms of basic *semantic units*, each of which semantically captures one essential aspect of a program's behavior. The relationship among these semantic units can be further obtained by applying the protocol knowledge behind the (server) program. We show that the integrated semantic flow model enables earlier detection and prevention of many attacks than existing approaches.**

***Keywords*-Intrusion detection; System calls; Protocol specification; Context.**

I. Introduction

Building an accurate program model is challenging but vital for the development of an effective host-based intrusion detection system (IDS). A strict model will likely generate alerts with high false positives while a loose model might not detect any advanced evasive attacks. To improve the detection accuracy, a number of models [4], [5], [12] have been proposed to precisely capture the intrinsic semantic logic of a program. Particularly, due to the efficiency and convenience in collecting system call logs as well as rich semantics of collected logs, system calls have been widely leveraged to build program models. For example, Forrest et al. [7] uses normal system call sequences to model program behavior and considers any violation as an intrusion; Gao et al. [5] applies a gray-box approach to reconstruct program execution graph and is able to detect anomaly system call sequences when any inconsistency is observed; Sekar et al. [1] leverages system call arguments to obtain a model that describes the inherent data-flow dependency.

From another perspective, note that a program's semantic logic usually contains control-flows (e.g., system call sequences), data-flows (e.g., system call argument relationships), and their inter-dependency. However, existing techniques consider either control-flows or data-flows, but *not* both, resulting in an inaccurate or incomplete program modeling. This weakness could be potentially exploited by advanced attackers to avoid their detection. For example, Wagner et al. [13] demonstrates that the mimicry attack can effectively evade the detection from system call sequence-based models and related IDSes.

To address the weakness, we present a new semantic flow-based model that naturally integrates control-flows, data-flows, and their inter-dependency. Different from previous program models, the semantic flow model describes program behavior in terms of basic *semantic units*. With collected system call sequences, arguments, as well as related run-time context information, each semantic unit semantically describes one essential aspect of a program's behavior. In addition, with the protocol knowledge behind the (server) program, the interweaved dependency among these semantic units can be naturally extracted and modeled. For example, the possible *data-control* relation describes the dependency from system call arguments to subsequent system calls and the *data-data* relation reveals the inherent semantic dependency among different system call arguments. Specifically, when compared with existing approaches, our semantic flow approach has the following three key advantages: (1) Logical integration of control-flows and data-flows. (2) Protocol-aware semantic analysis. (3) Early and accurate detection.

We have applied the semantic flow model to characterize most popular server programs (e.g., `httpd` and `ftpd`). For each one of them, we are able to observe those basic semantic units and then construct their semantic relations. The experimental results with real world attacks, including both control-flow and data-flow exploits, show that the semantic flow model can immediately detect them once any violation to the normal semantic flow model occurs, resulting in much earlier detection and prevention than existing approaches. We believe that the semantic flow model holds great promise for more precise and complete host-based intrusion detection.

II. Related Work

To construct a program behavior-based anomaly detection model, various approaches have been proposed. Starting from the work of Forrest et al. [7], the *black-box* approaches [12], [13] model the normal program behavior (e.g., based on system calls) and then detect intrusions by identifying anomaly within observed system calls. The *white-box* approaches apply static analysis on either source code [9], [14] or binary [6] to build program models. And the *gray-box* approaches further leverage the program runtime information to improve the accuracy of anomaly detection models [2], [4], [5]. Our work is more closely related to data-flow anomaly detection [1], which examines inherent data-flow dependencies among system call arguments to make the model more robust. However, none of the previous works utilizes protocol knowledge behind the modeled program, which inspires our work to fully exploit the semantic meanings of system call arguments and build semantic dependencies among extracted semantic units. Our approach makes one step further and allows to derive more complicated semantic dependencies, e.g., *data* → *control* and *control* → *data* relations. As such, our approach enables the construction of more accurate and complete program models for anomaly detection.

III. An Illustrative Example

In this section, we illustrate the semantic flow model with a representative example, i.e., the Apache web server. For each incoming web request, we can divide the corresponding Apache behavior (or the `httpd` worker daemon) into the following four logical phrases: (1) The Apache server waits for a client request, and prepares a worker thread. (2) The worker thread handles the request and process it. (3) The server generates response for the incoming request. (4) After the response is sent back to the clinet, the network socket used for the communication is closed.

Figure 1 shows the Apache behavior when answering an incoming request, both from a network/OS viewpoint as well as the semantic flow viewpoint. Specifically, Figure 1(a) contains a list of invoked system calls as well as their arguments while Figure 1(b) highlights some inherent dependencies within these system calls and their arguments. Instead of syntactically grouping adjacent system calls into sequences or mining arguments for possible relationships, the semantic flow model aims to leverage the protocol logic that has been implemented by the modeled program to characterize its behavior. In addition, we can verify the program logic by reconstructing the implemented protocol with semantic-sensitive information from observed system calls, arguments, or other run-time context information.

The above example illustrates system calls and arguments are strongly connected. The key to obtain their relationships lies in protocol-aware semantic analysis. Partial analysis on system call sequences or arguments without knowing their semantic implications will lead to incomplete and imprecise program modeling.

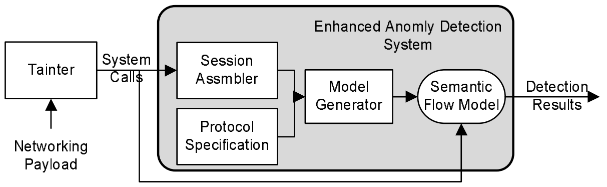


Figure 2. Overview of semantic flow model

IV. Designing Semantic Flow Model

A. Terminologies

In this section, we first define the terminologies that will be used throughout this paper.

- We denote the set of system calls and the set of system call arguments as $\mathcal{C} = \{c_i \mid 1 \leq i \leq m\}$ and $\mathcal{A} = \{a_i \mid 1 \leq i \leq n\}$, respectively. For simplicity, the return value of a system call will be considered as one of its arguments. We also represent the control-flow relation $\mathcal{R}_c$ on $\mathcal{C}$ as $\mathcal{R}_c \subseteq C \times C$ and the data-flow relation $\mathcal{R}_d$ on $\mathcal{A}$ as $\mathcal{R}_d \subseteq A \times A$. Note that existing models that are built upon $\{\mathcal{C}, \mathcal{R}_c\}$ fall into the *control-flow model* category and others built based on $\{\mathcal{A}, \mathcal{R}_d\}$ belong to the *data-flow model* category.
- We log system calls and save them as a record in the form of $sc = \{n, A\}$, in which $sc.n$ is the name of the system call, $sc.A$ is the set of arguments. When processing system calls, we simply consider them as an array sc. An argument $sc[i].a_j \in A$ is assigned by a value and a semantic type, which denoted as $sc[i].a_j.value$, and $sc[i].a_j.type$, respectively.
- The *semantic set* S_{sem} is the super set of system calls and arguments and can be simply represented as $S_{sem} = \mathbf{2}^{\mathcal{C} \cup \mathcal{A}}$. The semantic relation $\mathcal{R}_s$ on S_{sem} is similarly denoted as $\mathcal{R}_s \subseteq S_{sem} \times S_{sem}$. We call models build upon $\{S_{sem}, \mathcal{R}_s\}$ as *semantic flow models*.

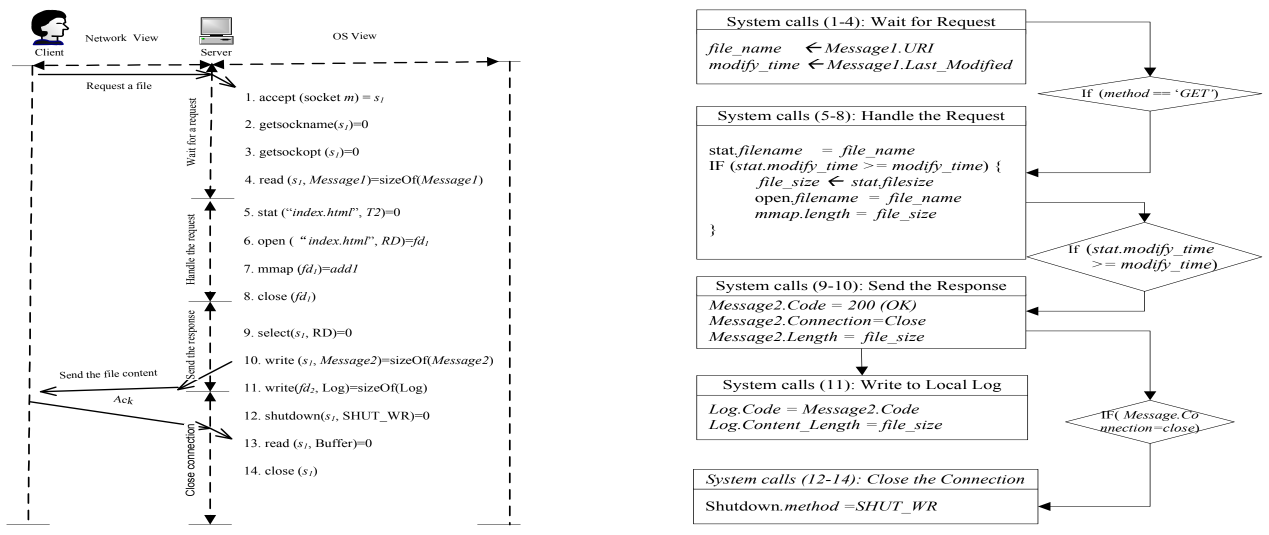
B. System Overview

Figure 2 shows our semantic flow-based intrusion detection model, which has three main components: (1) The *session assembler* propagate tainted networking payload to invoked system calls within a networking session (Section V-A); (2) The *protocol selector* leverages protocol knowledge and matches semantic units with pre-defined protocol specification (Section V-B); (3) The *semantic flow model generator* will reconstruct semantic relations among semantic units and build the program behavior model as the corresponding semantic flow model (Section V-C). The doted line circulated the major components.

V. Methodology

A. Networking Input Propagation

To correlate the networking traffic with system calls and their arguments, we use tain techniques, which have been discussed in [11], [15]. Specifically, we initially *taint* the string in packet payload received by networking-related


Client
Network View
Server
OS View
Request a file
Wait for a request
1. accept (socket m) = s1
2. getsockname(s1)=0
3. getsockopt (s1)=0
4. read (s1, Message1)=sizeOf(Message1)
Handle the request
5. stat ("index.html", T2)=0
6. open ("index.html", RD)=fd1
7. mmap (fd1)=add1
8. close (fd1)
Send the response
9. select(s1, RD)=0
10. write (s1, Message2)=sizeOf(Message2)
Send the file content
Ack
11. write(fd2, Log)=sizeOf(Log)
12. shutdown(s1, SHUT_WR)=0
13. read (s1, Buffer)=0
14. close (s1)
Close connection
System calls (1-4): Wait for Request
file_name ← Message1.URI
modify_time ← Message1.Last_Modified
If (method == 'GET')
System calls (5-8): Handle the Request
stat.filename = file_name
IF (stat.modify_time >= modify_time) {
file_size ← stat.filesize
open.filename = file_name
mmap.length = file_size
}
If (stat.modify_time >= modify_time)
System calls (9-10): Send the Response
Message2.Code = 200 (OK)
Message2.Connection=Close
Message2.Length = file_size
System calls (11): Write to Local Log
Log.Code = Message2.Code
Log.Content_Length = file_size
IF(Message.Connection=close)
System calls (12-14): Close the Connection
Shutdown.method =SHUT_WR

(a) Network and OS views (b) Semantic dependencies between system calls

Figure 1. The simplified network/OS view (left) and the semantic flow (right) of Apache when answering an incoming request. In the OS view, the recorded system calls are sequentially labeled (some of them are omitted for readability). The semantic flow highlights some inherent dependencies among invoked system calls and their arguments.

system calls, such as sys_socket. We also instrument the data movement instructions (e.g., `mov`) and arithmatic/logic instruction (e.g., `add`, `mul`, `and`), such that the tainted string can be propagated through the lifetime of string processing. For a data movement instruction, we check whether the source operand is marked. If yes, we will annotate the destination operand, which can be a register or a memory location, with the source operand's annotation, i.e. its offset in the original message. If the source operand is not marked, we will simply unmark the destination operand. If two marked operands appear in the same instruction, we will union their annotations (e.g., for the `add` operation, the result is the union of the operands if they are both marked).

Then, we need to re-map system call arguments based on *semantic types*, based on the protocol specification. Semantic types are used to more precisely capture the semantic meaning of system call arguments as they cannot be naturally obtained from the original argument types according to the neutral system-wide system call convention. An an example, the first argument of the `open` system call, which originally defined as a *string*, is now redefined as the `Filename` semantic type. Its return value will be similarly redefined as the semantic type `FileDescriptor`, instead of *int*.

```
Name  Filename         Flag    FileDescriptor
open  ``/etc/passwd''  ``RD''   6
```

Besides the knowledge of system call convention, we further use protocol specification to extend our knowledge of semantic meaning. We used the technique in [10] to discover protocol formatting specification. In the following, we illustrate the snippet of `SERVICE_REQUEST` specification for the HTTP protocol.

```
<SERVICE_REQUEST>
SYSCALL = Read(FD, BUFFER, RET)
FD = %Accepted_Socket
BUFFER =((GENERAL_HEADER|REQUEST_HEADER)\13\10)*\13\10
GENERAL_HEADER = %Method %URI %Dummy\13\10
REQUEST_HEADER = From|Host|If-Match|Last-Modified...=\%VALUE
RET = sizeOf(BUFFER)
```

Recall the *read* system call in the line 4 of Figure 1(a). We can capture its semantic meaning with the above protocol specification. More specifically, the file descriptor equals to the accepted socket number after *accepting* the incoming request. The argument *BUFFER* contains two fields, `GENERAL_HEADER` and `REQUEST_HEADER`, each of them can be further parsed into various sub-fields and eventually casted into more specific semantic types. For example, the `REQUEST_HEADER` field can be analyzed based on the following format:

```
From: Type = Email, Format = %username@%hostname
Host: Type = IP|Host_Name, Format = %{4B}|String
Last-Modified: Type = Date, Format = Timestamp
```

The first line states that the `From` field should be parsed as an email address. The second line specifies that the `Host` field should be defined as an IP address or a host name. The third line is to define the type of `Last-Modified` field as the default timestamp format.

B. Algorithm for Constructing Semantic Units

To describe the high-level functionalities of a networking protocol, we introduce the concept of *user session* $\mathcal{S}$ to represent a execution path of one server program, and *semantic unit* U, which intended to capture one essential aspect of modeled program behavior. As an example, the `accept` and the `close` system call are the starting point and the ending point of the user session shown in Figure 1. Semantic units comprises of a number of system calls, their arguments, as well as return values. In our current implementation, we organize semantic units from adjacent system calls based on whether they share the same file descriptors, filenames, or network sockets. In other words, adjacent system calls that manipulate the same file descriptor, filename, or network socket will be grouped to the same semantic unit. For example, the following system call sequence is a *semantic unit* as the three system calls `open`, `read`, and `close` are used to access a file named *"/etc/passwd"* by referring to the same file descriptor.

```
open("/etc/passwd", RD)=6,
read(6, buf)=123,
close(6)=0
```

Algorithm 1: *SemanticUnitExtraction(sc, U)*

```
input : A system call $sc$, and the semantic unit
        array $U$.
output: The updated array of semantic units $U$.
begin
    for $i=1$ to $N$ do
        for $j=1$ to $M$ do
            if $sc.a_i.type = U[i].a_j.type$ and
            $sc.a_i.value = U[i].a_j.value$ then
                $U[no\_of\_su] = UNION(U[i], sc)$;
                break;
            else
                no_of_su++; instantiate
                $U[no\_of\_su]$;
                $U[no\_of\_su] = sc$;
                break;
end
```

With collected system calls, our algorithm *SemanticUnitExtraction(sc, U)* groups them into different semantic units. The algorithm works as follows: It maintains a global variable *no_of_su* (initialized with 0) that keeps the current number of semantic units in $\mathcal{S}$. For each collected system call sc, the algorithm will be check whether it is a member of the existing semantic unit $U[no\_of\_su]$. If yes, it will be added to $U[no\_of\_su]$ (via the UNION(U[i], sc) function) and the global variable remains intact. Otherwise, a new semantic unit will be created and the *no_of_su* will be incremented by 1. We need to point out that adjacent system calls manipulate the same file descriptors, file names, or sockets will be grouped into the same semantic unit. However, not all system calls that manipulate the same file descriptor, filename, or socket will be included into the same semantic unit. This design choice makes the Algorithm 1 easy to implement.

Example 1 We illustrate the algorithm by revisiting the simplified *httpd* case in Section III. First, when the first system call – `accept` – is encountered, it will be included in a new semantic unit U_1. The following three system calls (at line 2-4) will also be grouped into the same semantic unit U_1 as they essentially wait for (and then receive) incoming requests and manipulate the same socket (as the `accept` system call). After that, the *stat* system call at line 5 will start with a new semantic unit U_2 as it is not related to the previous socket, and their main purpose is to handle the request. Moreover, since the following system calls at lines 6-8 handles the same file named *"index.html"* with the *stat* system call, they will join with the second semantic unit because they send back the response to the requesting client. In a similar manner, system calls at line 9-10 (U_3) send back the response to the requesting client; the requesting behavior is locally recorded at line 11 (U_4); and the communication channel is finally shutdown and closed at lines 12-14, U_5).

C. Constructing Semantic Specification

Different from previous approaches that solely depend on either control-flow or data-flow relations, a semantic relation flow $\mathcal{R}_s$ covers the inter-dependencies between them. In this paper, we focus our semantic flow relations in three categories: Data → Control, Data → Data, and Control → Data, which illustrate in Table I.

VI. Evaluation

We have implemented a proof-of-concept system that runs on the Fedora 13. The system calls, arguments, and return values are collected with a customized loadable Linux kernel module (LKM). The experiments are performed on a PC with Intel Core 2 Due 2.83GHZ CPU and 2G physical memory.

A. Effectiveness

We evaluate the effectiveness of our approach with a number of real-world attacks that are publicly obtained from [3]. Table II contains the list of five experimented server programs as well as attacks exploiting their vulnerabilities. Within these attacks, two of them are control-flow attacks which directly hijack the control flow of vulnernable programs, while the other three are data-flow attacks that are able to manipulate security-critical data to evade traditional detection techniques. Since server programs of `wu-ftpd` and `ghttpd` are vulnerable to both control-flow and data-flow attacks, we simply use a subscript to differentiate them. For instance, we use *wu-ftpd*$_1$ to represent the control-flow attack and *wu-ftpd*$_2$ to represent the data-flow attack against `wu-ftpd`.

In the following, we use three examples to show that how the three types of semantic relations, i.e., *Data → Control*, and *Data → Data* are used to detect attacks.

Data → Control Violation Detection All versions of *wu-ftpd* before 2.6.1 contain a vulnerability that can be exploited to trigger a heap corruption vulnerability (CVE-2001-0550). The vulnerability is located in the *ftpglob* function, which fails to properly handle the FTP commands and consequently allows remote attackers to execute arbitrary commands via a $\sim\{$ argument [16].

Figure 3(a) shows the related semantic flow specification that will be violated by this attack. More specifically, there exist three related semantic units for the exploited *wu-ftpd* sub-session. The first semantic unit receives the command request from the client and interprets it to be a *CWD* command. The following semantic unit will actually execute the `CWD` command by invoking the `chdir` system call. The return value of `chdir` will determine the *code* field that will be later sent back to the client in the third semantic unit. The *code* field essentially notifies the client whether the operation is successful or not.

Our approach detected this attack when the server sent its response to the client via a *write* system call. Based on the ftp protocol, the raw command `CWD pathname`

Category	Subcategory	Meaning	Example
Data → Control	**Single data to control relations**	Relations that a single argument determines follows system calls	In *ftp* protocol, the argument `CWD` determines system call `chdir`
	Multiple data to control relations	Multiple arguments together determine system calls later	The *readfds* and *writefds* arguments of *select* system call determine the following *read* or *write* system call
	Number of loops relations	Relations that arguments determine the number of system calls that will appear later	The argument `st_size` of system call `stat` determines the number of `write` system calls be invoked later
Data → Data	**Logical relations**	Relations that a single argument might determine future system calls	The return value of `-13` (meaning *Permission denied*) of `open` determines the error code `304` in the reply buffer.
	Numeric relations	Relations that evaluate two numeric values v_1 and v_2	*LargerThan*(v_1, v_2), *SmallerThan*(v_1, v_2), *EqualTo*(v_1, v_2)
	Timing relations	Relations evaluate two timing values d_1 and d_2	*Before*(d_1, d_2), *After*(d_1, d_2), and *At*(d_1, d_2)
Control → Data		Relations determine system calls to system call arguments	The system call `write` determines certain keywords in the reply buffer, such as *Code*, *Connection*, and *Length*

Table I
SEMANTIC RELATIONS R_s IN OUR FRAMEWORK

Program (version)	**Reference**	**Attack description**	**Program size(KB)**	**Total # of system calls**	**# of system calls in attack session**	**Violation**
wu-ftpd$_1$(2.6.1)	CVE-2001-0550	Heap corruption allows execute arbitrary commands via a $\sim\{$ argument to commands	2916	1372	2	*Data → Control*
ghttpd$_1$(1.4)	CAN-2001-0820	Long arguments passed to the `Log` function in `util.c` allows attackers to get shell	311	27	20	*Data → Control*
wu-ftpd$_2$(2.6.0)	S.Chen et al. [3]	Format string overwrite user ID	2916	15754	8	*Data → Data*
ghttpd$_2$(1.4)	S.Chen et al. [3]	Stack overflow to overwrite backup value of `ESI`	311	105	14	*Control → Data*
null-httpd(0.5)	S.Chen et al. [3]	Two `POST` commands corrupt CGI-BIN configure string	806	230	72	*Data → Data*

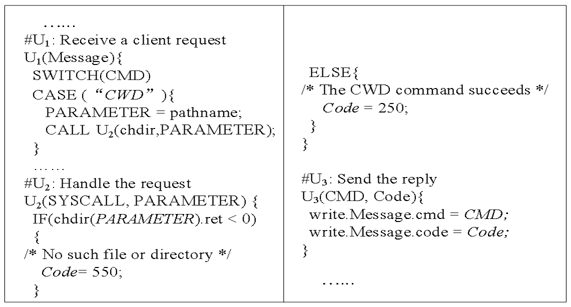
Table II
VULNERABLE SERVERS AND REAL-WORLD ATTACKS USED IN OUR EVALUATION

allows the client to change the current working directory to `pathname`. As such, in our semantic flow specification (Figure 3(a)), the semantic unit U_2 will invoke the system call `chdir`. After invoking the `chdir`, the server will notify the client with the return code either `250`(indicating "the CWD command is successful"), or `550`(meansing "No such file or directory").

When considering the actual attack sequence, it violates at least twice our semenatic flow specifications: First, there does not exist a subsequent `chdir` system call. Second, the response message will usually contain return code of `250` or `550`. For previous approaches that detect control injection attacks, the same attack could be detected when the attack invokes the *execve* system call to obtain a command shell (*"/bin/sh"*), which is much later than the detection point by our approach. Figure 3(b) shows the difference between the detection point by our approach and the detection point by other approaches.

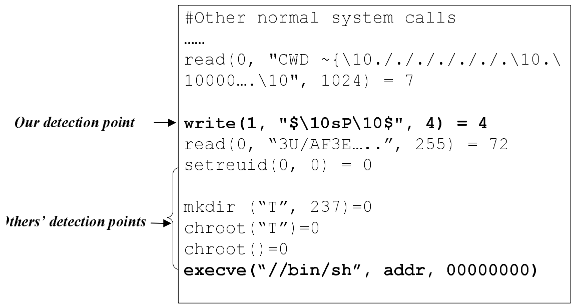
Data → Data Violation Detection The same *wu-ftpd* server (versions 2.6.0 and earlier) contains another vulnerability, i.e., a format string bug (CVE-2000-0573), which can be exploited with a specially-crafted string to the *SITE EXEC* command. Instead of overwriting the return address on the stack, this attack use format string to overwrote a security-critical variable $pw \rightarrow pw\_uid$ to 0. After that, the attack further established another data connection and issues a *get* command, which essentially invoked the function *getdatasock()* in the wu-ftpd server. Due to the corruption of $pw \rightarrow pw\_uid$, the execution of the function will set the `EUID` of the process to 0, elevating the process privilege to the super-root. As such, an originally non-privileged user is able to access the system with the root privilege. This overall exploitation is a typical data-flow attack [3].

It is interesting to point out that data-flow-based anomaly detection is also able to detect this attack. As discussed in [1], this attack can be detected as a violation of the equality relation between the `seteuid` system call and another `setuid` system call (in function *pass()*). However, the root cause of this attack is that the attacker crafts a format string, in the form as `SITE EXEC` $aaabcd\%.f\%.f\%.f\%...\%d...|\%.8x$, to overwrite $pw \rightarrow pw\_uid$ to 0. And our semantic flow-based detection is able to identify this attack when the *Equal* relation between the file name `execve` invoked and the file name in reply message is been violated, which is earlier than the previous detection point.



```
  ……
#U1: Receive a client request
U1(Message){
 SWITCH(CMD)
 CASE ( "CWD" ){
   PARAMETER = pathname;
   CALL U2(chdir,PARAMETER);
 }
  ……
#U2: Handle the request
U2(SYSCALL, PARAMETER) {
 IF(chdir(PARAMETER).ret < 0)
 {
/* No such file or directory */
   Code= 550;
 }
 ELSE{
/* The CWD command succeeds */
   Code = 250;
  }
}

#U3: Send the reply
U3(CMD, Code){
 write.Message.cmd = CMD;
 write.Message.code = Code;
}

   ……
```

(a) Partial semantic specification for wu-ftpd



```
#Other normal system calls
……
read(0, "CWD ~{\10./././././.\10.\
10000….\10", 1024) = 7

write(1, "$\10sP\10$", 4) = 4
read(0, "3U/AF3E…..", 255) = 72
setreuid(0, 0) = 0

mkdir ("T", 237)=0
chroot("T")=0
chroot()=0
execve("//bin/sh", addr, 00000000)
```

(b) Logged (attack) system calls and the detection points

Figure 3. A control-flow attack based on the wu-ftpd heap corruption vulnerability (CVE-2001-0550): The system call sequences shown in Figure 3(b) violates the semantic specification in Figure 3(a).

VII. Conclusion

In this paper, we have presented a semantic flow-based host intrusion detection model that seamlessly integrates control-flow and data-flow dependencies. When compared with existing approaches, which only focus on control-flows, or data-flows but not both, our approach greatly improves the accuracy and completeness of the obtained program behavior models. An efficient algorithm is presented to accurately extract basic semantic units, each of which characterizes an essential aspect of the modeled program behavior, and then obtain the semantic dependencies among them. Our experimental results show that our model enables earlier detection and prevention of many attacks than existing approaches and holds great promise for more precise and complete host-based intrusion detection.

Acknowledgment

We would like to thank Zhiqiang Lin from Purdue University for providing the source code of AutoFormat, the networking protocol reverse engineering tool.

References

[1] S. Bhatkar, A. Chaturvedi, and R. Sekar. "Dataflow Anomaly Detection". In *S&P '06: Proceedings of 2006 IEEE Symposium on Security and Privacy*, Oakland, CA, USA. pp. 48-62. May 2006.

[2] M. D. Bond, V. S. Kathryn, S. McKinley, V. Shmatikov. "Efficient Context-Sensitive Detection of Real-World Semantic Attacks". In *PLAS '10: Proceedings of the 5th ACM SIGPLAN Workshop on Programming Languages and Analysis for Security*. pp. 1-10. ACM, 2010

[3] S. Chen, J. Xu, E. Sezer, P. Gauriar and R. Iyer. "Non-Control-Data Attacks Are Realistic Threats". In *USENIX Security '05: Proceedings of the 14th USENIX Security Symposium*, Baltimore, MD, USA. pp. 177-192. August 2005.

[4] H. Feng, J. Giffin, Y. Huang, S. Jha, W. Lee, and B. P. Miller. "Formalizing sensitivity in static analysis for intrusion detection". In *S&P '04: Proceedings of 2004 IEEE Symposium on Security and Privacy*, Oakland, California, USA, pp. 194-208. May 2004.

[5] D. Gao, M.K. Reiter and D. Song. "Gray-Box Extraction of Execution Graphs for Anomaly Detection". In *CCS '04: Proceedings of the 11th ACM Conference on Computer and Communications Security*. Washington, DC, pp. 318-329. October 2004

[6] J. Giffin, S. Jha, and B. Miller. "Efficient contextsensitive Intrusion Detection". In *NDSS '04: Proceedings of The 11th Annual Network and Distributed System Security Symposium*, San Diego, CA.

[7] S. A. Hofmeyr, S. Forrest, and A. Somayaji. "Intrusion Detection Using Sequences of System Calls". Journal of Computer Security (JCS), 6(3):151-180, 1998.

[8] C. Kruegel, D. Mutz, F. Valeur, and G. Vigna. "On the Detection of Anomalous System Call Arguments". *In ESORICS'03: Proceedings of the 8th European Symposium on Research in Computer Security*. Gjovik, Norway, pp. 101-118. October 2003

[9] L. C. Lam and T. Chiueh. "Automatic Extraction of Accurate Application-specific Sandboxing Policy". *In RAID '04: Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection*. French Riviera, France. pp. 1-20.

[10] Z. Lin, X. Jiang, D. Xu, and X. Zhang. "Automatic Protocol Format Reverse Engineering Through Context-Aware Monitored Execution". *In NDSS '08: Proceedings of the 15th Network and Distributed System Security Symposium*, San Diego, CA, February 2008

[11] P. Saxena, D. Akhawe, S. Hanna, F. Mao, S. McCamant, D. Song. "A Symbolic Execution Framework for JavaScript". *In S&P '10: Proceedings of IEEE Symposium on Security and Privacy*, Oakland, California, USA, May 2010.

[12] R. Sekar, M. Bendre, P. Bollineni and D. Dhurjati. "A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors". *In S&P '01: Proceedings of 2001 IEEE Symposium on Security and Privacy*, Oakland, California, USA. pp. 144re-155. May 2001.

[13] D. Wagner and P. Soto. "Mimicry attacks on host-based intrusion detection systems". *In CCS'02: Proceedings of the 9th ACM Conference on Computer and Communications Security*, Washington, DC, USA. pp. 255-264. Noverber 2002.

[14] D. Wagner and D. Dean. "Intrusion Detection via Static Analysis". *In S&P'01: Proceedings of 2001 IEEE Symposium on Security and Privacy*, Oakland, California. May 2001.

[15] H. Yin, D. Song, E. Manuel, C. Kruegel, and E. Kirda. "Panorama: Capturing system-wide information flow for malware detection and analysis". *In CCS'07: Proceedings of the 14th ACM Conferences on Computer and Communication Security*, pp. 116-127. October 2007.

[16] Wu-ftpd vulnerability. http://www.securityfocus.com/bid/3581/references (June 2, 2011)

StegoWeb: Towards the Ideal Private Web Content Publishing Tool

Tamás Besenyei, Ádám Máté Földes, Gábor György Gulyás, Sándor Imre
Department of Telecommunications
Budapest University of Technology and Economics
Magyar tudósok körútja 2., 1117 Budapest, Hungary
tamas@besenyei.net, foldesa@hit.bme.hu, gulyasg@hit.bme.hu, imre@hit.bme.hu

Abstract—**Privacy breaches through profiling constitute a considerable threat to users of Web 2.0 services. While many concepts have been proposed to address this issue by allowing users to encrypt, obfuscate, or otherwise conceal information of their choice, all have certain limitations. In this paper, we survey the available solutions, and propose a taxonomy for classifying them based on a revised evaluation scheme that builds upon our previous work. Our main contribution is a model that harnesses steganographic techniques in order to hide sensitive data, and the description of a proof-of-concept implementation thereof that allows a user to hide profile data on a website without installing any sort of software aside from a conventional web browser.**

Keywords-Web 2.0; web privacy; user content; steganography

I. Introduction

As the use of Web 2.0 services – most notably Social Networking Sites or SNSes – is becoming more and more widespread, the privacy-related questions of the sensitive information published there gain significance in a similar tact. The term ‘profiling’ is used to describe activities which involve collecting data about a person from various sources (e.g. customer preferences in a webshop and personal data published on social networking websites), and merging the pieces of information into a single record, called a profile. Since Web 2.0 services are based on user-created content, profilers can use these services to complement their profiles [4]. Accurate user profiles serve as useful bases for many dubious or outright malicious activities, including targeted advertising and dynamic pricing. This tendency is likely to get worse as real-time searching becomes a core feature in search engines, which makes revocation of information impossible. Therefore, this problem is gaining importance frighteningly fast.

The techniques of profiling have evolved greatly since the birth of the World Wide Web. When IP addresses were fixed, they could be used to identify a user on the Web. Later on, as Internet Service Providers adopted the use of dynamic IP addresses, the main basis of identifying a user became unique identifiers in HTTP cookies and, later, ‘Flash cookies’ or LSOs [2]. The evolution of tracking techniques is continuous; the concept of Evercookies [15] and the Panopticlick browser fingerprinting experiment [16] indicate that research and improvements in the area have certainly not concluded. Furthermore, information superpowers – service providers that offer a wide range of products to their users – are a major threat [4], because they can have access to various data about the user.

As such, there is a need for applications that protect the user against these actors through limiting the information of personal nature that a profiler potentially has access to. Our previous work [4] introduced such a piece of software called BlogCrypt, a Firefox extension that could encrypt and decrypt data on websites with as little user interaction as possible. In that paper, we showed that BlogCrypt was an efficient countermeasure against profiling, but, as it does not conceal encryption, users are likely to face countermeasures on Web 2.0 sites where encrypting or otherwise obfuscating user content is forbidden by the Terms of Use.

Our main contribution in this paper is a steganographic approach to this problem, which, albeit not a direct successor or an improved version of BlogCrypt, addresses the same issue as it did, but in a slightly different context. The main reason is that steganography is ‘expensive’, i.e. only a small amount of data can be stored with such techniques. Therefore, while BlogCrypt was a useful solution to encrypt blog posts, StegoWeb is more likely to be applicable in the context of profile data on SNSes. If our application is used for this purpose, a profiler will not be able to link our personal information to other data she has potentially obtained about us.

The paper is structured as follows. In Section II, we survey already existing implementations and concepts that are destined to hinder profiling, and provide a taxonomy for classifying them. Then, in Section III, we discuss our own implementation, and analyse it in terms of advantages and drawbacks. In Section IV, we evaluate our implementation from the aspect of key management, and propose some improvements. Section V describes how the concept can be used for identity management purposes. Finally, we conclude our work in Section VI.

II. Evaluation of Existing Solutions

In this section, we discuss the already existing solutions for the aforementioned issues, and categorise them into a taxonomy. Some of these solutions are discussed and classified in our previous work [4].

A. *Existing Solutions*

There are many different solutions for protecting user content on Web 2.0 sites. We discussed the merits and shortcomings of most of them in our previous work [4]. The currently available solutions can be categorised as follows:

1) Universal applications. These applications can cooperate with arbitrary services that provide a generic interface such as a textbox. Some of them (e.g. BlogCrypt and NOYB – Secret Messaging [9]) are implemented as standalone Firefox extensions, while others need additional external software to operate (e.g. FireGPG [10]).

2) Site-based applications. These programs (or models) are destined to work with a single Web 2.0 website. Examples of such applications include Lockr [11], FaceCloak [12] and FlyByNight [13]. Newer concepts include Persona [7], which is essentially a privacy-enhanced SNS. As a proof-of-concept implementation, the authors integrated their model with Facebook in such a way that the backend is a Facebook application. Another site-based application is FaceVPSN [8], which is a Firefox extension that allows users to import fellow Facebook users' profile data into a local database, and substitute the corresponding attributes with the locally stored information (if available) when the profile page of a user is loaded. Lastly, SeGoDoc [6] is essentially a middleware for encrypted storage on Google Docs. It is implemented as a Firefox extension, and works on-the-fly, completely automatically.

3) Models without available implementations. These are models that were published in academic research, but their implementation is not available. Examples for this are NOYB – Social Networking [14] and an unnamed community-based access control concept that – similarly to BlogCrypt – assumes server-side storage of encrypted data [5].

B. Evaluation Model

In our previous work [4], we discussed a categorisation scheme for the available solutions, and described the principles based on which its attributes are defined. The revised version of the categorisation scheme is summarised in Table I. We do not discuss the results that have already been published in our previous work, and listed only some solutions that have appeared since then, namely SeGoDoc, Persona, FaceVPSN and our own solution StegoWeb. Furthermore, we have done away with the attribute 'Autonomy', and defined new attributes and categories, too:

1) Key distribution. Possible categories: manual (M), partially automatised (PA), fully automatic (Auto).

2) Independence. Possible categories: operating system independent (OS), browser independent (B), service independent (S).

3) Realisation. Possible categories: external software (ExSw), browser extension (Ext), bookmarklet (B).

C. Taxonomy of Private Web Publishing Solutions

We have introduced a new taxonomy for these services, which is depicted on Fig. 1. The leading idea during the preparation of the taxonomy was to model how the user relates to the application before starting to use it. Therefore, the first set of attributes which we chose to branch the universe of access control applications for published data were gradual deployment, realisation and ease of installation. If gradual deployment is not possible, the application is realised as external software, or its installation is complicated, we put the solution into the category 'impractical'; otherwise it is labelled as 'practical'.

The fork of the category 'practical' has been chosen to be based on the independence of the application. If the program is not at least operating system independent, or it is service specific, it is classified as 'dependent', else it is put into the category 'independent'. The reason for this is that the user is likely to prefer solutions that can be used in several environments, e.g. if she intends to run an application both on her corporate computer with Windows and her home computer running Linux.

The split on category 'independent' has been based on discoverability and key distribution, because these factors have major influence on the security properties of the software. (We have not included the type of encryption, since it does not tell much about the security properties of the application.) If the presence of an application is discoverable, i.e. its discoverability attribute is 'crypto', or key management is not automatic, the solution is classified as 'recommended'. In other cases, it is assigned the label of 'smoothly usable'.

Lastly, the fourth split considers compromises and ease of usage. These factors have a major impact on user experience, so they are likely to influence the user's relation to the software in the long run. Based on this idea, we have split the category 'recommended' into 'average' and 'good'; if the application seriously hinders the use of the host application (i.e. the Web 2.0 service it is applied on), or it is cumbersome to use, we use the former category, else we put it in the latter. The category 'smoothly usable' is split into 'powerful' and 'ideal' based on a similar reasoning.

The column 'Taxonomy type' in Table I. summarises the results of fitting the taxonomy onto the applications discussed in Section II A. It can be seen that all current solutions that are discussed in this paper are 'dependent', since they are service specific. Our own solution, besides having other merits, is service independent, and is easy to use, as can be seen in the discussion in Section III.

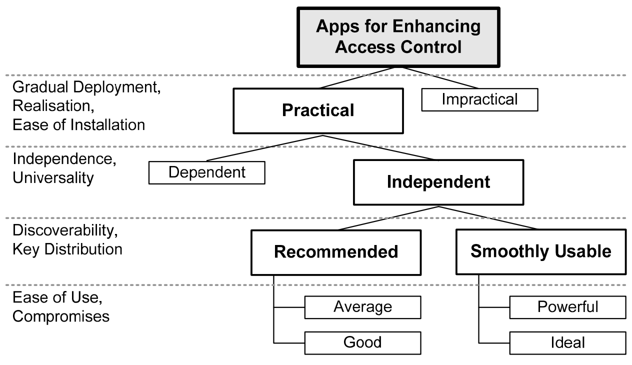

Apps for Enhancing Access Control
Gradual Deployment, Realisation, Ease of Installation
Practical
Impractical
Independence, Universality
Dependent
Independent
Discoverability, Key Distribution
Recommended
Smoothly Usable
Ease of Use, Compromises
Average
Good
Powerful
Ideal

Figure 1. The taxonomy of access control solutions.

TABLE I. CLASSIFICATION OF CURRENT SOLUTIONS

Program	Gradual deployment	Inde-pendence	Universality	Compro-mises	Ease of usage	Ease of installation	Encryption	Discover-ability	Key distribution	Realisation	Taxonomy type
SeGoDoc [6]	N/A	OS	SS	50-99%	Min	NoConf	Sym	Crypto	N/A	Ext	Depend-ent
Persona for Facebook [7]	Possible	OS	SS	50-99%	Min	NoConf	Both	Crypto+ Fake	Auto	Ext	Depend-ent
FaceVPSN [8]	Possible	OS	SS	50-99%	Normal	NoConf	N/A	Fake	N/A	Ext	Depend-ent
StegoWeb	Possible	OS, B, S	GI	50-99%	Normal	NoConf	Sym	Stego+ Crypto	M	B	Good
Ideal	Possible	OS, B, S	UI	100%	Min	NoConf	N/A	Stego+ Crypto	Auto	Ext or B	Ideal

Therefore, it falls into the category 'good'. It must be noted that this result could be improved by implementing the ideas discussed in Section IV.

III. STEGOWEB: A SIMPLE BOOKMARKLET

Our solution called StegoWeb is implemented as a bookmarklet, i.e. as a simple program that can be executed by clicking on a bookmark in the browser. In this section, we describe the model on which it is based, and then provide a description of the actual implementation. The simple usability and the absence of special software requirements were fundamental aspects during the design phase.

A. *Model of StegoWeb*

Our model defines four separate entities:

- Browser: Operations can be controlled by the user with installed bookmarklets.
- Web service: It stores the fake data which serves as a pointer to the location of the real data on a URL shortener service (see below).
- Application storage: It stores JavaScript libraries realising the core functionality of StegoWeb.
- URL shortener service: It stores the real data in an obfuscated form. Arbitrary URL shortener service will do, provided that it supports the aliasing feature, preferably complemented by the ability to delete already registered aliases, e.g. to revoke keys.

We define the following primitives for describing the operation of the algorithms:

- `e(x, k)`: Encrypts `x` with key `k`.
- `d(x, k)`: Decrypts `x` with key `k`.
- `h(x)`: Returns the one-way hash (digest) of `x`.
- `cat(`x_1`, `x_2`, ...)`: Concatenates its arguments.
- `fetch(x)`: Returns the data field of the URL registered under the alias `x` at the URL shortener service.

The inputs of the hiding and revealing algorithms are as follows:

- `KEY`: A key for a symmetric-key encryption algorithm.
- `FAKE_URL`: The address of the website which contains the fake data.
- `REAL_DATA`: An atom of data to be hidden, corresponding to some content on `FAKE_URL`.
- `XPATHS`: The XPath expressions corresponding to the elements in `FAKE_URL` for which `REAL_DATA` is to be hidden. (In our implementation, the user has to provide these by highlighting text on the webpage.)

When performing hiding, three parameters are considered for each piece of fake data: its XPath, the corresponding original data, and the key. The operation is executed in two steps. First, the XPath expressions are hidden, and then the real data. These algorithms can be described as follows:

- XPath hiding:
```
ALIAS := h(cat(KEY, FAKE_URL))
DATA := e(XPATHS, KEY)
```
- Data hiding:
```
ALIAS := h(cat(KEY, FAKE_URL, XPATH))
DATA := e(REAL_DATA, KEY)
```

After each sub-operation, a URL *http://example.com/?data='DATA'* is registered under the alias `ALIAS` at the URL shortener service. Technically, the domain part of the address is arbitrary, but it is wise to choose a popular website to avoid attracting attention.

Real data can be revealed in two steps, too, as follows:

- Revealing XPaths:
```
ALIAS := h(cat(KEY, FAKE_URL))
XPATHS := d(fetch(ALIAS), KEY)
```
- For each entry in `XPATHS`, revealing real data:
```
ALIAS := h(cat(KEY, FAKE_URL, XPATH))
DATA := d(fetch(ALIAS), KEY)
```

The entire process of revealing the real data is depicted on Fig. 2.

B. *Description of the Implementation*

Our implementation is realised as a set of bookmarklets that download a short JavaScript code which realises the aforementioned operations. We have used MD5 as a hash algorithm, AES-CBC as a cipher, and *http://is.gd* as a URL shortener service.

To hide data, the user has to navigate to the website that contains the fake data, click on the selection bookmarklet, and select some text on the page (Fig. 3 (a)). The user is then prompted to enter the corresponding real data. These steps can be repeated as many times (i.e. with as many pieces of text) as desired. When finished, the user has to click on the hiding bookmarklet, which asks the user to type the key to be used for the hiding (Fig. 3 (b)).

To reveal data, the user has to go to the website containing the fake data, and click on the revealing bookmarklet. A dialog box appears where the user has to enter the key. If the right key was provided, each piece of fake data is substituted with the corresponding real data, completely automatically (Fig. 3 (c)).

C. *Analysis*

Our solution slightly deviates from the classical idea of steganography, where information is hidden directly into a cover media, the result of said operation being the stego media. However, our opinion is that classifying this technique as steganographic is appropriate, since the result of its application is that the very fact of the existence of hidden information is hidden from all unauthorised parties. In this – somewhat broader – sense, the cover and stego media can be defined as the combination of the fake website and the set of URLs registered at the shortener service. Indeed, it is impossible to tell if an alias contains hidden information without the key and the fake webpage, provided that the domain part does not give it away. In other words, StegoWeb makes the fake URL hide in the crowd of real URLs. Of course, URLs registered by StegoWeb can only accidentally lead to valid websites, but the case of a user error and that of the use of StegoWeb is not easily distinguishable by the service provider. This is a steganographic quality, even if symmetric-key encryption is used as a core idea of the algorithm. (E.g. TrueCrypt Hidden Volumes [17] are based on hiding cryptograms, too.)

The major advantage of the model is that it can be implemented with free and public web services (just like our proof-of-concept implementation), and it does not require any software to be installed. It is easy to use, and the real data is accessible only on the client side, so the solution is not dependent on the trustworthyness of third party services. Indeed, the URL shortener service and the application server can be easily replaced by other providers, as long as the former complies with the requirements discussed in Section III. A. Furthermore, it is independent both from the browser and the operating system, and therefore it can be used on any platform that can run JavaScript code, which is customary in all modern browsers.

A potential disadvantage is that bookmarklets do not run automatically once the webpage containing the fake data is

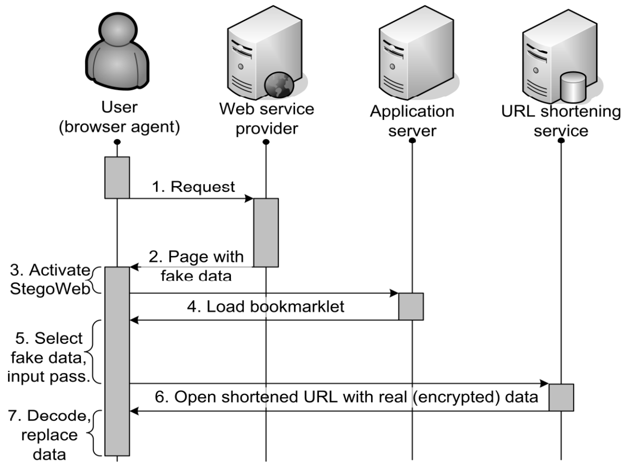
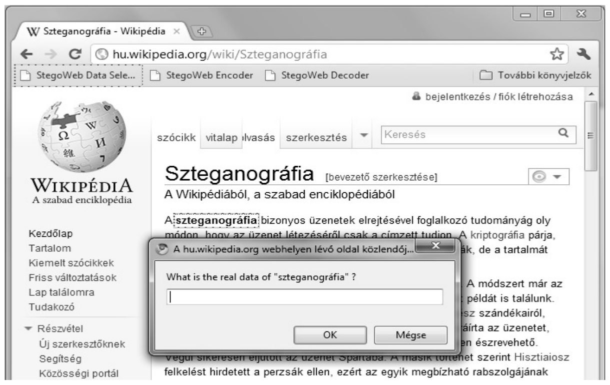
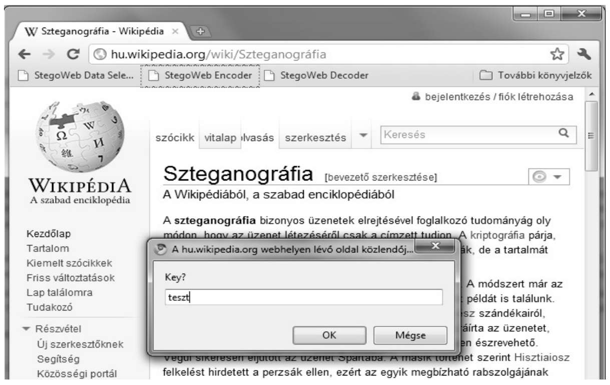


Figure 2. Revealing real data.


Szteganográfia - Wikipédia
hu.wikipedia.org/wiki/Szteganográfia
StegoWeb Data Sele...
StegoWeb Encoder
StegoWeb Decoder
További könyvjelzők
A hu.wikipedia.org webhelyen lévő oldal közlendőj...
OK
Mégse

What is the real data of "szteganográfia" ?

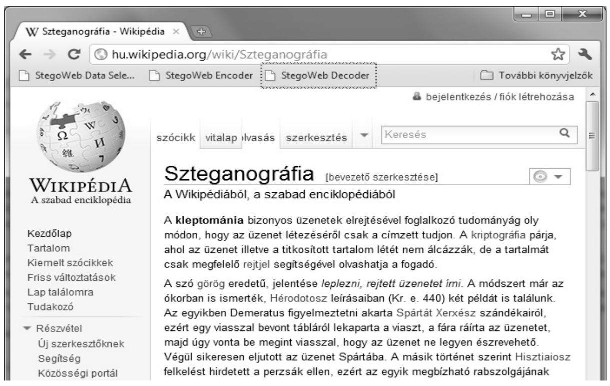
(a)


Szteganográfia - Wikipédia
hu.wikipedia.org/wiki/Szteganográfia
StegoWeb Data Sele...
StegoWeb Encoder
StegoWeb Decoder
További könyvjelzők
A hu.wikipedia.org webhelyen lévő oldal közlendőj...
OK
Mégse

Key?

teszt

(b)



Szteganográfia [bevezető szerkesztése]

A Wikipédiából, a szabad enciklopédiából

A **kleptománia** bizonyos üzenetek elrejtésével foglalkozó tudományág oly módon, hogy az üzenet létezéséről csak a címzett tudjon. A kriptográfia párja, ahol az üzenet illetve a titkosított tartalom létét nem álcázzák, de a tartalmát csak megfelelő rejtjel segítségével olvashatja a fogadó.

A szó görög eredetű, jelentése *leplezni, rejtett üzenetet írni*. A módszert már az ókorban is ismerték, Hérodotosz leírásaiban (Kr. e. 440) két példát is találunk. Az egyikben Demeratus figyelmeztetni akarta Spártát Xerxész szándékairól, ezért egy viasszal bevont tábláról lekaparta a viaszt, a fára ráírta az üzenetet, majd úgy vonta be megint viasszal, hogy az üzenet ne legyen észrevehető. Végül sikeresen eljutott az üzenet Spártába. A másik történet szerint Hisztiaiosz felkelést hirdetett a perzsák ellen, ezért az egyik megbízható rabszolgájának

(c)

Figure 3. Using StegoWeb [18].

loaded, and therefore user interaction is required to reveal the real profile. Furthermore, the password cannot be stored in the browser, so it must be typed again for each execution. (This drawback will be easily eliminated when the penetration of HTML5 local databases will make efficient browser-side storage possible.) Updating the real data is also a problem given that the key has to be renewed every time. Moreover, revocation is possible only if the URL shortener supports deleting aliases, so it is wise to choose such a service that supports this feature.

It must be noted that there is a risk pertaining to the third parties (i.e. the web service, the application storage, and the URL shortener) that our solution relies on. First of all, any of these may stop functioning, e.g. if a server falls victim to a denial-of-service attack. As such, this vulnerability constitutes an availability problem. Furthermore, the scripts hosted on the application storage might be 'poisoned', as is the case when a cracker replaces the StegoWeb libraries with her malicious code. This is a classical system security risk. Fortunately, both vulnerabilities can be eliminated by using multiple application storage and URL shortener services and comparing the information obtained from all sources. Digital signatures may also be considered for additional protection.

To test the implementation, we have verified our solution with several websites. Test runs of the software can be accessed through the webpage *http://stegoweb.pet-portal.eu*.

IV. Key and Identity Management

In this section, we discuss the possible ways of managing keys and identities in our model. If HTML5 databases were widespread enough, these ideas would have made it into the final implementation. Until then, however, it does not make much sense to try to implement automatic key management into our bookmarklet, because an efficient, yet platform-independent means of client-side storage is absent.

A. Key Management

Symmetric-key algorithms use only one key for encryption and decryption (or the encryption key can easily be transformed into the decryption key and vice versa). In our model, data is stored only on the sites of URL shortener services, and, assuming a symmetric-key algorithm as a basis of the implementation, the alias depends on this single key and the URL of the page containing the fake data. Consequently, there is no obstacle to hiding information with different keys for a given profile page. This key can be unique either for each user or for a group, depending on the recipients themselves. The revocation of a key is also simple by deleting the URL alias (if such a feature is implemented on the URL shortener).

Asymmetric-key algorithms use two different keys: a public for encryption and a private for decryption. In StegoWeb, the algorithm for hiding using asymmetric encryption could be similar to the symmetric case. The main difference is that a symmetrical message key is created for each occasion of hiding, and this key is encrypted by the addressees' public keys, who get the cryptograms in private messages. (N.b., if asymmetrical keys were the bases of the encryption phase during hiding, the 'hidden' URLs would be easy to reproduce by everyone, and therefore the goal of steganography would be thwarted, hence the idea of message keys.) The revocation of a key can be performed with the deletion of the alias, in this case, too.

The main problem of asymmetric-key cryptography is the distribution of keys. This obstacle can be easily overcome for certain services; for instance, users of a social network may share this key on their profile page. A script can then collect the keys from their friends' profiles, without explicit communication with them. In other cases, a PGP-like web of trust mechanism [1] can be used for distributing keys, but this involves communication between the participating users.

B. Privacy-Enhancing Identity Management

Here we propose a Privacy-Enhancing Identity Management (PIDM) model for StegoWeb. Our model is based on a social networking service whence public keys of users can be obtained, and where the objective is to conceal profile attributes. A general PIDM scheme offers a hierarchy for managing profiles where attributes of the profiles are inherited from their ancestors in the tree structure or set by the user for the specified profile [3].

This concept can be customised for StegoWeb based on asymmetric and symmetric-key cryptography. If all users publish their public keys on their profile page, sending information (e.g. real profile data) involves looking up the public key, using the hiding algorithm as described in Section III. A, and then notifying the addressee out-of-band, e.g. through the private messaging feature of the social networking website. This works excellently for a single user, but it would require too many aliases at the URL shortener for many addressees. As such, we suggest a hierarchy of groups, each of which has its assigned symmetric key. The users themselves are at the bottom of the hierarchy, and each of them gets the secret keys of the groups that are located between the root of the tree and them when they are added. Then, if one wants to reveal her profile data to a group of users, she does the encryption part of the hiding algorithm with all the intermediate keys, and then registers the URL at the shortening service. Each addressee can then try the revealing algorithm with all key combinations she is aware of. (N.b., all of them got at least one combination of secret keys when they were added to the hierarchy, but they could have got multiple if they were included in multiple groups.)

It can be seen that the number of users in a group defines a trade-off between the number of URLs to be created by the sender and the difficulty of revoking a group key; if one defines groups with high granularity, the sender has to use many different key combinations to publish her real profile information, but 'unfriending' someone is not difficult due to the low number of users in the same group, and conversely, if a group has many users, the initial effort required from the sender is low, but rekeying the group is cumbersome if the fluctuation is high.

It is also interesting to consider the model from the perspective of plausible deniability. One can create multiple hierarchies for the same set of social network friends, and send different profile information on both. Then, if an addressee is forced to surrender her keys, she can hand over those that lead to fake information, and deny the existence of another set of keys, provided that the fake profile information is plausible. This way, the sender's profile data can be effectively protected even from powerful third parties.

V. Conclusion and Future Work

In this paper, we have discussed the threats that profiling poses to users, and categorised the existing solutions that aim to address this problem. The basis of comparing the applications we could find in academic literature was a categorisation scheme based on important attributes that apply to virtually all such applications, and a taxonomy based on these attributes. Then we proposed an own model, and discussed our own implementation thereof. We showed that, through the realisation as a bookmarklet, our solution is not only secure thanks to the underlying steganographic principles, but it is also very easy to use, versatile and as platform independent as possible, the only requirement being a browser that can interpret JavaScript code.

As far as possible improvements are concerned, we believe that the most crucial deficiency is the lack of key and identity management. We have described some alternatives for this in Section IV. When HTML5-based local storage becomes a standard in all modern browsers, this feature can be easily implemented in a completely platform independent way, which is, we argue, paramount for such solutions. Additionally, these features can be enhanced with GUIs created with JavaScript.

Another way of implementing key and identity management would be to realise our solution as a browser extension, so that the application could use the local storage space of the browser itself. This could possibly lead to being bound to a single platform; however, implementing the application as an extension can be advantageous, because the revealing algorithm could be triggered automatically. Such a feature is suitable especially for use with social networking profiles, as these are webpages that have a more or less fixed structure, in contrast to blogs, photo sharing websites, etc. It must be noted that the extension to be implemented is very simple, so it could be easily realised for all modern browsers.

Finally, it would be interesting to verify the implementation in an experiment involving several users. This way, both user experience (e.g. ease of use) and other fundamental properties of the algorithm (e.g. its steganographic security and capacity properties) could be assessed. The results would provide important feedback about what we should improve in the implementation, and a more in-depth comparison to other similar implementations would be possible, too.

Acknowledgement

We would like to thank the High Speed Networks Laboratory for financially supporting our work.

References

[1] B. Schneier, Applied Cryptography, John Wiley & Sons, Inc., United States of America, 1996.

[2] B. Krishnamurthy and C. Wills, "Privacy diffusion on the web: a longitudinal perspective," Proc. of the 18th international conference on World wide web, April 2009, pp. 541–550., doi: 10.1145/1526709.1526782.

[3] G. Gy. Gulyás, R. Schulcz, and S. Imre, "Modeling Role-Based Privacy in Social Networking Services," Proc. of the 2009 Third International Conference on Emerging Security Information, Systems and Technologies, SECURWARE, June 2009, pp. 173–178, doi: 10.1109/SECURWARE.2009.34.

[4] T. Paulik, Á. M. Földes, and G. Gy. Gulyás, "BlogCrypt: Private Content Publishing on the Web," Proc. of the 2010 Fourth International Conference on Emerging Security Information, Systems and Technologies, SECURWARE, July 2010, pp. 123–128, doi: 10.1109/SECURWARE.2010.28.

[5] Y. Zhu, Z. Hu, H. Wang, H. Hu, and G-J. Anh, "A Collaborative Framework for Privacy Protection in Online Social Networks," Cryptology ePrint Archive: Report 2010/491, 2010.

[6] D'Angelo, G., Vitali, F., and Zacchiroli, S, "Content Cloaking: Preserving Privacy with Google Docs and other Web Applications," Proc. of the 25th Annual ACM Symposium on Applied Computing, March 2010, pp. 826–830, doi:10.1145/1774088.1774259.

[7] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," Proc. of the ACM SIGCOMM 2009 conference on Data communication, August 2009, pp. 135–146, doi: 10.1145/1592568.1592585.

[8] M. Conti, A. Hasani, and B. Crispo, "Virtual private social networks," Proc. of the first ACM conference on Data and application security and privacy, February 2011, pp. 39–50, doi: 10.1145/1943513.1943521.

[9] NOYB: Posting Secret Messages on the Web, http://adresearch.mpi-sws.org/noyb.html, retrieved on March 27th, 2011.

[10] FireGPG – Welcome to the official website of FireGPG!, http://getfiregpg.org/s/home, retrieved on February 19th, 2010.

[11] A. Tootoonchian, K. K. Gollu, S. Saroiu, Y. Ganjali, A. Ganjali, and A. Wolman, "Lockr: social access control for Web 2.0.," Proc. of the first workshop on Online social networks, August 2008, pp. 43–48, doi: 10.1145/1397735.1397746.

[12] W. Luo, Q. Xie, and U. Hengartner, "FaceCloak: an architecture for user privacy on social networking sites," Proc. 2009 International Conference on Computational Science and Engineering, IEEE Press, August 2009, pp. 26–33, doi: 10.1109/CSE.2009.387.

[13] M. M. Lucas and N. Borisov, "FlyByNight: mitigating the privacy risks of social networking," Proc. of the 7th ACM workshop on Privacy in the electronic society, October 2008, pp. 1–8, doi:10.1145/1456403.1456405

[14] S. Guha, K. Tang, and P. Francis, "NOYB: privacy in online social networks," Proc. of the first workshop on Online social networks, August 2008., pp. 49–54, doi:10.1145/1397735.1397747.

[15] evercookie - virtually irrevocable persistent cookies, http://samy.pl/evercookie/, retrieved on June 1st, 2011.

[16] Panopticlick, https://panopticlick.eff.org/, retrieved on June 1st, 2011.

[17] TrueCrypt, http://www.truecrypt.org/, retrieved on June 1st, 2011.

[18] StegoWeb, http://stegoweb.pet-portal.eu/index_en.html#usage, retrieved on June 1st, 2011.

End User Computing Environments for Improved Information Security

Pankaj Goyal
MicroMega Inc.
Denver, USA
e-mail: Pgoyal@micro-mega.biz

***Abstract*— Access control does not prevent an authorized "insider" inadvertently or deliberately leaking information to an unauthorized external or internal party. The "insider threat" is one of the greatest threats to enterprise security, and nearly 70% of recently surveyed organizations view Web 2.0 (and by extension cloud computing environments) as a serious data loss risk. The primary focus has been on Data Loss Prevention (DLP) methods to prevent "malicious" data leakage; data leakage includes data loss as well as inadvertent data sharing. In today's highly interconnected world, with a proliferation of camera equipped cell phones, preventing data loss by a determined insider, possibly in collusion with other insiders is impossible. However, if as multiple analyses of data breaches show, the majority of data breaches (as high as 80% of all data breaches) occur from end-user error then the incidence and resulting loss from data breaches can be significantly reduced. This paper presents a method for organizing the end-user computing (EUC) environment to prevent inadvertent data leakage and, thus, improve information security.**

Keywords- information security; data loss prevention; insider threats; end-user computing environment.

I. INTRODUCTION

Ubiquitous untethered anywhere anytime access to vast amounts of information, applications, services and computing resources is fast becoming a reality. Even today, companies and individuals store and provide access to their intellectual property, trade secrets, confidential private information and other assets over a network. Ensuring who gets access to and do what is the subject of both physical and logical security. The major objective of security is to deny, deter, delay and detect unauthorized access.

Crime Prevention Through Environment Design (CPTED) is a holistic multi-disciplinary approach to security. The objective of CPTED is to consider all aspects of the environment in deterring and denying opportunities, including impulsive, for crime. The factors include facilities, Heating, Ventilation, & Air Conditioning (HVAC), utilities (electric, water, waste), Fire, perimeter, physical access control, and intrusion detection.

In both physical and logical access control, the purpose of Identification is to establish the "who" the user is, Authentication is to confirm the veracity of that claim, and Authorization is to verify whether the user has access to the "object" (services, resources, information, documents and other assets). Similar to the physical world, presenting some credential from a trusted credential issuing authority may establish identification. In a highly distributed environment and with no central trusted authority this can be a major challenge; another issue is of preventing fraudulent credentials. Authorization establishes the "rights" of a user to perform some set of actions on an "object" and access control methods aim to protect the "object" from unauthorized access or actions. This is again a challenging problem with distributed and mobile "objects;" in a cloud computing environment (CCE) the objects may relocate to meet performance or other requirements.

In the physical world, perimeter security consists of fences, gates, rooms, doors, dogs and guards; guards, motion detectors and closed circuit television (CCTV) provide surveillance to detect intrusion. In the logical world, network perimeter security through endpoints, intrusion detection systems (IDS), access control and logs all help to delay and detect, and intrusion prevention systems (IPS) to prevent access to achieve security. Safes and secure rooms embedded deep within other rooms, with multiple levels of access control including physical guards, provide asset security. Securing servers and networks is not quite a match for this level of physical deep depth defense; even with hardened infrastructure, the user environment and its usage is not subject to rigorous control.

In the current and emerging ubiquitous computing environments, the "castle defense" mentality of trusting everyone within the organization is flawed. Access control does not prevent an authorized user inadvertently or deliberately leaking an object to an unauthorized party – data breach or data loss; the ubiquitous access to and ease of distribution (or leaking) of objects in a CCE vastly compounds this problem. This "insider threat" is one of the greatest threats to enterprise security. Almost 70% of all organizations view Web 2.0 as a serious concern for data loss prevention (DLP) [8]. Organizations around the world

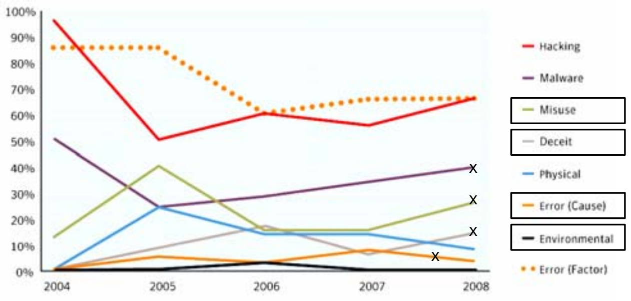


Figure 1. Threat categories over time by percent of breaches [34] (adapted to highlight insider threats).

are facing increased regulatory pressure to secure and safeguard at "rest" or "in flight" digital information; in the US, in addition to federal regulations such as Health Insurance Promotion and Accountability Act (HIPAA)[15], Sarbanes-Oxley[33], Gramm-Leach-Bliley[14], a number of states have imposed their own sometimes more stringent regulations and penalties for breach. However, in today's highly interconnected world preventing data loss by a determined insider, possibly in collusion with other insiders, is impossible; malicious person(s) with appropriate administrative system privileges or even camera equipped cell phones. Although intellectual property theft accounted for less than one percent of all cybercrimes against businesses, it resulted in more than 50 percent of the total monetary loss [31].

An analysis of insider data breaches continues to show that the majority of breaches, as high as 80%, are inadvertent and non-malicious [28, 34]. Fig. 1 shows data breaches by category with the insider breaches boxed; the malicious or "deceit" insider breaches are about half of the insider misuse breach and much less when combined with other breaches such as those due to error and environments. This Data Loss Prevention (DLP) from negligence and non-malicious human error is a major challenge and largely unaddressed, except for the creation of and training on policies. An aspect of DLP that remains unanalyzed is the affect of internal data leaks on performance, such as productivity.

Kulkarni et al [20] addressed the issue of the corporate risks due to uncontrolled storage of data associated with End User Computing Applications, such as spreadsheets, databases etc., with the potential risks posed to the confidentiality, integrity, and availability of this data due to its existence on EUC Applications. It is estimated that around 32% of financial data resides in EUC Applications [27].

The prevention of inadvertent data leakage – the predominant cause of all data leaks – will significantly reduce the incidence and resulting loss from data breaches. This paper presents a method for DLP applicable to the large percentage of such external and internal inadvertent data breaches. The paper examines some well known security mechanisms and shows their inapplicability for inadvertent data breaches.

II. Background and Related Work

In a usage control policy, the object stakeholder defines the allowed accesses to a target object on a target platform. A stakeholder can be the owner of the object, or a provider delegated by the object owner to protect the object. An object can be services, tools, systems, resources, facilities, information, data, messages, or even a credential [37].

A. *Identity Management*

The special issue of Computer magazine [29] on identity management contains a number of papers that deal with the very important issue of digital identity management including interoperable trusted identity [26], federated identity management [7], challenges for federated assurance [23], multifactor identity verification [25] and an identity management framework [22]. The user-centric IDM approach [18] that allows users to control their digital identities and uses identifiers or attributes to define a user.

DLP methods do not require any special identity management and the end-user environment may be subject to multiple different identity management techniques. Access control, DRM and other usage control techniques do not prevent data leakage by authorized users.

B. *Access Control*

Access control technologies enforce or enable enforcement of usage control policies. Access control aims to protect objects from unauthorized access or use by "agents;" agents includes users and tools/systems. Or alternately, grant agents permissions to perform some set of actions on objects. Most popular method is role-based access control (RBAC) that employs the concept of "roles" assigned permissions or "rights" on an "object;" no assignment of individual rights [32]. RBAC is very efficient for large numbers of users, and can deal with a wide range of security policies. Role hierarchies, for example, reflecting some line of authority and responsibility are a common aspect of RBAC models. Role hierarchies support role inheritance a very useful feature when assigning common permissions to large groups. Inheritance also creates role hierarchies where a senior role has more permissions than a junior role; the senior role (r_s) inherits the permissions of the junior role (r_j) and may have additional permissions of its own [35]; the role inheritance satisfies the constraint: (Permissions (r_j) subset of Permissions (r_s)) and (Authorized-Agents (r_s) subset of Authorized-Agents (r_j)).

Attributes-based access control (ABAC) is useful in highly distributed heterogeneous environments [19, 36] and can also be used in conjunction with RBAC. Environment roles can capture the security-relevant context of the environment, as access decisions may depend on the context of the requests [10]; context-based RBAC provide fine-grained access control [16, 21]. Environment roles support the securing of context-aware applications and security policies that make use of environment roles to control access to resources. Constraints in RBAC [35] deal with static and dynamic separation of duty such that either no users are assigned to conflicting roles (static separation of duties), or users cannot be activated for conflicting roles simultaneously (dynamic separation of duty). Rule-based RBAC [1] provides mechanisms to assign roles based on rules defined by the security policy; the rules may establish seniority relationships and, thus, a roles hierarchy. Temporal Role Based Access Control (TRBAC) controls role activation time constraints [4, 17]; limits access to certain times. Temporal and other constraints are a subset of general conditions that control usage; access permitted only when user at specified premise and after a certain event occurring. Other extensions include generalization of roles to include subject, object and environmental roles [24], history [12] and privacy-preserving protocols using zero-knowledge proof-based techniques [3]; and assignment of user rights and permissions for web services based on the strength of the identification mechanism in a context dependent RBAC can be a viable approach for access control in web-based services [35].

In very large organizations or in extended or virtual organizations, centralized RBAC administration is an issue and major challenge. In decentralized domain level RBAC, different administrated domains are independent [2]; the administrator in a domain manages a subset of all the roles and users, and can even define different roles. This however, is a problem for inter-domain role and user rights management.

Most of these access control methods do not address information flow restrictions; for example, usage control of an object released into an end-user environment.

C. Usage Control

Previous work on usage control enabling mechanisms mainly focuses on digital rights management (DRM). Usage control in distributed environments requires the enforcement of security policies on a remote client platform with high assurance and verifiable trust. However, in general, in use DRM mechanisms cannot support enforcement in an EUC Environment for an authorized user. Most importantly, DRM mechanisms are usually proprietary, work best in closed environments and do not interoperate with other DRM techniques. DRM techniques use encryption/decryption and some externally managed trust (key, certificate, rights) or content server; decryption may be restricted to a specific target environment and a particular application; the approaches do not support environment and application heterogeneity. The lack of interoperability (including standards), the need for a centralized external trust server or the need for continuous control hampers adoption in highly distributed environments.

Zhang et al. [37] present general security requirements for usage control and propose a general framework. Their approach requires a hardware-based trusted subsystem that includes a root-of-trust, trust chain, and a policy transformation and enforcement mechanism such that a policy stakeholder can deploy sensitive data and services on the subsystem.

Pretschner et al. [30] present a taxonomy of enforcement mechanisms for usage control and provide an overview of the existing usage control mechanisms. To plan for a future enforcement mechanism the team [30] elicited functional usage control, actions, conditions and obligations requirements from many different organizations and users. In their model, conditions constrain usage restrictions and action requirements, and specify circumstances under which usage restrictions or action conditions apply; conditions are concerned with time, cardinality, events that happened, purpose, and environment. Rare and limited support exists for action requirements and event-defined conditions in current usage control mechanisms.

D. Information Life Cycle and Security Supply Chain

Traditional security aims to protect the IT infrastructure and systems – the perimeter and the structures – that house, manipulate and transport the valuable information assets. Information Lifecycle Management (ILM) manages the flow of information from creation, storage, transport, use, to its deletion. One possible way to identify and address information security issues, over the information lifecycle, is to consider the information security supply chain (ISSC) over the information life cycle [5]. Boyson et al. [5] present a Cyber Supply Chain Assurance Reference Model that draws from supply chain risk and physical security management, and defines key actors, processes, vulnerabilities, and identifies strategic interdependencies at each node of the supply chain.

An ISSC, should make accessible all relevant security-related information to every relevant company in the supply chain to optimize and deliver the most effective security over the entire supply chain and life cycle rather than sub optimize for local the company; lacking this capability, information security is the weakest of the supply chain and life cycle participants. Effective security delivery over the supply chain and life cycle requires both transactional (raw) and analytical security information; analytical information that predicts future possibilities or future impacts of current and past events and decisions is critically important.

III. Insider Threats

The exposure of confidential information is now the single greatest threat to enterprise security. According to one survey, Web email or Web posting (e.g., message board, blog) accounted for 37% of information leaks and that almost 70% of all organizations view Web 2.0 as a serious concern for DLP [8]. Company employees who inadvertently violate data security policies continue to represent a major factor in occurrence of data breaches – some 67% in one analysis and 88 percent in another [28]. Insider threats manifest in a number of ways [29]. Proliferation of information is the natural result of business activity and a productive workforce. As a result of data proliferation, most organizations do not know how much sensitive data exists on their systems and where.

The most common type of data breach occurs when confidential data has been stored, sent or copied by well-meaning insiders [34]. This type of DLP from negligence and non-malicious human error is a major challenge and largely unaddressed, except for creation of and training on policies.

Current mechanisms to thwart insider threats include: (a) content monitoring and filtering solutions; (b) gateway appliances that protect data in motion, such as that in e-mails, instant messages and general Web traffic; and (c) monitoring for suspicious usage patterns. Event and usage logs provide compliance related controls and audits, in addition to the ability to detect threats. Detecting policy violations involving nebulous concepts—such as the transfer of sensitive information and trade secrets—is more difficult and generates numerous false positives [6].

Insider adversaries continue to defeat the current individual and mostly non-integrated protection strategies, and, so a systems-based approach, considers all operational activities including the insider’s characteristics, motives, and capabilities [11]. A controlled double-blind experiment, however, with 50 participants randomly divided into benign and malicious users to detect insiders who misuse their privileges, did not identify any one behavior that distinguishes malicious users from benign ones [6]. The team also reviewed the current state of the art and reviewed publicly available information on malicious insider cases.

IV. End-User Computing Workspaces Model For Information Security

Individually, we participate or operate in a number of social contexts: work, family, professional, etc. (Fig. 2). We possess an identity and perform some role in each of these contexts; our identities and roles may not be identical in these different contexts. In a given context, the identity determines the role but the role identifies a set, possibly null, of individuals (identities); please note that the identity and role cannot determine the context.

In the non-digital world, Identity Management (and analogously role management) is the natural human behavior of generating, managing and choosing roles according to a found social context; where these roles are often mandated by socio-cultural norms and possibly refined by each individual ("role making") [9]; sometimes we choose identities for specific contexts. In a particular context, people choose an appropriate role ("role taking") or perform the role assigned to them, and exhibit a natural capability of resolving any apparent role conflicts in their behavior. They have learned an intuitive understanding of what information to divulge and how to react to information received, in the context of the environment and their own as well as their communication partners' role.

Both in the real world (non-digital world) and the virtual world, these social compacts sometimes wittingly or unwittingly breakdown – "insider threat."

A. *End-User Computing Environment Workspaces*

The EUC environment, commonly referred to as the desktop environment, refers to all of the programs, applications, processes, and data used by an end-user; In this paper the term EUC environment is used as it does not denote any particular end-user hardware. EUC virtualization separates an EUC environment from a physical machine, with the EUC environment (the "virtualized" desktop) residing on a remote central server and running in a virtualized machine. This allows users to access their desktops from any capable device.

An EUC workspace, in the digital world, corresponds to the real world social context. The EUC workspace is a virtualized computing environment characterized by some features, for example, organization, team, role, user identity, applications and operational environment with the objective of, say, managing usage and access control governed by policies and event conditions. An EUC workspace provides a functional EUC environment and encapsulates everything above the operating system kernel – applications, data, and any non-privileged operating system subsystems; each EUC workspace also contains Gateway Services (see below).

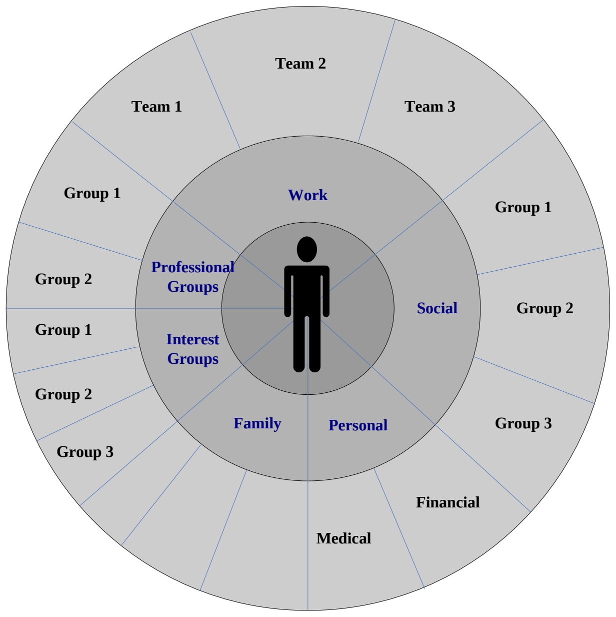


Figure 2. An Individual's social contexts; in the digital world, domains correspond to social contexts.

Please note that these workspaces are quite different from Linux workspaces; every Linux workspace contains the same desktop, the same panels and the same menus, but does not provide workspace isolation.

The EUC environment can, thus, be partitioned into isolated multiple EUC workspaces. Applications within a workspace can interact with each other but are isolated from those in the other workspaces. An application can exist in multiple workspaces but applications in different workspaces are distinct; for example, an email program in two workspaces may have different address books, where the address books contain only entries of authorized recipients, and, thus, only information within the workspace can be emailed to another authorized user. Additionally, an application in different workspaces may be configured differently; for example, some functionality may exist in a workspace but not in another. Thus, the isolation provided by EUC workspaces, the configuration of the applications, authorized user lists, etc. prevents even inadvertent sharing of information with non-authorized personnel. While multiple EUC workspaces may be operating simultaneously, the end-user can only manipulate one workspace at any given time i.e., focus can only be on one EUC workspace. To operate in another workspace, the user has to switch workspaces; switching workspaces does not terminate any running programs, applications, etc.

Secondly, the Gateway Services ensure not only proper logging for audit trail but also ensure that all information (and message exchanges) is automatically encrypted, abides to all policies that govern the workspace, such as information duplicated to the appropriate corporate store.

The EUC workspaces can be arranged in a hierarchy based on security levels (security hierarchy); this is particularly useful for enterprises that typically have an employee participate in multiple teams and where hierarchies are natural. This hierarchical organization of EUC workspaces can be used for asset fragmentation, location distribution and customization and, hence, risk reduction. Promotion of assets, to a higher secure classification level for access in a more secure environment, only after asset security examination – virus scan, integrity checks, and required sign-offs. Demotion of assets, to a lower classification level for wider accessibility, only after asset meets reclassification conditions and organizational policies; thus, a document released for wider audience only after, say, all reviews completed and publication date met. When an asset is successfully reclassified it cannot be accessed from the previous workspace.

Object supply chain (usage, movement, etc.) creates an object use graph (OUG). The OUG for object O_i includes information on all usage (life cycle) and on "parent" objects – objects with superset of content or "simultaneously" accessed objects while O_i created or modified, where "simultaneously" is defined as within some time interval independent of the EUC workspace. The OUG for O_i (or OUG(O_i)) determines the set of objects on which O_i may have a content dependency. Our model is an extension of the dependency graph of [29]. An object O_i is demotable to a target EUC workspace (can be moved to the target workspace – parent hierarchy) if and only if (i) all objects in OUG(O_i) already demoted (usable in target workspace), and (ii) the policy governing demotion of O_i is satisfied (for example, approval by a review committee).

Typically, workspaces exist on an EUC device but there are no technical obstacles for them to exist in a virtualized environment; the virtualized environment will have to support the underlying operating system.

1) *Physical EUC devices*

EUC devices are partitioned into virtual computing environments; can be either hardware level virtualization through the use of hypervisor or supported through the Operating System. An EUC workspace is associated with one and only one virtual computing environment. The current generation of EUC devices is quite capable of handling virtualization.

2) *EUC on Cloud Computing Environment (CCE)*

The same principle applies, except that in a CCE the virtual machines hosting the EUC workspaces may not be on the same server hardware or even the same CCE.

B. *Gateway Services*

Gateway Services control incoming and outgoing messages; only permissible messages are allowed. Ongoing analysis of the gateway logs of all events, messages, etc. and their correlation with other logs (other gateways, access controls, etc.) provides a mechanism for real-time surveillance. Each workspace gateway services serves as Message Intrusion Detection (MIDS) and Message Intrusion Prevention System (MIPS). The gateway/proxy architecture of [13] easily adopts to perform the MIPS and MIDS functions. Analysis of MIPS and MIDS logs, events and messages can take advantage of parallel computation and use of multiple analysis techniques, such as those used to identify behavioral patterns and statistical anomalies.

V. CONCLUSION AND FUTURE WORK

Our individual participation in different social (including work) environments in both our real and digital (virtual) world is a source of data/information loss. In the real world, we are trusted to intuitively safeguard information from unauthorized recipients. This also applies to the digital world. Given easy access and the possibility of errors, for example use of an incorrect email address (of another similarly named individual or a personal instead of official address), the problem of inadvertent, or malicious, data loss is magnified. In many ways, the term and challenges with data protection are being redefined with the advent of virtual and cloud computing environments. The existing problem of insider threats if not properly addressed would be magnified in these environments. Work to date, in the security field has concentrated on preventing unauthorized access to information, whether by internal users or external miscreants. This paper outlines our model for the prevention of non-malicious insider threats.

The model is an EUC focused solution that reduces the risk of inadvertent data leakage. The solution is easy to implement. For example, some capabilities of the model are easily simulated in a multi-user environment by the assignment of different user-ids for each domain; it has also been implemented on a desktop running virtualization software and multiple instances of Linux and even a MS Windows environment. Other capability implementations require changes to, for example, applications; again feasible with open-source applications.

It is impossible to design experiments that simulate actual behaviors in multiple organizations; the data [28, 34], with significant variations among reporting organizations, is of reported breaches and the human error rate ranges between 70 and 90% (Fig. 1). Thus, it is difficult to evaluate the efficacy of the proposed solution until it is adapted by a significant percentage of organizations; then their prior to and post-adaption results can be compared. Though, if the existing data surveys and analysis is correct, then the errors due to inadvertent written disclosure of information shall be minimized; it should be noted that there is no way to prevent verbal disclosure.

A major issue concerns software application licensing and how the major software vendors would treat multiple virtual environments in EUC devices. The solution presented here is also applicable to cloud computing environments. The issue of how Software as a Service (SaaS) providers would treat users that have multiple distinct environments is still open and beyond the scope of this paper; an equitable resolution may increase the appeal of using SaaS in large enterprises.

In the very near future, if not already today, environmental conditions such as "noisy" and "unsecure," and proximity to unauthorized would be easily detectable and, thus, incorporable in a insider threat risk mitigation plan.

REFERENCES

[1] M.A. Al-Kahtani and R. Sandhu. "Induced role hierarchies with attribute-based RBAC," Proceedings of the 8th ACM symposium on Access control models and technologies, 142-148, 2003.

[2] J. Bacon, K. Moody, and W. Yao, "A model of OASIS role-based access control and its support for active security," ACM Trans. Information Systems Security, vol. 5, 492-540, 2002.

[3] E. Bertino, F. Paci, and R. Ferrini, "Privacy-Preserving Digital Identity Management for Cloud Computing," IEEE Computer Society Data Engineering Bulletin, 1–4, Mar. 2009.

[4] E. Bertino, P.A. Bonatti, and E. Ferrari, "TRBAC: A temporal role-based access control," ACM Transactions on Information and System Security, 4(3), 191-223, 2001.

[5] S. Boyson, T. Corsi, and H. Rossman, "Building a Cyber Supply Chain Assurance Reference Model," Science to Solutions Magazine, SAIC, vol. 2, 2009.

[6] D. Caputo, M.A. Maloof, and G.D. Stephens, "Detecting Insider Threat of Trade Secrets," IEEE Security & Privacy, vol. 7, 14-21, 2009.

[7] D.W, Chadwick and G. Inman, "Attribute Aggregation in Federated Identity Management," IEEE Computer, Vol. 42, 33-40, 2009.

[8] C. .A. Christiansen, B. E. Burke, and G. Pintal, "Web Security SaaS: The Next Generation of Web Security," IDC White Paper, 2008.

[9] S. Clauss and M. Kohntopp, "Identity Management And Its Support Of Multilateral Security," Computer Networks, vol. 37, 205-219, 2001. http://dx.doi.org/10.1016/S1389-1286(01)00217-1.

[10] M.J. Covington, W Long, S. Srinivasan, A.K. Dev, M. Ahmad, and G.D. Abowd, "Securing context-aware applications using environment roles," Proceedings of the 6th ACM symposium on Access control models and technologies, 10-20, 2001.

[11] F.A. Duran, S.H. Conrad, G.N. Conrad, D.P. Duggan, and E.B. Held, "Building a System for Insider Security," IEEE Security & Privacy, vol. 7, 30-38, 2009.

[12] G. Edjlali, A. Acharya, and V. Chaudhary, "History-based access control for mobile code," ACM Conference on Computer and Communication Security, 38-48, Nov. 1998.

[13] P. Goyal, "An Interoperability Enabling Framework: Services, Processes and Clouds," 2009 IEEE Asia-Pacific Services Computing Conference (IEEE APSCC), 174-179, 2009.

[14] Gramm-Leach-Bliley Act, "Financial Services Modernization Act," 1999, http://www.gpo.gov/fdsys/pkg/PLAW-106publ102/content-detail.html

[15] Health Insurance Portability and Accountability Act (HIPAA), 1996, http://www.hhs.gov/ocr/privacy/hipaa/administrative/statute/index.html

[16] J. Hu, A. Weaver, A Corradi, R Montanari, and D Tibaldi, "Context-based access control for ubiquitous service provisioning," 28th Annual International Computer Software and Applications Conference (COMPSAC 2004), 444-501, 2004.

[17] J.B. Joshi, R. Bhatti, E. Bertino, and A. Ghafoor, "Access Control Language for Multi-domain Environments," IEEE Internet Computing, 8(6), 40–50, 2004.

[18] M. Ko, G.-J. Ahn, and M. Shehab "Privacy-Enhanced User-Centric Identity Management," IEEE Int'l Conf. Communications, pp. 998–1002, 2009.

[19] D.R. Kuhn, E.J. Coyne, and T.R. Weil, "Adding Attributes to Role-based Access Control," IEEE Computer, Vol. 43, 79-81, June 2010.

[20] A. Kulkarni, E. Williams, and M.R. Grimaila, "Mitigating Security Risks for End User Computing Application (EUCA) Data," IEEE International Conference on Privacy, Security, Risk and Trust, 1171-1176, 2010.

[21] A. Kumar, N. Karnik, and G. Chafle, "Context sensitivity in role-based access control," ACM SIGOPS Operating Systems Review, 36(3), 53-66, 2002.

[22] G. Lopez, O. Canovas, A.F. Gomez-Skarmeta, and J. Girao, "A SWIFT Take on Identity Management," IEEE Computer, Vol. 42, 58-65, 2009.

[23] P. Madsen, H. Itoh, "Challenges to Supporting Federated Assurance," IEEE Computer, Vol. 42, 42-49, 2009.

[24] M. Moyer and M. Ahamad, "Generalized role-based access control," 21st international conference on distributed computing systems (ICDCS'01), 391-398, 2001.

[25] F. Paci, R. Ferrini, A. Musci, K. Steuer Jr., and E. Bertino, "An Interoperable Approach to Multifacotr Identity Verification," IEEE Computer, Vol. 42, 50-57, 2009.

[26] P. Pacyna, A. Rutkowski, A. Sarma, and K. Takahashi, "Trusted Identity for All: Toward Interoperable Trusted Identity Management, Systems," IEEE Computer, Vol. 42, 30-32, 2009.

[27] E. Perry and J. Jinnet, "Spreadsheet Chaos: Impact of Federal Bailout and Other Developments," Prodiance Corporation, April 2009.

[28] Ponemon Institute, Fourth Annual US Cost of Data Breach Study: Benchmark Study of Companies, Ponemon Institute, 2009, http://palisadesystems.com/common/files/Ponemon_CODB_2009.pdf

[29] S. Pramanik, V. Sankaranarayanan, and S. Upadhyaya, "Security Policies to Mitigate Insider Threat in the Document Control Domain," 20th Annual Computer Security Applications Conference (ACSAC'04), 304-313, 2004.

[30] A. Pretschner, M. Hilty, F. Schutz, C. Shaefer, and T. Walter, "Usage Control Enforcement: present and Future," IEEE Security & Privacy, vol. 6, 44-53, 2008.

[31] R.R. Rantala, "Cybercrime against Businesses, 2005," Bureau of Justice Statistics Special Report, Sept. 2008; www.ojp.usdoj.gov/bjs/pub/pdf/cb05.pdf.

[32] R.S. Sandhu, E.J. Coyne, H.L. Feinstein, and C.E. Youman, "Role based access control models," IEEE Computer, 29(2), pp. 38-47, February 1996.

[33] Sarbanes–Oxley Act of 2002, "Public Company Accounting Reform and Investor Protection Act," 2002, http://www.gpo.gov/fdsys/pkg/PLAW-107publ204/content-detail.html

[34] Symantec, "Anatomy of a Data Breach: Why Breaches Happen…and What to Do About It," Symantec, 2009.

[35] R. Wolf, T. Keinz, and M. Schneider, "A Model for Context dependent Access Control for Web-based Services with Role-based Approach," In the Proceedings of the 14th International Workshop on Database and Expert Systems Applications(DEXA'03), 2003, pp.209-214.

[36] E. Yuan and J. Tong, "Attributed Based Access Control (ABAC) for Web Services," IEEE International Conference on Web Services (ICWS'05), pp.561-569, July 2005.

[37] X. Zhang, J-P Seifert, and R. Sandhu, "Security Enforcement Model for Distributed Usage Control," 2008 IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing, 2008, pp. 10-18.

Using Avatars for Improved Authentication with Challenge Questions

Nicholas Micallef
School of Informatics
University of Edinburgh
Edinburgh, UK
nicholasmicallef@gmail.com

Mike Just
School of Engineering & Computing
Glasgow Caledonian University
Glasgow, UK
mike.just@gcu.ac.uk

***Abstract*—We present a novel method for improving the security of challenge question authentication, which traditionally requires a user to answer questions such as *"What is your Mother's Maiden Name?"*. In our method, users create an Avatar representing a fictitious person, and later use the Avatar's information to authenticate themselves. The Avatar Profile consists of basic identifying information (e.g., name, address) as well as personality information (e.g., pets, interests). This info is pseudo-randomly generated from a large corpus of information. For authentication purposes, a small amount of the Avatar Profile information is used to respond to *challenge questions*. In terms of security, the use of information that is not personally associated with the user is intended to thwart *observation* attacks such as, for example, knowing the user's mother's maiden name. In terms of usability, our design establishes a bond between the user and their Avatar using graphical images and periodic associations by, for example, presenting an image of the Avatar at each login. This *nurturing* of the bond between a user and their Avatar leverages known psychological phenomena. At the same time it also provides a novel adaptation to security of the emotional investments that users exhibit in *virtual worlds* and *massively multi-user online graphical environments*. In this paper, we describe our work-in-progress towards an Avatar Authentication design, partially guided by an initial pilot experiment. Our initial results are promising and point to a possible future for the use of avatars for authentication.**

***Keywords*-authentication; avatar; security; usability.**

I. Introduction

A secure and usable authentication system is of critical importance for applications nowadays. A secure system prevents impersonation, which otherwise could lead to monetary loss, embarrassment, inconvenience, and other problems. A usable system not only encourages more secure behaviour, but also results in a more enjoyable user experience and can increase user enrolment.

Password authentication is the most ubiquitous authentication method used today, yet its security and usability weaknesses have been known for many years [1][2][3]. It is also important to note that with increases in attacker capabilities and an ever-increasing quantity of accounts that users must manage, these problems are further exacerbated. Alternatives such as biometrics and smartcards continue to present usability issues and deployment challenges [4].

Recently, significant research effort has been focused on alternatives to *text* passwords. Numerous graphical password systems have been developed and tested, and recent research suggests good potential, and a path toward more consistent evaluation [5]. Challenge question authentication is as ubiquitous as passwords, and is often used as a form of secondary authentication in case a user forgets their original password. Their use of personal information and memories, as opposed to *memorized* passwords, has been the main focus of the latest research [6][7][8][9]. Unfortunately, in their basic implementation, the answers to challenge questions are most vulnerable to *observation attacks* in which the information and memories used by a user to authenticate, are also known to (or easily determined by) attackers.

Our approach to authentication uses an *Authentication Avatar* which represents the identity, including personality, of a fictitious person that is almost randomly generated from minimal user input. An Avatar Profile (AP) contains information about the avatar, and a subset of the AP information is used by the user to respond to challenge questions such as *"What is your Avatar's pet's name?"*, or *"What was your Avatar's secondary school?"*. In this way security might be improved since, unlike the user's own information, the avatar information is not as easily determined by an attacker. In terms of usability, since such fictitious information is likely to be more challenging for a user to recall (than their own, personal information), our design uses techniques such as repeated exposure and graphical imagery whereby, for example, a user might be exposed to an image of their avatar at every login in order to improve the memory association. Such images can be associated with the avatar itself, and also with elements of the AP, e.g., a picture of the Avatar's pet. From our early designs, it appears that such methods can be seamlessly integrated into the authentication process. In this way, as with avatars used in *virtual worlds* and *massively multi-user online graphical environments*, our authentication avatar design attempts to build upon the degree of emotional investment that users exhibit with avatars.

In Section II we describe several factors influencing our design. In Section III we describe our current, prototype implementation and highlight some possible design variations. In Section IV we describe some plans for our final design, and discuss our plans for measuring security and usability. Section V provides some concluding remarks and direction for future work.

II. Motivation and Design Considerations

In designing an avatar for authentication purposes, we were motivated by existing work in secure and usable authentication, and also reflective of avatars designed for other purposes, such as virtual worlds. Our goal was to build a structure containing information that could be used by a user to authenticate in such a way that the structure holds some meaning for the user, but not for attackers. For the authentication protocol, we chose to base our solution upon challenge question authentication, and note that avatars may similarly prove useful for other security methods.

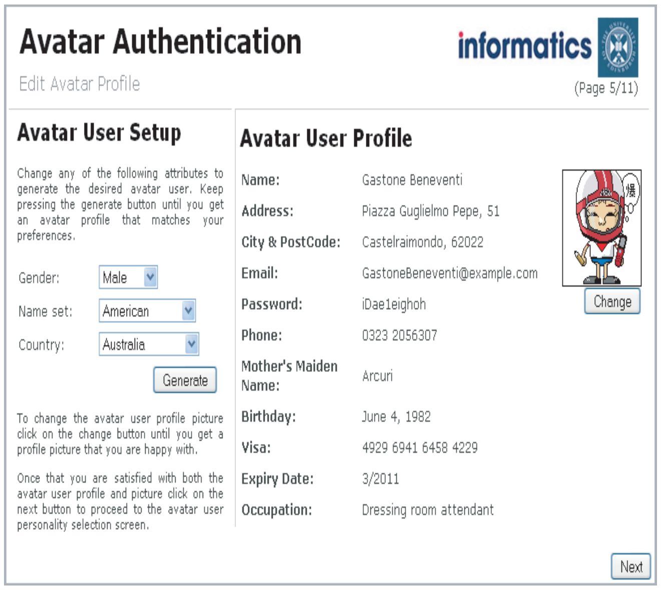
There are two main issues to securing challenge question authentication. Firstly, answers with inherently small, or non-uniform answer spaces should be avoided [7][8]. And while most questions share this risk, it can be partially mitigated with appropriate question selection, and the enforcement of a requirement to use multiple authentication questions. Secondly, the answers to challenge questions are often *observable*, so that a determined attacker can observe or recover answers to challenge questions with relative ease [6][9]. For this reason our design introduces a proxy for the answers to the challenge questions: an Avatar.

In terms of usability, the same research indicates that users struggle in recalling their own answers to challenge questions, despite the fact that the answers are *already known* to the user (and shouldn't require additional memorization). There are several potential reasons for this, including a user's changing or conflicting memories. In this sense, it may be that current designs have relied too heavily upon users to accurately and specifically recall one of their various memories, especially when it is likely that the memory originated in some other other context. The potential of authenticating with an avatar is to create new memories, but in such a way that users establish a close, personal bond with this information. The idea of creating this information in the context of their authentication application is to make the information more memorable when re-used in that same context to later authenticate. For our solution we attempt to build this relationship through nurturing, where an electronic pet is used as a daemon [11]. In our solution, the user is consistently exposed to their avatar through a representative image, ideally forming an association that improves the recall of the answers to the related challenge questions. We also apply some additional memory techniques at registration such as story writing and repetition [12].

In terms of the data upon which the AP is constructed, we collected a large corpus of profile and personality information based upon existing sources, often used for different purposes, such as registering for aggregate news sites [13][14][15]. Whereas such sites are often used for "one time" registrations (e.g., when information is collected for marketing purposes), it is a portion of our AP information that is used for repeated and consistent authentication.

III. An Authentication Avatar Implementation

From these considerations, we can begin to design, implement and evaluate an avatar authentication system. Below, we describe some additional detail involved in this design, and include a couple of screen-shots from our current prototype implementation. The design focuses on the creation of an *Avatar Profile* during user registration. The AP includes information that would likewise be associated with a real person, e.g., name, family. From this information, the user will then choose three challenge questions where the answers relate to the information contained in the AP. The AP which builds upon elements used elsewhere for fake name generation [14], consists of two parts: the Avatar User Profile (AUP), and the Avatar Personality (APY). We now describe the process for building this profile in more detail.



Avatar Authentication

Edit Avatar Profile

(Page 5/11)

Avatar User Setup

Change any of the following attributes to generate the desired avatar user. Keep pressing the generate button until you get an avatar profile that matches your preferences.

Gender:	Male
Name set:	American
Country:	Australia

To change the avatar user profile picture click on the change button until you get a profile picture that you are happy with.

Once that you are satisfied with both the avatar user profile and picture click on the next button to proceed to the avatar user personality selection screen.

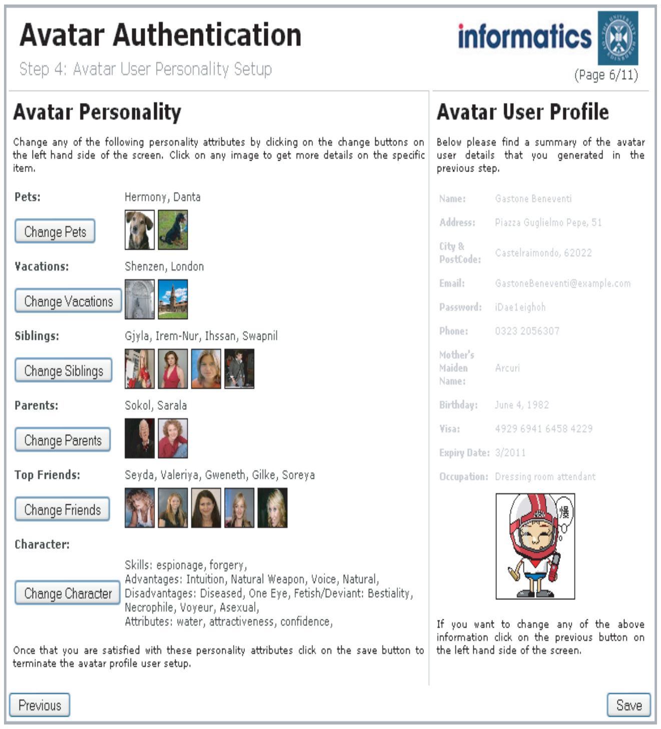
Avatar User Profile

Name:	Gastone Beneventi
Address:	Piazza Guglielmo Pepe, 51
City & PostCode:	Castelraimondo, 62022
Email:	GastoneBeneventi@example.com
Password:	iDae1eighoh
Phone:	0323 2056307
Mother's Maiden Name:	Arcuri
Birthday:	June 4, 1982
Visa:	4929 6941 6458 4229
Expiry Date:	3/2011
Occupation:	Dressing room attendant

Figure 1. Avatar User Profile Setup

Figure 1 shows the first stage of user registration. The user *seeds* the AUP with information about the avatar, selected from three drop-down lists: the avatar's *gender*, *name set*, and *country of birth*. In our current implementation, the gender choice is either male or female, the name set is one of 18 different cultures (e.g., American, Arabic, Hispanic), and the country is one of 19 countries, where details for the latter two were taken from Fake Name Generator [13]. The generated AUP information includes a name, address, city & postcode, email address, password, phone number, mother's maiden name, birthday, credit card number & expiry date, and occupation for the Avatar, in addition to a random image.

To continue, the user populates the *Avatar Personality (APY)* as depicted in Figure 2, consisting of information about pets, vacations taken, family (siblings & parents), friends, as well as various character traits. In our current implementation, rather than seeding from user input, the APY information is initially chosen randomly, and a user is thereafter able to toggle the selections in each category. Also notice that for many of the elements of the APY, an image is associated with the Avatar information.



Avatar Authentication

Step 4: Avatar User Personality Setup

(Page 6/11)

Avatar Personality

Change any of the following personality attributes by clicking on the change buttons on the left hand side of the screen. Click on any image to get more details on the specific item.

Pets:	Hermony, Danta
Vacations:	Shenzen, London
Siblings:	Gjyla, Irem-Nur, Ihssan, Swapnil
Parents:	Sokol, Sarala
Top Friends:	Seyda, Valeriya, Gweneth, Gilke, Soreya
Character:	Skills: espionage, forgery, Advantages: Intuition, Natural Weapon, Voice, Natural, Disadvantages: Diseased, One Eye, Fetish/Deviant: Bestiality, Necrophile, Voyeur, Asexual, Attributes: water, attractiveness, confidence,

Once that you are satisfied with these personality attributes click on the save button to terminate the avatar profile user setup.

Avatar User Profile

Below please find a summary of the avatar user details that you generated in the previous step.

Name:	Gastone Beneventi
Address:	Piazza Guglielmo Pepe, 51
City & PostCode:	Castelraimondo, 62022
Email:	GastoneBeneventi@example.com
Password:	iDae1eighoh
Phone:	0323 2056307
Mother's Maiden Name:	Arcuri
Birthday:	June 4, 1982
Visa:	4929 6941 6458 4229
Expiry Date:	3/2011
Occupation:	Dressing room attendant

If you want to change any of the above information click on the previous button on the left hand side of the screen.

Figure 2. Avatar Personality (APY) Setup

For the AUP and APY, we chose elements with large answer spaces, and random selection should produce relatively flat distributions (to be verified as part of our experiments).

Once the Avatar Profile (including the AUP and APY) has been populated, there are at least two options for selecting the challenge questions. Firstly, the user could be presented with a list of candidate challenge questions and be asked to choose three questions from the list, providing the answers that correspond to the information from the AP.[1] This is the option that we have currently implemented for our prototype design. Alternatively, the user could select three categories of information from the AP, and then challenge questions associated with this information could be presented to the user. Based upon early results from our pilot experiment (see Section IV) we plan to implement the second option in our next prototype version as it should allow the user to more clearly focus on elements of the AP that are most relevant (and ideally, memorable) to them.

[1]The answer information could be automatically populated, but requiring the user to enter the information would help to improve answer retention.

As noted above, our current implementation associates images with most of the elements of the AP. For example, our sample Avatar is represented by a small character with a red and white helmet, and family members such as pets and siblings are also represented with images. These images can serve as *cues* to the AP information, and can help to prompt the user for their answer. For example, if the user registers the challenge question *"What is the name of my sister?"*, an image of the Avatar's sister would be associated with the answer. When later authenticating, the user would be presented with the question *and the image cue*, and be asked to provide the answer.

If challenge questions are used as a user's primary form of authentication, then the user would regularly be presented with the images at login, reinforcing the answers at each login as a form of nurturing [11]. If, as is more typical today, the challenge questions are less regularly used for situations such as recovery due to a forgotten password, then the images could still be regularly presented to the user as a way of reinforcing the memory of the answers. For example, at password login the user could be shown the image of their Avatar as a reminder of their Avatar's character. This regular engagement with the avatar images will (hopefully) encourage improved recall of the avatar information, and has a side benefit of contributing to the authentication of the server to the user. We plan to validate these hypotheses as part of an experiment on our final design.

IV. Fine Tuning our Implementation

To inform our final design we conducted a pilot experiment of our initial prototype with approximately 100 staff and students from the University of Edinburgh. Participants configured an Avatar Profile, registered a set of three challenge questions and corresponding answers, and then returned after two weeks to attempt to authenticate with the answers to the challenge questions.

In terms of security, our design supports a random population of the AP information suggesting that for a challenge question with n possible answers, an attacker would have a guessing probability of $1/n$. However, the answers aren't completely random due to the impact of user choice in their selection. For example, in the pilot experiment, while almost 75% of participants used the Avatar Profile information to populate answers to their challenge questions, 25% did not (and possibly used their own personal information). For this reason, we are implementing the aforementioned modification in which users will be presented challenge questions based upon their identification of preferred information from the AP. In addition, we need to determine whether users exhibit a bias in choosing the Name Set or Country for their Avatar User Profile, or similar bias with the Avatar Personality (APY), e.g., users might toggle to choose more familiar pet or sibling names. Such effects weren't noticed in our pilot experiment, but need to be confirmed with a larger set of diverse users.

In terms of usability, even though we had not yet implemented the nurturing features of our system, more than one-third of the challenge questions were answered correctly by our participants – a surprisingly positive result in that some users were able to recall newly memorized information for a fictitious person, giving us further hope when we implement our methods to increase this bond. We expect this number to increase significantly upon implementation of our nurturing features and will compare the recall results of users with baseline results for existing challenge question systems [6][7].

V. Conclusion and Future Work

Avatar authentication is a new way to view information-based authentication which utilizes fake personas (an *Avatar*) that can be created by users in order to authenticate themselves. The use of an Avatar is intended to thwart attackers who are otherwise able to obtain personal information about a user. We describe how *nurturing* of the bond between the user and the Avatar leverages known psychological phenomena and provides a novel adaptation to security of the emotional investments that users exhibit in *virtual worlds* and *massively multi-user online graphical environments* in order to better recall information associated with the Avatar. We described our initial prototype design and some plans for improvement following a pilot experiment with 100 participants in which participants showed a surprising ability to recall information associated with their Avatar (despite the fact that our initial prototype had not yet included key nurturing features).

Looking to future work, there may be other ways to set-up a fake persona for authentication purposes, for example, by randomly gathering information from disparate users on a social network, or even by gathering information related to digital objects [16]. Also, there are likely different ways to build the bond between the user and the Avatar, perhaps more fully leveraging components of a virtual world.

VI. Acknowledgments

Thanks to Bill Cheswick for initially suggesting the use of *false information* for authentication.

References

[1] E. H. Spafford, "Observing reusable password choices," in *In Proceedings of the 3rd Security Symposium. Usenix*, pp. 299–312.

[2] D. V. Klein, "Foiling the cracker: A survey of, and improvements to, password security," in *Proceedings of the 2nd USENIX UNIX Security Workshop*.

[3] J. Yan, A. Blackwell, R. Anderson, and A. Grant, "Password memorability and security: Empirical results," *IEEE Security and Privacy*, vol. 2, no. 5, pp. 25–31, 2004.

[4] L. Ballard, S. Kamara, and M. K. Reiter, "The practical subtleties of biometric key generation," in *SS'08: Proceedings of the 17th conference on Security symposium*. Berkeley, CA, USA: USENIX Association, 2008, pp. 61–74.

[5] R. Biddle, S. Chiasson, and van Oorschot P.C., "Graphical Passwords: Learning from the First Twelve Years," *ACM Computing Surveys*, 09 2009.

[6] S. Schechter, A. J. B. Brush, and S. Egelman, "It's no secret. measuring the security and reliability of authentication via 'secret' questions," in *SP '09: Proceedings of the 2009 30th IEEE Symposium on Security and Privacy*. Washington, DC, USA: IEEE Computer Society, 2009, pp. 375–390.

[7] M. Just and D. Aspinall, "Personal choice and challenge questions: a security and usability assessment," in *SOUPS '09: Proceedings of the 5th Symposium on Usable Privacy and Security*. New York, NY, USA: ACM, 2009, pp. 1–11.

[8] J. Bonneau, M. Just, and G. Matthews, "What's in a name? evaluating statistical attacks on personal knowledge questions," in *Proceedings of Financial Cryptography 2010*. N/A: SpringerLink, 2010.

[9] A. Rabkin, "Personal knowledge questions for fallback authentication: security questions in the era of facebook," in *SOUPS '08: Proceedings of the 4th symposium on Usable privacy and security*. New York, NY, USA: ACM, 2008, pp. 13–23.

[10] N. Yee, "The psychology of massively multi-user online role-playing games: Motivations, emotional investment, relationships and problematic usage," in *Avatars at Work and Play*, ser. Computer Supported Cooperative Work, R. Schroeder and A.-S. Axelsson, Eds. Springer Netherlands, vol. 34, pp. 187–207.

[11] P. Briggs and P. Olivier, "Biometric daemons: authentication via electronic pets," in *UPSEC'08: Proceedings of the 1st Conference on Usability, Psychology, and Security*. Berkeley, CA, USA: USENIX Association, 2008, pp. 1–5.

[12] A. D. Baddeley, *Essentials of Human Memory*. Psychology Press, 1999.

[13] CorbanWorksLLC, "Generate a random name, last accessed: 2010-08-18, http://www.fakenamegenerator.com/gen-female-sw-us.php," 2006. [Online]. Available: http://www.fakenamegenerator.com/gen-female-sw-us.php

[14] WikiHow, "Create a fake real person, last accessed: 2010-08-18, http://www.wikihow.com/create-a-%22fake-real-person%22," 2009. [Online]. Available: http://www.wikihow.com/Create-a-%22Fake-Real-Person%22

[15] ——, "How to make an imaginary friend, last accessed: 2010-08-18, http://www.wikihow.com/make-an-imaginary-friend," 2009. [Online]. Available: http://www.wikihow.com/Make-an-Imaginary-Friend

[16] R. Biddle, M. Mannan, van Oorschot P.C., and T. Whalen, "User Study, Analysis, and Usable Security of Passwords Based on Digital Objects," School of Computer Science, Carleton University., Tech. Rep., 10 2010.

Application of Scenario-driven Role Engineering to the *MinaBASE* Process Knowledge Database

Daniel Kimmig*, Andreas Schmidt†*, Klaus Bittner*, and Markus Dickerhof*
**Institute for Applied Computer Science*
Karlsruhe Institute of Technology
Karlsruhe, Germany
E-mail: {daniel.kimmig, andreas.schmidt, klaus.bittner, markus.dickerhof}@kit.edu
†*Department of Informatics and Business Information Systems,*
University of Applied Sciences, Karlsruhe
Karlsruhe, Germany
E-mail: andreas.schmidt@hs-karlsruhe.de

***Abstract*—Collaborative systems, which are often used in short-term virtual enterprises or long-term cooperation networks, often contain informations about the manufacturing and fabrication competences of the technology partners which should only be made available to a very restricted group of persons for example to support feasibility studies in the context of actual customer requests. This is a new important feature to be supported in nowadays knowledge management systems. Hence, the goal of this paper is so to present a methodology for implementing an access system that supports the definition of fine granular access rights for cooperative process knowledge management systems, that could protect such sensible information. In this paper we present an adaption of the scenario-driven role engineering method to the special needs in a collaborative process knowledge management system with fine granular access requirements. Beside the adaption of the scenario-driven role engineering method to such a system, the adapted method will be concretely applied to the process knowledge management system *MinaBASE*, which was developed in our institute. To complete, an implementation will be shown with the help of the inversion of control framework "Spring Security". Here static aspects as well as dynamic aspects of security will be presented. The paper shows in a detailed manner the usability of the scenario-driven role engineering method for applications in the field of collaborative knowledge management.**

***Keywords*-Data confidentiality and integrity; knowledge management; RBAC;**

I. Introduction

Both corporate groups as well as small and medium-sized enterprises (SME) are experiencing increasing competition and shorter product lifecycles. The resulting necessity of shortening the product development process also has to be mastered by enterprises in the field of microsystems technologies (MST) that are characterized by a high interdisciplinarity and complex, multi-stage, and hardly standardized fabrication processes. Frequently, every product is produced by an individually tailored fabrication process [1]. While larger MST enterprises still manage a wide spectrum of technologies, SME rather tend to offer solutions in a high specialized area. To offer more complex solutions, they establish technical partnerships with other SME. These may have the form of short-term virtual enterprises or long-term cooperation networks. To support such organization forms in the field of MST, the *MinaBASE* process knowledge database was developed by the Institute for Applied Computer Science of Karlsruhe Institute of Technology. By means of this database, the manufacturing and fabrication competences of the technology partners are made available to a central coordinator who then assesses technical and economic feasibility. This information, however, includes internal know-how, the confidentiality, secrecy, and integrity which is of crucial importance to the company's existence. Acceptance of *MinaBASE* therefore does not only depend on whether it meets functional requirements, but also on aspects like safety and access protection. To prevent an undesired disclosure of company secrets of a technology partner, access shall be controlled by the established role-based access control (RBAC) [2]. Here, authorizations are not assigned directly to subjects, i.e., the users of a system, but to abstract roles to which the users are assigned afterwards. In this way, the frequently error-prone maintenance expenditure is reduced and safety is increased. However, this requires the definition of an adequate role concept. Information systems often use standard models with system-wide administrators, owners of information objects, and guests having restricted read access. It is not considered in which business processes the system is used and which particular requirements result in terms of confidentiality and data integrity. For *MinaBASE*, it should be possible to provide a partner with insight in the product-related properties of a microsystem during the sales process, but not to disclose the configuration of machine parameters to produce these properties. Such an application exceeds the modeling capacities of the standard role concept described, as external partners are not the owners of this information. This results in a high expenditure to maintain the finely detailed authorizations. This problem will be solved by a role concept individually adapted to *MinaBASE* . This role concept is based on a systematic approach, scenario-driven role engineering. The main contribution of the paper is twofold: First, the adaption of the scenario-driven role engineering method will be adapted to the requirements of collaborative knowledge management systems. And second, the suitability of existing access frameworks to implement the adapted method will be shown by means of the IoC-framework spring security.

The paper will be structured as follows: In the next section, the *MinaBASE* process knowledge database will be presented. Then, the scenario-driven role engineering approach [3] and the adaptations to the background and objective of *MinaBASE* will be outlined. Afterwards, the model will be applied and a role concept for RBAC ensuring data integrity and confidentiality for *MinaBASE* will be derived. The final section describes the implementation of this role concept within an IoC-Framework by demonstrating how Spring Security and technolgies such as AOP can be used to fulfill static and dynamic security requirements.

II. *MinaBASE* Process Knowledge Database

The knowledge required to produce values added is no public property, but a company resource that has to be administrated efficiently in order to ensure economic success. To support this process, knowledge management systems have been established [4]. In process-oriented knowledge management [5], these methods are applied to highly knowledge-intensive fabrication processes, as those used in MST. The *MinaBASE* process knowledge database was developed within the framework of the MicroWebFab joint project funded by the BMBF [5]. It was used by the technology partners for the structured storage of technical fabrication parameters of the methods and materials used in MST and of the partner-specific technical competences. In *MinaBASE* , the smallest information entity is the so-called technical aspect (TA). It is used to model materials, machines, and fabrication technologies [6]. By means of generalization hierarchies, TAs are arranged in taxonomies. The number and contents of taxonomy trees can be specified and modified during runtime, such that a flexible structure meeting MST requirements can be defined for the storage of fabrication know-how. TAs may be assigned properties referred to as technical parameters (TP). A TP is a string of characters, integers, or floating-point numbers in a certain unit and references an attribute, e.g., the density. The TP of a TA are inherited by lower partial hierarchies of the hierarchy tree in analogy to the object-oriented approach. Hence, no error-prone multiple input is required. In addition, lower hierarchy levels can further refine the inherited TP by specifying general value ranges. Typing of the TP places them in a certain context, such that a TP refers to a product or its fabrication and, hence, is either product-specific or fabrication-specific. Product-specific TP describe the properties of a microsystem, such as the depth of a groove reached by the fabrication process of milling. Fabrication-specific TP refer to the machine configuration needed to produce a specific product property. For modeling the capabilities of a technology partner, competences [7] are considered to be a set of various TA from disjunct hierarchy trees, which is illustrated in Figure 1. This figure schematically represents the competence “injection molding of a rib with polystyrene using the Arburg Allrounder machine” together with some TP. From the hierarchy trees of process, machine, material, and geometry element, the TAs are selected. These TAs are characterized by own TP, such as the injection pressure of the injection molding process. The combination of these TAs results in the competence that is reflected by other TP, such as the edge quality and surface roughness. Consequently, a competence is a type of view of a certain combination of TAs with properties in the form of TP that apply to this combination only, i.e., that characterize the competence in more detail. TAs can be used in several competences. They represent reusable, encapsulated, smallest information entities. An extension of the *MinaBASE* concept has been developed in order to reuse these information entities to allow process modeling of manufacturing sequences based on semantic technologies [8].

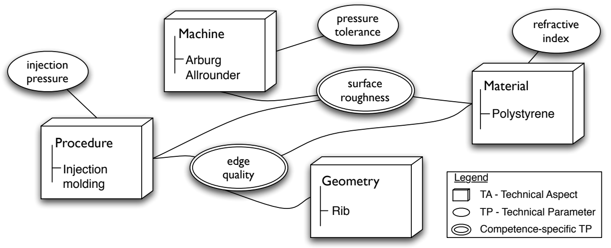


Figure 1. Schematic representation of a *MinaBASE* competence [8]

III. Scenario-driven Role Engineering

The term of role engineering (RE) in the context of RBAC means the design and specification of roles, authorizations, secondary conditions, and restrictions as well as of a hierarchic role model [9]. RE is used to create a concrete model for RBAC-based access control. In [3], Scenario-driven role engineering (SDRE) is defined as an approach based on scenarios, such as sequences of actions and events from the user’s perspectives. This sequence in a scenario can be subdivided into subscenarios and steps. Scenarios are embedded in a task, i.e., a problem or a work area, which links the scenarios of a system with its users. The users mostly apply a system to fulfill a task of their work profile or their job description. This structurization into various levels serves to break down a job description of a user into atomic steps, each of which may be associated with an authorization to access a resource. For various types

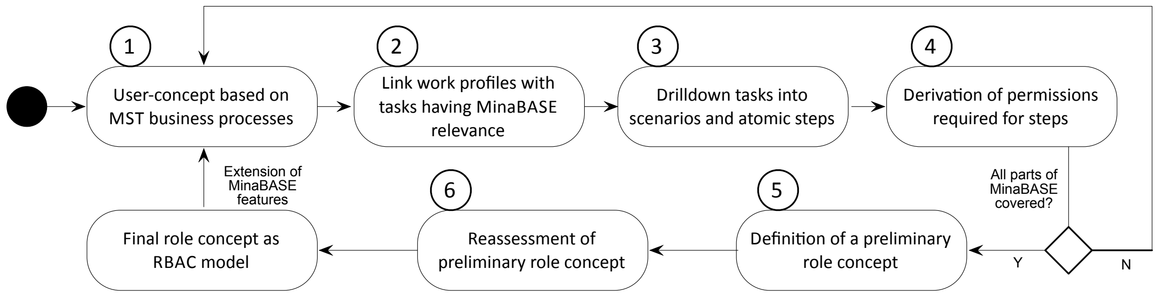

1
User-concept based on MST business processes
2
Link work profiles with tasks having MinaBASE relevance
3
Drilldown tasks into scenarios and atomic steps
4
Derivation of permissions required for steps
All parts of MinaBASE covered?
Y
N
5
Definition of a preliminary role concept
6
Reassessment of preliminary role concept
Final role concept as RBAC model
Extension of MinaBASE features

Figure 2. Scenario-driven role engineering according to [3] with adaptions

of users, the minimum amount of authorizations required for the execution of the tasks can be derived. In this way, the principle of least privilege [10] is implemented. For documentation, various models are generated by the SDRE approach, which are interlinked in terms of contents and used for the derivation of the role concept. The *scenario model* describes all scenarios and steps, *task definitions* serve to structure scenarios, the *work profile* summarizes tasks for job descriptions. The *permission catalog* lists the individual permissions or authorizations. It may be complemented by a constraint catalog of special limitations [3]. While the permission catalog is focused on static assignments of authorizations to specific resources, constraints describe dynamic conditions, which are evaluated at runtime. Hereinafter, the SDRE process will be described in general. First, the use scenarios of the system are compiled and their actions and events are documented. Then, subscenarios and steps are defined and the authorizations required for them are included in the permission catalog or special limitations are listed in the constraint catalog. When this step is completed for all scenarios, similar scenarios are generalized. Very complex scenarios are divided into smaller parts which are then included in the scenario model. On this basis, tasks are formed by grouping scenarios. These tasks are then classified into various work profiles. This results in a preliminary role concept and minimum authorizations can be assigned to the individual activities. As a rule, this preliminary model contains duplicates of roles with identical authorizations, which then have to be fused in a last step. This yields the RBAC model as a role concept. The SDRE process represents a systematic approach to RE. It was applied to information systems for the health care sector by the technical committee of HL7 already [11]. Due to this practical test, SDRE in principle may be applied to *MinaBASE* . However, certain adaptations are required, because the background and objective of *MinaBASE* differ from those of the HL7 systems. The paramount objective of *MinaBASE* is the support of knowledge-intensive business processes of MST enterprises by a structurization of the knowledge required for the execution of these business processes. A criterion for the acceptance of knowledge management is its integration in workflows of the users and an efficient and complete coverage of information needs [12]. As such the SDRE approach is to be applied to the use of *MinaBASE* in business processes of MST enterprises and cooperation networks. The model given in [3] is therefore subjected to the following adaptations:

- In the standard SDRE methodology, scenarios for a system are the main input, to which required authorizations are allocated. Afterwards these scenarios are generalized and assigned to tasks and work-profiles which create a preliminary role concept, that needs to be revised afterwards.
  For *MinaBASE* however an alternative input is more persuasive. Instead of starting with the scenarios of the system, work areas within the business processes of an MST-company are examined, whether they include tasks in which *MinaBASE* can be used to increase added value. To these tasks scenarios will be assigned in order to obtain the information, which resources are required for fulfilling them and what authorizations are needed. Based on this information, a preliminary role concept can be derived analogous to the standard model due to the minimal set of authorizations for each role. By these adapations, a switch of input variables to the methodology - the basic principle of SDRE is preserved, while better results for the creation of the role concept are expected, because of the adapted methodology being closer to the business processes of a MST-company.
- The scenarios to be formulated are not based on consequences of actions and events, but will also contain all definable steps. Although these do not occur in sequential order, they can be characterized by a certain access authorization.
- For reasons of clarity, special limitations extending beyond the static allocation of authorizations are included directly in the permission catalog and not in the

constraint catalog, such that both models fuse.

The adapted process is illustrated in Figure 2. It comprises six steps, the execution of which shall be described in more detail in the following section.

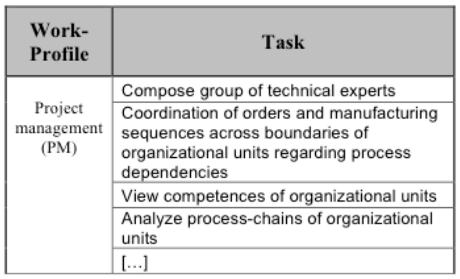
IV. Application of SDRE to *MinaBASE*

Using parts of the models created by the SDRE process, it shall now be demonstrated how the role concept can be generated systematically.

A. Step 1: Generation of the User Concept

Application of the model is based on an analysis of the business processes of a model MST company for possibilities of using *MinaBASE* and for activities, where the use of *MinaBASE* may result in a value added. Functions and units of an MST company, which may be potential users of *MinaBASE*, are:

- *Sales, external guest: MinaBASE* supports the sales process in the strategic assessment of the feasibility of customer orders, because these decisions can be made based on an IT documentation of competences and fabrication know-how. Strategic means that a general decision is made without taking into account technical details. In addition, the customer order is typed depending on whether a standard product is to be manufactured or a specification has to be met by enforcing the development in a project. The documented competences can be used as a database for sales promotion. External guests, e.g., customers or suppliers, may be given access to the system in order to inform themselves about fabrication processes used by the company or the cooperation network.
- *Project management, development:* If a customer order is classified to be not directly producible by the sales division, a project team is established based on the customer's specification. This team is composed of the project manager and technical experts. In an iterative process, they specify general solution alternatives, the commercial feasibility of which is assessed. In addition, solution approaches, such as functional patterns or prototypes, are developed in detail, the technical feasibility of which is guaranteed. Upon successful agreement with the customer, exact fabrication planning is started in the next step. Planning is based on the results of the development of a commercially and technically feasible solution.
- *Construction, fabrication planning:* Planning of fabrication, i.e., of the individual steps of production flow, may be initiated by a successful development process or a directly producible customer order, e.g., the repetition of an already executed fabrication process. In the latter case, *MinaBASE*, a system for process-oriented knowledge management, provides support by the storage of process elements of process steps and process sections and their combination in process chains, as this allows for the direct use of already executed fabrication processes [8]. This principle in weaker form may also be applied to fabrication planning based on a technical solution alternative from development. By copying or adapting existing process models or process elements, planning of the fabrication process can be accelerated. Construction and fabrication planning result in a detailed schedule for production and defined quality management tests, during which data are measured in the production process.
- *Quality management, production:* Production focuses on the execution of the process steps defined by fabrication planning in a process chain to execute the order placed by the customer. Technicians working at the machines have direct access to production and are capable of using technical parameters of the individual process elements of the process chain for adjusting the machine parameters and of measuring real data during the tests. Various areas of quality management are covered. New fabrication knowledge of attributes and parameters of process elements is generated.

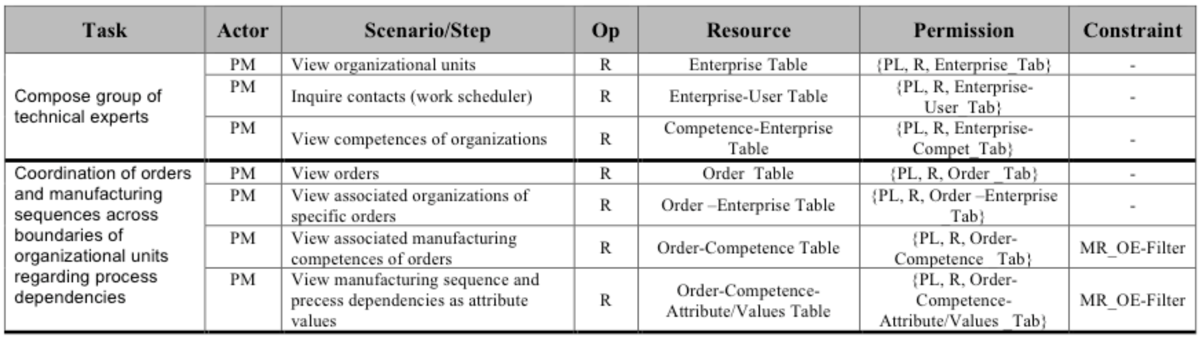


Work-Profile	Task
Project management (PM)	Compose group of technical experts
	Coordination of orders and manufacturing sequences across boundaries of organizational units regarding process dependencies
	View competences of organizational units
	Analyze process-chains of organizational units
	[…]

Figure 3. Work area project management with associated tasks from the work profile model

B. Step 2: Definition of Work Profiles and Task Definitions

According to the adaptations to the SDRE model, *MinaBASE* tasks are assigned to the enterprise units or work areas listed in the previous section. Figure 3 shows a part of the work profile model. In the work area of project management for the iterative development of solutions for a not yet solved development problem of a customer order, tasks are identified, to which the *MinaBASE* resources can be applied. These tasks are the pooling of technical experts, the analysis of fabrication competences and process chains, and the coordination of process dependencies beyond organizational units. The complete work profile model contains all tasks to which *MinaBASE* may be applied. These are the input variables for the detailed assignment of scenarios, steps, and authorizations to access resources in the following step.



Task	Actor	Scenario/Step	Op	Resource	Permission	Constraint
Compose group of technical experts	PM	View organizational units	R	Enterprise Table	{PL, R, Enterprise_Tab}	-
	PM	Inquire contacts (work scheduler)	R	Enterprise-User Table	{PL, R, Enterprise-User_Tab}	-
	PM	View competences of organizations	R	Competence-Enterprise Table	{PL, R, Enterprise-Compet_Tab}	-
Coordination of orders and manufacturing sequences across boundaries of organizational units regarding process dependencies	PM	View orders	R	Order Table	{PL, R, Order _Tab}	-
	PM	View associated organizations of specific orders	R	Order –Enterprise Table	{PL, R, Order –Enterprise Tab}	-
	PM	View associated manufacturing competences of orders	R	Order-Competence Table	{PL, R, Order-Competence _Tab}	MR_OE-Filter
	PM	View manufacturing sequence and precess dependencies as attribute values	R	Order-Competence-Attribute/Values Table	{PL, R, Order-Competence-Attribute/Values _Tab}	MR_OE-Filter

Figure 4. Refinement of tasks in use scenarios and assignment of authorizations to access the resources needed

C. Steps 3/4: Refinement of Scenarios and Assignment of Authorizations

For the first two tasks mentioned in the previous section, namely, the pooling of experts and the coordination of process dependencies, a part of the fused permission and constraint catalog is illustrated in Figure 4. For every scenario or every step, the associated operation on an object or resource is modeled, with R denoting read access (read), C denoting the creation of a new entry (create), U meaning processing (update), and D the deletion (delete) of a resource. The information of which actor accesses which resource with which operation is encapsulated as a permission by a triple of the type (actor, operation, object). At last, limitations or constraints of access are specified. For the first task, the organizational units, contact data of technical experts, and competences of the organizations stored in *MinaBASE* are considered as use scenarios. Read access (R) to the tables of the database and application components is required. In analogy, the second task is handled. Here, order data and detailed, production-related attribute values of process dependences are needed. In addition, an entry in the constraint catalog is made to ensure that the actor sees only those attribute values that are characterized as project-specific property and not as production-specific, internal know-how of an organization. This constraint cannot be implemented as a static authorization, as the assignment is made dynamically during runtime.

D. Steps 5/6: Derivation of the Role Concept

By applying the first steps of the adapted SDRE model to the identified *MinaBASE*-using enterprise units, a hierarchy corresponding to a preliminary model of the role concept may be derived on the basis of the authorizations. The highest point is the administration that is not only responsible for administrating users and their assignment to roles, but also has all other authorizations in the system. The lowest point is the external guest, who is given fewest access rights. In between, the graph may be structured horizontally and vertically. Vertical structurization results from the

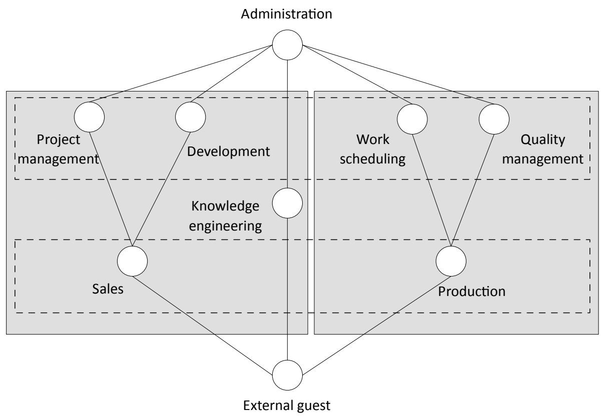


Figure 5. Preliminary role concept based on the permission catalog

degree of orientation to orders. This means that planning of working steps of a process chain and their execution are much more related to orders than the development of solution alternatives for a certain customer specification by technical experts. Horizontal structurization results from the authorization steps.

Then, the last step of the SDRE process follows, i.e., the analysis of the preliminary model for groupings of authorizations in the form of roles that exist several times and have a comparable amount of authorizations. These roles have to be eliminated. Otherwise, the catalog would list more roles than necessary, which might result in anomalies and undesired side effects in the administration of rights and roles. Review of the preliminary model taking into account the criteria described yields the role concept shown in Figure 6. Documentation of the authorizations for the individual company units shows that a separation between project management and development is not reasonable, as the access rights for the modeled scenarios and steps are identical. For this reason, both units are summarized by the developer role. The same applies to fabrication planning and

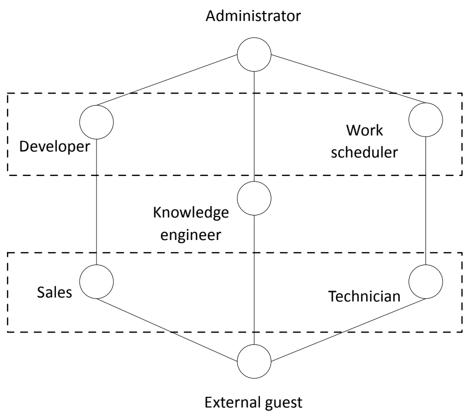


Figure 6. Revised role concept as RBAC model

quality management, as both units use *MinaBASE* for various objectives, but still have comparable use scenarios and, hence, identical authorizations. Consequently, they assume the same role in the use of *MinaBASE* , the role of the work planner. Below, the role of technician exists, who is responsible for the implementation of the plans made by the work planner. The technician is in the position to acquire the measurement data for the tests of the work planner and to define detailed parameters, such as machine instructions, as information added to process elements. To fulfill his task, the developer needs deep insight into the details of the competences and process chains, as he has to extend the strategic assessment of the sales division by a guaranteed technically possible feasibility assessment. The "knowledge engineering" component has already encapsulated the rights to update order-independent knowledge in the preliminary model. In this way, additional authorizations can be assigned specifically to a role.

V. Implementation in an IoC-Framework

This section describes the implementation of the RBAC model within *MinaBASE*. At first, the used framework, Spring Security, is introduced. After that, it is shown how static and dynamic security requirements stemming from the permission catalog as well as the constraint catalog can be fulfilled.

A. Spring Security

Spring Security is a subproject of the application framework "Spring" to control authentication and authorization in the JEE environment, i. e., in the range of business applications based on Java technology [13]. It is empowered by technologies provided by the core of Spring, such as "Inversion of Control" (IoC) using "Dependency Injection", which means a passive provisioning of an application component's dependencies by a central container known as the Spring "ApplicationContext", as well as the aspect-oriented programming (AOP) capabilities provided by that container. AOP is used for central encapsulation of cross-cutting concerns into so-called aspects, which avoids the scattering of duplicated code for realizing them across the codebase. By integrating with the hosting web container, a central hook is implemented by which a chain of filters can monitor and control the processing of HTTP requests as well as the execution of application components. This enables Spring Security to capture all elements of an application's architecture while thoroughly ensuring its security requirements using a declaratively configurable mechanism. This central hook determines whether an incoming request is trying to access a protected resource according to the supplied configuration. If this is the case, the "AuthenticationManager" (AM) is requested to authenticate and return the current principal, an abstract notion for, e. g., the currenlty logged in user, which is used by the "AccessDecisionManager" (ADM) to determine whether its role has the permission required for the protected resource in question. These two components can be controlled in a very flexible manner. For instance, during authentication, the AM can be configured to consult different providers, which in turn compare the principal's credentials by querying relational databases or LDAP directories. The ADM can be controlled by assigning static key/value-pairs of resources and required permissions or by enabling the dynamic execution of AOP-driven components. The following shows how Spring Security can be used to enforce compliance with the static and dynamic security requirements as specified in the permission- and constraint catalog in Figure 4.

B. Static aspects of security

Extending the security mechanisms across the entire architecture of *MinaBASE* requires the ADM to use a FilterSecurityInterceptor" (FSI) for securing the presentation layer as well as a MethodSecurityInterceptor" (MSI) for the application layer. To protect the application layer, a configuration of the MSI is required that determines which permissions the role of the current principal must possess to invoke components for data access and application logic. This can be realized by placing annotations directly in the application source code or through a central AOP configuration. The latter variant is used due to easier maintenance and therefore shown in Listing 1. For the protection of method invocations, a so-called pointcut, which is an entry point for the execution of code formulated as AOP-advices, is associated with a permission, whose presence will be checked by the MSI.

```
<global-method-security>
 <protect-pointcut
   expression="execution(*
    edu.kit.minabase.*CompetenceDAO.get*(..))"
   access="PERM_R_Competence"/>
</global-method-security>
```

Listing 1. Configuration of the MethodSecurityInterceptor

This restricts the data access to competences by requiring the presence of the ”PERM_R_Competence“ permission for the execution of methods, whose names start with ”get“ and are located within the CompetenceDAO class. The assignments of permissions to roles can be altered using an administrative interface at runtime. The invocation of methods for modifying and deleting information entities within *MinaBASE* as well as the execution of application logic can be restricted in a similar fashion. For securing the presentation layer, combinations of URL-patterns for protected regions and required permissions are specified, which are evaluated by the FSI during the monitoring of HTTP request processing. An excerpt of the necessary configuration is shown in Listing 2. Due to the URL-patterns being evaluated from top to bottom, the monitoring is at first disabled for static resources to achieve higher performance. Thereafter, permissions for visiting URLs matching the location for display and editing of competences are stated.

```
<http auto-config=”false”
 access-denied-page=”/denied.jsp”>
 <intercept-url
  pattern=”/static/*.*” filters=”none”/>
 <intercept-url
  pattern=”/competence/show/**”
  access=”PERM_R_Competence”/>
 <intercept-url
  pattern=”/competence/edit/**”
  access=”PERM_W_Competence”/>
 <form-login login-page=”/login.jsp”
  authentication-failure-url=”/login.jsp?error=1”/>
 <logout logout-success-url=”/logout.jsp”/>
 </http>
```

Listing 2. Configuration of the FilterSecurityInterceptor

The final step is the declaration of URLs, to which the AM will redirect unauthenticated users, that try to access a protected resource as well as URLs for authentication failures and the termination of a user’s session. These settings ensure that protected areas are not reachable for users that dont possess the required permissions. To improve the user experience, links to sections the user does not have access to should not be displayed in the first place. To achieve this, the generation of HTML needs to be controlled with permissions in mind. Spring Security is bundled with an extension that allows fragments of Java ServerPages (JSP) to be rendered according to the current user’s permissions, which is demonstrated in the following Listing 3.

```
<sec:authorize ifAllGranted=”PERM_W_Competence”>
 <a href=”/cometence/edit/...”>
   Edit this competence</a>
</sec:authorize>
```

Listing 3. Permission based generation of the user interface

This JSP-Tag assures that links to the area for editing competences are only rendered to those users that have the ”PERM_W_Competence“ permission. These three listings show how Spring Security can be used to employ a homogeneous system of permissions that stems from the SDRE permission catalog and covers the entire application architecture from data access to the presentation layer. At runtime these permissions are assigned to roles from the designed RBAC model.

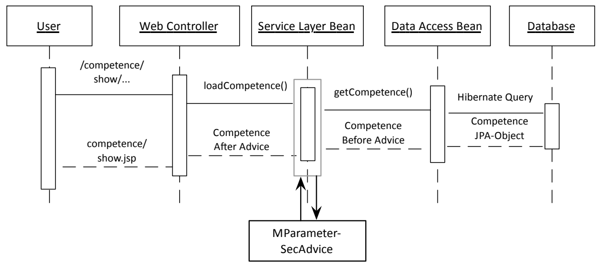


Figure 7. Integration of the MParameterSecAdvice in *MinaBASE*

C. Dynamic security aspects

In the previous section, security aspects were considered, which could be fulfilled by statically restricting access to a protected resource by requiring a specific permission to be held by the current principal. While most aspects of the permission catalog are covered by this approach, entries of the constraint-catalog as depicted in Figure 4 cannot be implemented in this fashion, because of their dynamic nature, wich means, these constraints cannot be enforced at build-time, but only at runtime. As an example, the filtering of Fabrication-specific TP of a competence’s detailed view is used. To avoid code duplication whenever Fabrication-specific TP of a *MinaBASE* information entity shall be filtered, this concern is encapsulated into a separate AOP-Advice called ”MParameterSecAdvice“, whose integration into the method’s call flow is illustrated in Figure 7. A request for the detailed view of a competence is received by a Web-controller, which initiates the data access for the current competence by invoking methods from the service layer. Once this competence is loaded as a database object, the MParameterSecAdvice is hooked into the execution flow using AOP-Weaving. The job of this component is to iterate over the competence’s parameter collection and filter out those parameteres to which the current principal has no permission. The revised competence object is then returned to the controller which starts the generation of HTML templates and sends the result to the browser. The advantage of this appraoch is the fact that the permission based filtering is encapsulated into the MParameterSecAdvice once and can be applied declaratively to multiple application components without code duplication and mixture of concerns by simple configuration in a similar fashion as described in Listing 1.

VI. Conclusion

In this paper, a role concept for the process knowledge database *MinaBASE* has been developed based on a systematic methodology called Scenario-driven role engineering. The implementation of this role concept within an IoC-Framework such as Spring has been demonstrated by utilizing Spring Security and technolgies such as AOP. At first, the *MinaBASE* approach as well as the Scenario-driven role engineering methodology were introduced. Following this, careful adjustments were made to the inputs of the SDRE process resulting from the background and purpose of *MinaBASE* without hurting the methodology's idea and principles. Then the application of the SDRE process was shown including examples on how to derive a minimal set of permissions enabling each role to fulfill its work profile. In the following section the implementation of the derived role concept using Spring Security is described in detail. Important concepts are Dependency Injection and AOP, as they enable Spring Security to ensure static and dynamic security requirements across the entire application architecture. For the implementation of security requirements that can be decided at runtime only, an example was given in order to prevent the disclosure of Fabrication-specific parameters for non-authorized persons.

References

[1] U. Hansen, C. Germer, S. Büttgenbach, and H. Franke, "Rule based validation of processing sequences," in *Techn. Proc. MSM*, 2002.

[2] D. F. Ferraiolo and R. Kuhn, "Role-based access control," in *In Proceedings of 15th NIST-NCSC National Computer Security Conference*, October 1992, pp. 554–563.

[3] G. Neumann and M. Strembeck, "A scenario-driven role engineering process for functional RBAC roles," in *SACMAT '02: Proceedings of the seventh ACM symposium on Access control models and technologies*. New York, NY, USA: ACM Press, 2002, pp. 33–42.

[4] I. Nonaka and H. Takeuchi, "The knowledge-creating company," *Harvard Business Review*, vol. 6, pp. 96–104, 1991.

[5] M. Dickerhof, "Prozesswissensmanagement für die Mikrosystemtechnik." *Statusseminar MikroWebFab, Karlsruhe*, 2003.

[6] M. Dickerhof and A. Parusel, "Bridging the Gap—from Process Related Documentation to an Integrated Process and Application Knowledge Management in Micro Systems Technology," *Micro-Assembly Technologies and Applications*, pp. 109–119, 2010.

[7] M. Dickerhof, O. Kusche, D. Kimmig, and A. Schmidt, "An ontology-based approach to supporting development and production of microsystems," *Proc. of the 4th Internat. Conf. on Web Information Systems and Technologies*, 2008.

[8] D. Kimmig, A. Schmidt, K. Bittner, and M. Dickerhof, "Modeling of microsystems production processes for the minabase process knowledge database using semantic technologies," *Proc. of the 3th Internat. Conf. on Information, Process, and Knowledge Management*, 2011.

[9] E. J. Coyne, "Role engineering," in *RBAC '95: Proceedings of the first ACM Workshop on Role-based access control*. New York, NY, USA: ACM Press, 1996, p. 4.

[10] J. H. Saltzer and M. D. Schroeder, "The protection of information in computer systems," in *Inproceedings of fourth ACM Symposium on Operating System Principles*, 1975.

[11] HL7 Security Technical Committee , "HL7 Role Based Access Control (RBAC) Role Engineering Process," January 2005.

[12] K. Boehm, W. Engelbach, J. Härtwig, M. Wilcken, and M. Delp, "Modelling and implementing pre-built information spaces. architecture and methods for process oriented knowledge management," 2005.

[13] B. Alex and L. Taylor, "Spring Security Reference Documentation," http://static.springsource.org/spring-security/site/docs/3.1.x/reference/springsecurity-single.html, last access: 06.04.2011.