SECURWARE 2012

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

ISBN: 978-1-61208-209-7

August 19-24, 2012

Rome, Italy

SECURWARE 2012 Editors

Rainer Falk, Siemens AG - München, Germany

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

SECURWARE 2012

Foreword

The Sixth International Conference on Emerging Security Information, Systems and Technologies [SECURWARE 2012], held between August 19-24, 2012 in Rome, Italy, continued a series of events covering related topics on 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.

We take here the opportunity to warmly thank all the members of the SECURWARE 2012 Technical Program Committee, as well as the numerous reviewers. The creation of such a 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 2012. 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 2012 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that SECURWARE 2012 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in emerging security information, systems and technologies.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the historic charm Rome, Italy.

SECURWARE 2012 Chairs:

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

SECURWARE 2012

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 2012 Industry Liaison Chair

Rainer Falk, Siemens AG - München, Germany

SECURWARE 2012 Research/Industry Chair

Mariusz Jakubowski, Microsoft Research, USA

SECURWARE 2012 Technical Program Committee

Habtamu Abie, Norwegian Computing Center - Oslo, Norway
Ali Ahmed, Monash University - Sunway Campus, Malaysia
Hamada Alshaer, Khalifa University of Science, Technology & Research (KUSTAR), UAE
Claudio Agostino Ardagna, Università degli Studi di Milano, Italy
David Argles, Haven Consulting, UK
Feng Bao, Institute for Infocomm Research, Singapore
Ilija Basicevic, University of Novi Sad, Serbia
Lejla Batina, Radboud University Nijmegen, The Netherlands
Georg T. Becker, University of Massachusetts Amherst, USA
Francisco Jose Bellido Outeiriño, University of Cordoba, Spain
Jorge Bernal Bernabé, University of Murcia, Spain
Catalin V. Birjoveanu, "Al.I.Cuza" University of Iasi, Romania
Lorenzo Blasi, Hewlett-Packard, Italy
Carlo Blundo, Università di Salern, Italy
Wolfgang Boehmer, Technische Universitaet Darmstadt, Germany
Ravishankar Borgaonkar, Technical University Berlin and Deutsche Telekom Laboratories, Germany
Jérémy Briffaut, ENSI - Bourges, France
Christian Callegari, University of Pisa, Italy
Juan Vicente Capella Hernández, Universidad Politécnica de Valencia, Spain
Hyunseok Chang, Bell Labs/Alcatel-Lucent, USA
Fei Chen, VMware, Inc., USA
Lisha Chen-Wilson, University of Southampton, UK

Table of Contents

Generic, Secure and Modular (GSM) Methodology for Design and Implementation of Secure Mobile Applications

Feng Zhang[1], Ioannis Kounelis[1,2], and Sead Muftic[1]
[1]Communication Systems
School of Information and Communication Technology
Royal Institute of Technology, Stockholm, Sweden
[2]Institute for the Protection and Security of the Citizen,
Joint Research Centre (JRC), European Commission, Ispra (VA), Italy
{fengz, kounelis, sead}@kth.se

***Abstract*—The generic, secure and modular methodology, described in this paper, provides a generic approach for the design and development of secure mobile applications. It is applicable to multiple mobile phone platforms and mobile operating environments. This approach treats a mobile application in a holistic way and structures it into four groups of modules: user interface modules, communication modules, security modules, and business logic modules. These four groups of modules can be designed and implemented independently and finally be integrated together. This approach not only simplifies the process of design and development of mobile applications, but also improves the reusability and robustness of mobile applications. In addition, this paper proposes a trusted layer model for designing the security modules of mobile applications, which provides generic application interfaces and comprehensive data protection. The paper finally gives an example of a secure mobile application, called SAFE Mobile Wallet, which was designed and implemented using GSM methodology.**

Keywords - mobile; generic; secure; modular.

I. INTRODUCTION

With the significant growth of mobile technologies, mobile phones have already become one of the most important accessories in people's everyday lives all over the world. Hundreds of thousands mobile applications exist today and their number is largely increasing every day. Until March 2012, only Apple's App Store offered over 500,000 iPhone applications [1]. There are various platforms, such as Android, Research In Motion (RIM), Symbian, Windows Phone, iOS (Apple), etc., allowing third parties to easily develop, test and deploy various mobile applications. Based on the reasons mentioned above, a lot of researchers, developers and fans of mobile technologies make great efforts today to design and develop various mobile applications.

At the moment, there are numerous hardware and software platforms used by different mobile devices. Therefore, the methodology of designing and implementating mobile applications varies. In addition, even though there are some generic approaches for building mobile applications [2], [3], these approaches have not taken security into account. Many security flaws [2] violate users' privacy [3] and data security and eventually affect wider deployment of those applications.

In order to solve the issues described above, this paper proposes the, here called, Generic, Secure and Modular (GSM) methodology for the design and implementation of mobile applications. The distinguished features of our GSM methodology are the following:

- It treats mobile applications in a holistic way, meaning that our methodology can be applied to develop applications for any platform, using any programming language;
- It uses a modular approach and structures mobile applications into several groups of modules. Each module is independently designed and implemented, making the implementation results reusable;
- It provides an approach for the design and implementation of security modules, which provides comprehensive security features for the mobile applications. The details are described in Section 3.4.

Therefore, we expect that the GSM methodology is a significant contribution to the process of designing, developing and deploying secure mobile applications.

The next section continues with related work and then our methodology is described in detail, covering all four modules. Finally, there is a conclusion section.

II. RELATED WORK

This section gives an overview of what has been already published related to secure mobile applications. Most of these contributions indicate how to design or implement a mobile application and how to integrate some specific features into it. Some describe suggestions for mobile application development, while others point out the challenges for the development of secure mobile applications.

Ueyama, Pinto and Madeira proposed a generic approach for constructing mobile device applications [4]. It is a middleware solution, based on generic components approach, to build adaptive applications, which can be deployed across heterogeneous devices with minimum resources overhead. It effectively solved the adaptability and flexibility issues.

Andre and Segarra proposed another generic approach, called MoleNE [5]. It works in such a way that it provides a set of generic base services as the abstraction of concepts to the developers so that they could use it to create customized applications.

Serhani, Benharref, Dssouli, and Mizouni listed the main issues to consider when developing mobile applications and mobile Web services [6]. A framework, which includes a mobile client environment and backend server/mobile Web services, is described. Finally, an example of a mobile application, developed by using the proposed framework, is given.

Ray Gonzales gives some suggestions on how to design mobile applications in [7]. These suggestions include how to structure user interfaces based on the size of a screen, some pre-defined design conventions, etc. in order to achieve convenient and comfortable interactions between users and mobile devices.

Christien Rioux presented challenges for developing secure mobile applications in [8]. He also analyzed security features for three popular mobile platforms: Windows Mobile, RIM Blackberry, and Google Android and listed some security vulnerabilities and malicious code for these platforms.

Our conclusion is that today there are not so many research and development results in the area of the generic methodology for developing secure mobile applications. Specially, few results concentrate on the security during the design and implementation phases. The methodology proposed in this paper addresses this shortage.

III. GSM METHODOLOGY

As shown in Figure 1, our GSM methodology structures mobile applications into four groups of modules: User Interface (UI) modules, communication modules, security modules, and business logic modules. The former three modules are generic, meaning that these three modules can be reused. The last module is specific, different for each mobile application. The reason for this approach is that these four groups of modules are necessary for most of the secure mobile applications. In other words, in order to create a secure mobile application, at least these four groups of modules should be included.

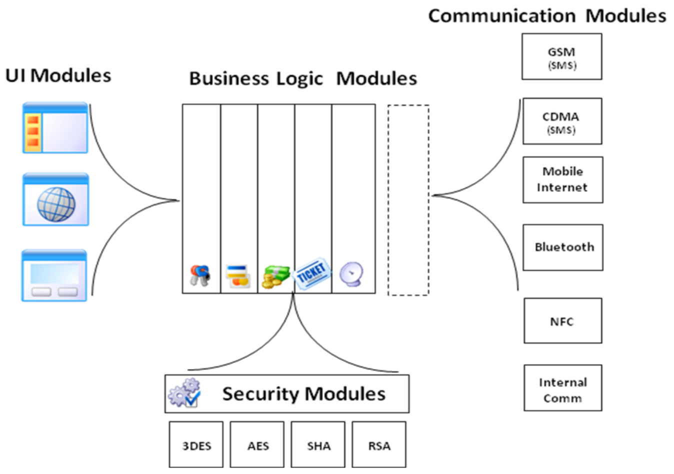


Figure 1. Internal structure of secure mobile application

Every mobile application must have user interfaces for displaying information to users and interacting with them. Some of them are simple and some are complex; this depends on the hardware and operating system of mobile phones. All mobile applications need also to communicate either with open networks or with internal modules of mobile devices. Communication modules exchange messages with networks, with other devices, and/or with other internal modules of mobile devices. Next, the security of mobile applications is also very important. Therefore, security modules that provide security features, such as generation of RSA keys, encryption, digital signatures, etc. are also necessary. These three groups of modules are not sufficient to create a complete secure mobile application, since they are separated and independent of each other and they do not perform any function specific to an application. Hence, business logic modules are needed to link all other modules together and coordinate them based on the services or functionalities provided by a specific mobile application.

GSM methodology also organizes the process of designing and creating mobile applications into three steps: (a) analyzing target platforms and devices, (b) designing and implementing the four groups of modules described above, and finally, (c) integrating all the modules and performing tests on the actual devices. As Greg Nudelman said, "*One of the challenges of mobile application design is understanding both the capabilities and limitations of each platform*" [9]. Therefore, the first step is to analyze target platforms in which the mobile application will be loaded. It includes technical details of the target platform, such as which programming language should be used, which libraries and Application Programming Interfaces (APIs) are supported, etc. The next step for developers is to design and implement the generic and specific modules as described in this paper and the final step is testing the application on the mobile devices.

A. User Interface (UI) Modules

Each mobile application must provide user interfaces for getting input from a user and for displaying output to a user. The design of user interfaces directly influences user experience. The UI may be different for different mobile devices or platforms, depending on the hardware and APIs. Therefore, the design and implementation of UI modules may also be different. There are lots of guidelines and suggestions for the design of UI modules for various devices and platforms. One of the approaches is to implement a generic object, the `UIProvider`, which provides all the necessary UI components, such as `TextField`, `Buttons`, `RadioBox`, etc. that may be used by all mobile applications running on the same platform. Even though the implementation of the `UIProvider` depends on the platform and the corresponding APIs, it should be implemented as generically as possible, so that it can be reused by mobile applications running on the same platform. In this way, various mobile applications could share one `UIProvider`, which saves development time and makes UI components easy and flexible to modify.

From a security point of view, the design of the application should always prompt the user when a security feature is not used. For example, a warning should be used when mail is sent over an insecure connection. Moreover, it is important, when having a dialogue that requires the user's interaction, to mark as default the response that will be on the user's favor, always in respect to his/her security and privacy. For instance, if an application asks the user if it is ok to sign on a page although there is no encryption, the default button should be no.

B. Communication Modules

Mobile applications must always access local or global networks. Therefore, network communication functions, such as sending/receiving Short Message Service (SMS), establishing connections through Bluetooth [10], Wi-Fi [11], 3G [12], etc. should be designed and implemented. As a result, communication modules should perform communications with open networks.

For the communication with open networks, we designed a generic object, the `CommunicationProvider`, which is convenient to manage all the communication channels, such as Bluetooth, General Packet Radio Service (GPRS) [13], Wi-Fi, SMS, etc. The `CommunicationProvider` works as a "coordinator" to support these communication channels, as required. For example, it dynamically and transparently selects the most convenient channel, controls the priority of the channels, adds/removes channels to/from the communication, etc. The priority of communication channels can be configured, depending upon usage requirements. Each specific communication channel is implemented individually and should be tested for the connections with the `CommunicationProvider`. Communication channels should be independent of each other and the procedure of establishing connections should be transparent to users.

Moreover, secure channels should be given priority over insecure ones. So, if there is an option to connect to a page with both http and https the second should be used by default, without requiring user intervention.

C. Security Modules

As explained in previous Sections, security is more and more important for mobile applications. In order to provide comprehensive security services for mobile applications, we designed a `SecurityProvider`, based on our proposed trusted layer model.

As shown in Figure 2, our trusted layer model structures the security modules into four trusted layers: Secure Element (Chip), applets, middleware and applications. The trusted layer model is equivalent with the Open Systems Interconnection (OSI) seven layer model; each layer provides services to its upper layer while receiving services from the layer below.

Layer 1: Secure Element (Chip)

This layer represents the chips used by mobile devices for mobile transactions, such as the Subscriber Identity Module (SIM)/Universal Integrated Circuit Card (UICC) chip, Micro SD chip, etc. The chip contains a microprocessor and a small memory, used in the mobile phone. It is secure and tamper resistant, so that it works as a Secure Element (SE), storing important information, such as credit card numbers, private keys, etc. Moreover, because of the compatibility and mobility of these chips, using them for storage of data ensures that users can easily migrate their data between different mobile devices. The chip layer defines how data is stored in the SE. It also defines the format of the application protocol data unit (APDU) that is used for upper layer to invoke the functionalities provided by the chips.

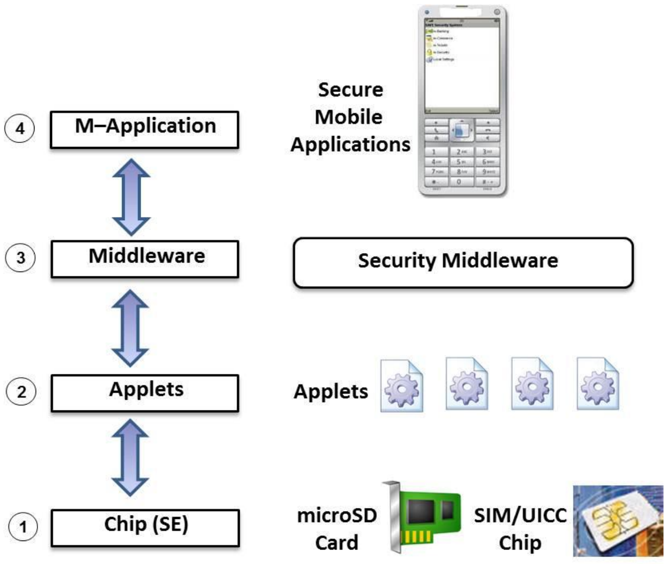


Figure 2. Trusted layer model

Layer 2: Applets

This layer represents the applets installed and functioning in the hardware of a chip. Applets interact with the chips using APDU commands, invoking functions provided by chips and providing those functions to the upper layer.

Layer 3: Middleware

Mobile applications use high-level programming APIs, but applets in the chips understand only APDU commands. Therefore, in order to make the communication between mobile applications and chips as smooth as possible, a middleware is needed. This approach is also suggested in the FIPS 201 (Personal Identity Verification - PIV) standard [14]. The middleware works as a "translator" between a high level application and an applet loaded in the chips. On one hand, it provides APIs to high level applications, so that developers should not need any knowledge about chips and applets loaded in them. As a result, it makes the invocation of the functions, which are provided by the applet in the chips, transparent to mobile applications. On the other hand, the middleware communicates with the chips using APDU commands.

The middleware layer defines how the upper layer software applications invoke the functions provided by applets. It provides APIs, for upper layer mobile applications to invoke the functions provided by the applets. The middleware layer contains the implementation of these APIs, the role of which is to translate the upper layer application data calls into APDUs and vice versa.

An implementation of the middleware can be seen in [15]. In this paper, we have created a middleware that allows the flow of data from a UICC to the upper layer applications and eventually the user of the mobile device. The core principle of this implementation was to make the whole process transparent to the application developers and even more to the end user. As Saltzer and Schroeder [16] stated in the 1970s, if the security mechanisms make the product or device harder to use, then the users will simply avoid using them.

Layer 4: Applications

The application layer is the one closest to the end-user. It comprises the Graphical User Interface (GUI), interacting with and providing security functionalities for users.

A `SecurityProvider` was designed in order to provide security functionalities. It is designed as follows:

The application layer comprises the interfaces to access the implementation of each security service. And it provides the `SecurityProvider` as an interface to get the instance of these security services. For example, the following pieces of code are used to create an instance of the `GenerateKeyPair` service:

```
SecurityProvider secprovider = new
SecurityProvider();
IGenerateKeyPair genkp =
secprovider.GenerateKeyPair();
genkp.generateKeys(AlgorithmID);
genkp.saveKeys(keypairs);
```

Security services are implemented in the security applets, functioning on the chips. Therefore, during the execution of the codes listed above, the application needs to communicate with a specific applet that implements the invoked security service through the middleware. The user is required to enter a correct Personal Identification Number (PIN) in order for the middleware to access the applet. The PIN-based authentication mechanism is one of the security properties of the applets. Each applet has a default PIN and the user is able to set his/her own PIN. If the user authentication is successful, a secure connection is established between the middleware and the applet. Meanwhile, the middleware translates the service request into corresponding APDU commands, according to the rules and formats defined by the applet. The middleware sends the APDU commands to the applet via the established secure connection. The applet executes the required function and returns the execution result in the form of APDU response code to the middleware, which then translates the code to an application-layer service response.

By this approach, the `SecurityProvider` can provide as many security services as the chip supports. In addition, the application-level security mechanisms can always be added in order to provide higher security. For example, the data exchanged between the mobile application and the applet is encrypted using AES with the hash value of the applet authentication PIN as the symmetric key. In this way, the data is stored/fetched in/from the chips in encrypted form, which prevents disclosure to the middleware so that the data is not revealed if the middleware is compromised.

D. Business Logic Modules

Business logic modules are the core of each mobile application. They process application data, messages, etc., coordinate various other modules and functions, and provide specific application services to mobile users. The design and implementation of business logic modules depends on the functions that need to be provided by a mobile application. A good approach is to structure them in a modular form, so that they are individually as simple as possible. In other words, these business logic modules should utilize modules from other three groups as much as possible. The process of implementing business logic modules includes also integration with all the other three groups of modules.

E. Example application: SAFE Mobile Wallet

This section describes an example of the design and implementation of one secure mobile application using our GSM approach. The mobile application is called SAFE Mobile Wallet, which is one of the components of our Secure Applications for Financial Environments (SAFE) System [17], [18]. It was implemented by using Java Platform Micro Edition (Java ME) and supports various financial transactions, such as mobile banking, mobile payment Over-the-Counter (OTC) / Over-the-Air (OTA), mobile ticketing, mobile parking, etc. SAFE Mobile Wallet is continuously expanding with new functionalities.

The UI of SAFE Mobile Wallet comprises two levels of menus and several data handling forms. The hierarchy of UI panels is shown in Figure 3. Each panel is designed and developed using `MenuItemsSelection` template and then all UI objects are linked with each other by buttons or actions. One generic object, `UIProvider`, was implemented, which provides UI related functions, such as `createFormImage`, `addTextBox`, `addStringItem`, etc.

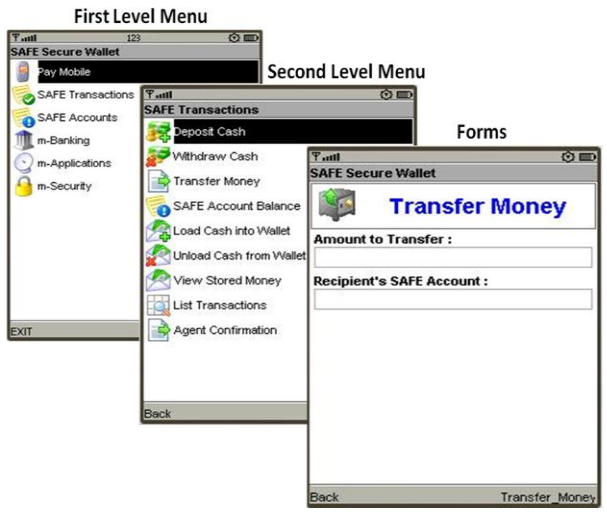


Figure 3. Hierarchy of UI Panels of SAFE Mobile Wallet

The SAFE Mobile Wallet uses three types of protocols to communicate with back-end servers: GPRS, Bluetooth, and SMS. Therefore, there are three corresponding generic communications objects: `BluetoothConnection`, `GPRSConnection`, and `SMSConnection`. Each of them handles one of the corresponding protocols. In addition, we implemented a generic `CommunicationProvider` object to manage these three communication objects. Each communication object returns its corresponding connection as an object to the `CommunicationProvider`. So, the business logic modules just need to invoke the `CommunicationProvider` to send out or receive messages through a specific communication protocol.

For communications with the UICC, SAFE Mobile Wallet uses UICC middleware in order to securely store and retrieve data from the UICC. This means that SAFE Mobile Wallet is able to invoke methods provided by the middleware internally without any separate communication method, as shown in Figure 4. In order to read and save data on the UICC, the Security and Trust Services API (SATSA) [19] was used.

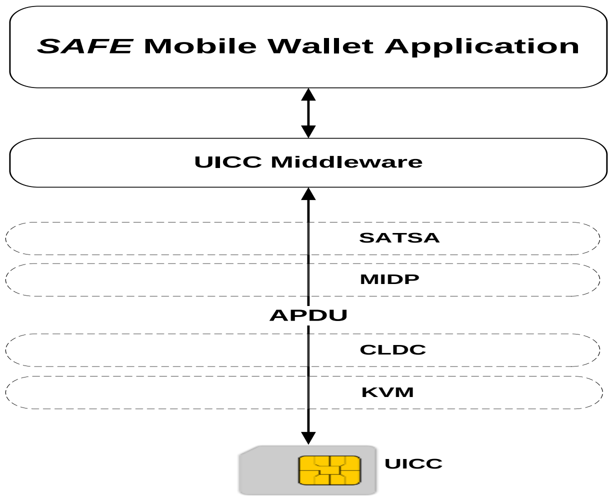


Figure 4. Communication between SAFE Mobile Wallet and UICC

The UICC middleware is a transparent, lightweight, secure and autonomous software module used by the SAFE Mobile Wallet. It was designed and implemented in such a way that the process of communicating with the UICC is transparent to the SAFE Mobile Wallet, which does not distinguish any difference between fetching and storing data from/to the smart card and manipulating data as files on its own system. Figure 5 shows an example of storing data on the UICC.

The SAFE Mobile Wallet provides several security services for its financial transactions, including user authentication, message confidentiality and integrity, non-repudiation, authorization, etc. [17]. For those services, `MobileSecurityProvider` module was designed and implemented. It provides various security functions, such as `generateRSAKeys`, `AESEncrypt/AESDecrypt`, `generateMessageDigest`, etc. All security services provided by the Wallet, are based on these functions. Therefore, whenever an application requires new security features, a developer just needs to extend the `MobileSecurityProvider` module for new functions and cryptographic algorithms. Implementation of the `MobileSecurityProvider` object can be different, depending on the platform, programming language, APIs, etc. `MobileSecurityProvider` was implemented using Java ME, so it can be used on any mobile phone that supports Java.

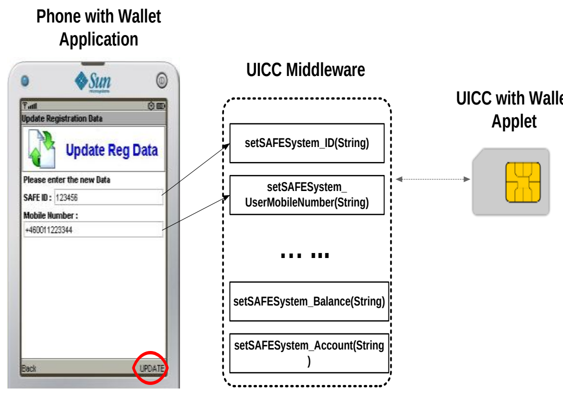

Phone with Wallet Application
Update Registration Data
Update Reg Data
Please enter the new Data
SAFE ID : 123456
Mobile Number : +46001122334
Back
UPDATE
UICC Middleware
setSAFESystem_ID(String)
setSAFESystem_UserMobileNumber(String)
... ...
setSAFESystem_Balance(String)
setSAFESystem_Account(String)
UICC with Wallet Applet

Figure 5. Example of the process for storing data on the UICC

The UICC middleware was also designed to function in a secure way. The middleware has no semantic knowledge of any data that it parses and passes in both directions. This gives mobile applications an option to store encrypted data in the UICC. Also, the middleware has no knowledge of passwords or keys used for authentication or cryptography. Another security feature is that the middleware keeps no data in its internal memory. All changes and modifications of data are saved directly in the UICC, which provides strong protection of data and also prevents synchronization issues (mostly when dealing with online, real-time transactions). Finally, the middleware follows the authentication mechanism required by the UICC: strong authentication protocol and secure channel between an application and UICC [20]. As a result, mobile users must authenticate themselves to the Wallet applet before being able to read and write data. If a user fails to provide the correct password after a predefined number of attempts, the UICC locks and can only be unlocked by the administrator.

As one of the components of our SAFE System, the SAFE Mobile Wallet is used to perform several types of mobile applications, such as mobile banking, mobile payment, mobile ticketing, etc. Therefore, business logic modules for those applications are designed as follows:

1. When a user starts SAFE Mobile Wallet, the Wallet asks the user to specify his/her PIN, if it is initial activation of the Secure Mobile Wallet. The PIN is then encrypted using AES algorithm, with hash value of the PIN as an encryption key, and is stored locally. Later, when a user starts the Wallet, the correct PIN is required in order to authenticate the user. After that, the Wallet initiates detection of available communication protocols. If a SAFE PoS Station supporting Bluetooth protocol is detected, the Wallet

will use Bluetooth to communicate with the Station and pass it through with other SAFE Servers. Otherwise, the Wallet tries to activate GPRS protocol. If GPRS is not supported, Wallet will use SMS as the communication channel. The described detection procedure is performed automatically and in the background;

2. After successful authentication, the user can perform all Wallet transactions [17]. All SAFE transactions are executed in a request-response way. The user enters the required data for a transaction. The Wallet creates a transaction request message and sends it to the SAFE Servers through an available communication protocol. SAFE Servers process the transaction and send the transaction result back as a response.

IV. CONCLUSION AND FUTURE WORK

This paper achieves several goals. First, it describes our proposed GSM methodology for the design and development of mobile applications. This methodology structures mobile application into four groups of modules and each of these modules is independently developed and reused, which makes the design and development of secure mobile applications more efficient, flexible and expandable. Second, it describes our proposed trusted layer model. This model provides a framework for designing secure mobile applications and more importantly, it represents a way of integrating the security mechanisms, provided by chips, with mobile applications. Finally, it introduces the approach of using a middleware for the communication between mobile applications and chips. This approach effectively protects the data, since the data is stored in the chips instead of mobile devices. It is much more difficult to compromise the chips than mobile applications.

Future research activities can be conducted in the following aspects: a. make the middleware more generic so that it is not dependent on upper layer application and lower layer chips; b. improve the trusty and security of middleware; c. extend the security modules to use biometric information, such as finger prints.

REFERENCES

[1] The App Store, There's an app for that. Over 500000, actually, Apple. In: http://www.apple.com/iphone/built-in-apps/app-store.html [Accessed March 16, 2012].

[2] P. Hornshaw: Mobile Apps Can Create Security Issues, Oct. 5 2010. In: http://www.appolicious.com/tech/articles/3388-mobile-apps-can-create-security-issues [Accessed March 16, 2012].

[3] iPhone Applications that transmit credentials using 'unsafe' protocols, Oxolab Blog, Apr. 26, 2010. In: http://blog.0x0lab.org/2010/04/unsafe-iphone-applications/ [Accessed March 16, 2012].

[4] J. Ueyama, V. Pinto and E. Madeira, "Exploiting a Generic Approcah for Constructing Mobile Device Applications", Proceedings of the Fourth International ICST Conference on COMmunication System software and middleware, Verona, Italy, July, 2011.

[5] F. Andre and M. Segarra, "A Generic Approach to Satisfy Adaptability Needs in Mobile Environments", Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, Maui, Hawaii, Januare, 2000.

[6] M.A. Serhani, A. Benharref, R. Dssouli, and R. Mizouni: Toward an Efficient Framework for Designing, Developing, and Using Secure Mobile applications, International Journal of Human and Social Sciences (IJHSS), World Academy of Science Engineering and Technology, Volume 5 Number 4, pp. 271-278, 2010.

[7] Tips for Designing Mobile Applications, Clickbrand, January 12, 2009. In: http://www.clickbrand.com/blog/web-design/tips-for-designing-mobile-applications/ [Accessed March 16, 2012].

[8] C. Wysopal: The Challenges of Developing Secure Mobile Applications, Veracode, July 22, 2009. In: http://www.veracode.com/blog/2009/07/the-challenges-of-developing-secure-mobile-applications/ [Accessed March 16, 2012].

[9] G. Nudelman: Designing Mobile Search: Turning Limitations into Opportunities, UXmatters, March 8, 2010. In: http://www.uxmatters.com/mt/archives/2010/03/designing-mobile-search-turning-limitations-into-opportunities.php [Accessed March 16, 2012].

[10] "Bluetooth", Wikipedia [website], Available: http://en.wikipedia.org/wiki/Bluetooth [Accessed March 16, 2012].

[11] "Wi-Fi", Wikipedia [website], Available: http://en.wikipedia.org/wiki/Wi-fi [Accessed March 16, 2012].

[12] "3G", Wikipedia [website], Available: http://en.wikipedia.org/wiki/3G [Accessed March 16, 2012].

[13] "GPRS", Wikipedia [website], Available: http://en.wikipedia.org/wiki/Gprs [Accessed March 16, 2012].

[14] The Federal Information Processing Standards Publication Series of National Institute of Standards and Technology (NIST): Personal Identity Verification of Federated Employees and Contractors (FIPS 201), March 2006

[15] I. Kounelis, H. Zhao, S. Muftic, Secure Middleware for Mobile Phones and UICC Applications, Mobile Wireless Middleware, Operating Systems, and Applications, Springer, 2012, pp. 143-152

[16] M. Bishop, Computer Security: Art and Science. Boston, MA: Addison-Wesley, 2003, pp. 348-349

[17] F. Zhang, "Secure Applications for Financial Environments (SAFE) System", Licentiate Thesis, Royal Institute of Technology, Stockholm, Sweden, June 2010

[18] S. Muftic, F. Zhang, and K. DeZoysa, "SAFE System: Secure Applications for Financial Environments Using Mobile Phones", Proceedings of IADIS International Conference IADIS e-Society, February 2009.

[19] "SATSA Developer's Guide", Oracle [website], Available: http://download.oracle.com/javame/config/cldc/opt-pkgs/api/security/satsa-dg/index.html [Accessed March 16, 2012].

[20] CardContact, "OpenCard Framework", December 28, 2010, Available: http://www.openscdp.org/ocf/ [Accessed March 16, 2012].

Security Aspects for Large Scale Distributed Environments

The AutoBAHN use case

Giorgos Adam[1,2], Christos Bouras[1,2], Ioannis Kalligeros[1,2], Kostas Stamos[1,2,3], and Ioannis Zaoudis[1,2]

[1]Computer Technology Institute and Press "Diophantus", N. Kazantzaki Str, University Campus 26504, Rio Greece
[2]Computer Engineering and Informatics Dept., University of Patras
[3]Technological Educational Institute of Patra, Greece
{adam,stamos,bouras,zaoudis}@cti.gr, kallige@ceid.upatras.gr

Abstract — **Automated Bandwidth Allocation across Heterogeneous Networks (AutoBAHN) is a tool under active development that supports a Bandwidth on Demand (BoD) service, intended to operate in a multi-domain environment using heterogeneous transmission technologies. The AutoBAHN system aims at providing a guaranteed capacity, connection-oriented service between two end points. Due to the level of access that the tool has to critical parts of the network, the importance of a trustworthy Authentication and Authorization Infrastructure (AAI) cannot be overestimated. This paper highlights the design and implementation for the Authentication and Authorization Infrastructure which is part of the AutoBAHN service and the decisions taken. The AAI is a service dedicated to enforce system security and to prevent unauthorized access and usage of resources. The BoD service modules may interact with AAI multiple times during a single request execution. After the initial authentication and authorisation check, the BoD system will apply additional, specific to BoD rules and policies to the request. The security mechanisms allow trustworthy operations in a multi-domain service without significant impact on performance.**

Keywords - Bandwidth on Demand; AAI; security; Quality of Service

I. INTRODUCTION

The GN3 European project [1] is a research project funded by the European Union and Europe's National Research and Education Networks (NRENs). It is a continuation of the previous GN2 project [19] and aims at building and supporting the next generation of the pan-European research and education network, which connects universities, institutions and other research and educational organizations around Europe and interconnects them to the rest of the Internet using high-speed backbone connections.

In the context of this project, a BoD service is being developed and the service is supported by the AutoBAHN tool. The AutoBAHN system is capable of provisioning circuits in heterogeneous, multi-domain environments that constitute the European academic and research space and allows for both immediate and advanced circuit reservations. The overall architecture of the AutoBAHN system, its goal and the network mechanisms it employs are thoroughly presented in [2]. This paper highlights the AAI architecture of the AutoBAHN service, the implementations challenges, the decisions taken and the basic security aspects and components of the AutoBAHN system.

AutoBAHN is using part of GÉANT's AA Framework [20] which addresses security issues for a number of different multi-domain network services in the GÉANT Service Area. This framework provides software developers with a common and flexible authentication and authorization solution to facilitate their software development process.

The rest of the paper is structured as follows: Section 2 describes the GN3 project while Section 3 presents the general architecture of the AutoBAHN system. In Section 4, we analyze how similar projects handled AA issues while Section 5 presents the general architecture related to authentication and authorization procedures. Section 6 describes the way that communication between system components can be considered secure. Section 7 presents some quantitative measurements and finally, Section 8 concludes the paper and presents future fields of this study.

II. GN3

Gigabit European Advanced Network Technology (GÉANT) is the main European multi-gigabit computer network for research and education purposes. It brings together over 400 participants from 32 NRENs, TERENA [21], DANTE [22], and over 20 subcontractors and third parties. It provides a dedicated, high capacity, 50,000 km data network that brings together 40 million users in research institutions across 40 countries, underpinning critical projects that would previously have been impossible without its capacity and reliability.

In the context of the GN3 project, a number of activities and services are prototyped and tested. Among them is AutoBAHN BoD service.

This BoD service is an end-to-end, point-to-point bidirectional connectivity service for data transport. It allows users to reserve bandwidth on demand between the participating end points. The data transport capacity dedicated to a connection can range from 1 Mbps up to 10 Gbps in steps of 1 Mbps. This service is offered collaboratively by GÉANT and a set of adjacent domains (NRENs or external partners) that adhere to the requirements of the service. These joint networks form a multi-domain area where the service is provided.

The service offers a high security level in the sense that the carried traffic is isolated from other traffic. It has to be noted that the traffic is isolated at logical layer and not necessarily at physical layer. This means that the core network will carry data from multiple users but there will be no “crosstalk” between the traffic streams.

III. AUTOBAHN BOD SYSTEM

The AutoBAHN system is comprised by the Inter-Domain Manager (IDM), a module responsible for inter-domain operations of circuit reservation on behalf of a domain. This includes inter-domain communication, resource negotiations with adjacent domains, request handling and topology advertisements.

Furthermore, in order to build a real end-to-end circuit, the Domain Manager (DM) is another module that manages intra-domain resources. The IDM has an interface to the local DM from which it undertakes all intra-domain functions (abstracting the topology towards the IDM, scheduling and pre-reserving resources, monitoring etc.). This southbound interface of the IDM is the part of the AutoBAHN system that needs to be tailored to the domain-specific conditions.

In each domain, the data plane is controlled by the DM module using a range of techniques, including interfaces to the Network Management System (NMS), signaling protocols or network elements. As part of the AutoBAHN, a dedicated and independent Technology Proxy module allows the support of a range of technologies and vendors according to domain and global requirements.

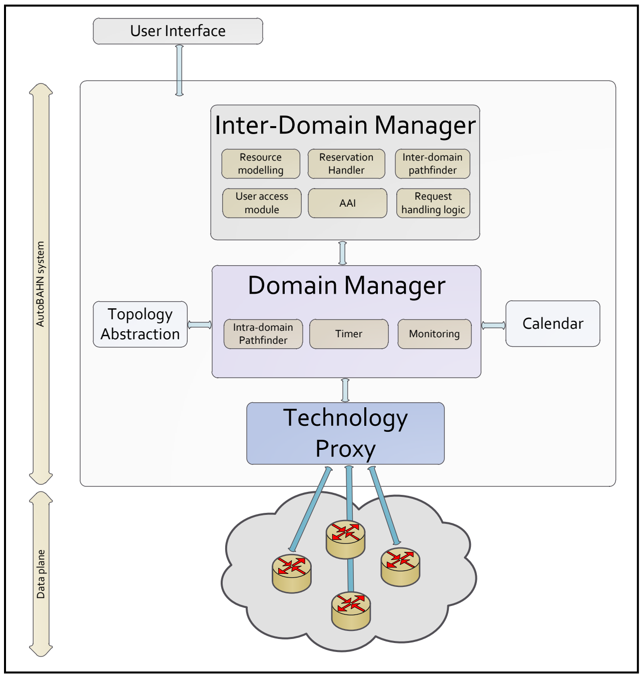


Figure 1. Basic architecture of AutoBAHN

The local NMS or service provisioning system, monitoring infrastructure, administration policies and security, may need to be adjusted for each networking domain making each Technology Proxy implementation and configuration unique. However, the design of the DM has been optimized to support modular deployment and leverage the management infrastructure already deployed in any domain.

The above set of modules is deployed in each domain (NREN) that participates in the BoD service. A web based graphical environment (WebGUI) is used as a centralized portal for user access to the whole set of deployed instances.

The above described architecture defines a multitude of communication interfaces that transfer sensitive information over the network. A potential security compromise (eaves dropping, message modification, unauthorized access, message replay) could have very important consequences as the system manages production networks. Furthermore, because of its distributed nature, the system needs a well-defined distributed authentication and authorization architecture as it spans a large number of independent administrative domains.

IV. RELATED WORK

The AutoBAHN BoD system has been influenced by a number of other projects dealing with similar challenges for bandwidth on demand provisioning. In this section we present some of the most closely related ones, with an emphasis on their approach to AAI.

The Dynamic Resource Allocation across GMPLS Optical Networks (DRAGON) project [3] is also conducting research and developing technologies to enable dynamic provisioning of network resources on an inter-domain basis across heterogeneous network technologies. It mainly deals with GMPLS enabled domains and in a smaller scale compared to AutoBAHN. Regarding the AAI, the DRAGON project incorporates AAA policy into path computation, resource allocation and signaling functions. This requires high level associations of policy with users (or groups of users) as well as lower level associations of policy with actual network elements at a fidelity sufficient to implement meaningful policy based resource allocations. These two levels are loosely described as call control and connection control. Their approach is the synthesis of higher level AAA information into policy information which is associated with the Traffic Engineering (TE) resource level. They also utilize higher level AAA information to develop a set of policy rules. The TE policy data and the policy rules are used during path computation to incorporate AAA policy into provisioning operations. AAA policy decision can be combined with TE based provisioning decision [4].

OSCARS/BRUW project [5] which is another BoD service focuses on Layer 3 Multiprotocol Label Switching (L3 MPLS) QoS. Regarding AA, requests for inter-domain reservations are authenticated in the originating domain on the basis of an individual user and in the subsequent domains on the basis of the adjacent domain [6]. Users are authenticated and authorized for actions in their home domain and inter-domain authorization depends on the domains that participate. It also depends on the assurance that a request is coming from a trusted server in a trusted domain. Normally, all requests forwarded between domains are signed SOAP messages. The forwarded message adds the name of the originating user in case other domains wish to use that information for authorization or auditing. Currently, at the time of provisioning no further authentication is done. Provisioning is triggered by the time of the reservation. Once the provisioning has been completed, any traffic coming from the specified ingress router is able to use the requested bandwidth.

Moreover, the University of Amsterdam's Advanced Internet Research group has published a number of papers describing both the networking and the AAA issues for such a system [7][8][9]. They are using IETF's AAA Framework [10] and OASIS eXtensible Access Control Markup Language (XACML) Version 2 to describe policy, which is also followed by OSCARS for Authentication. They have also defined a Network Description Language, which is an RDF-based method to describe networks and to facilitate inter-domain interoperability [11].

V. AAI ARCHITECTURE

AutoBAHN uses part of GÉANT's AA Framework which utilizes existing frameworks, industry standards and best practices in order to avoid re-inventing the wheel and to take advantage of its extensible design. It is Java-based, making use of Spring Security Framework, Crowd Integration library, OIOSAML java library and Maven.

In addition, we have developed our own architectural elements, such as our multi-domain user authorization which is described below, in order to bind and supplement the above technologies and meet our unique AAI requirements.

The current AA Framework implementation allows developers to make their own choice of Authentication Providers, User Attributes Providers and ACL services to use: the diagram below (Figure 2) shows the options offered to the service developers.

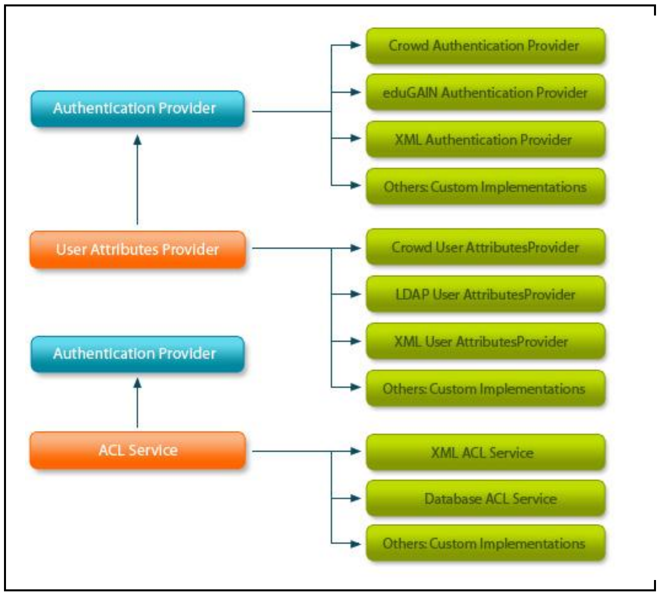

Authentication Provider
Crowd Authentication Provider
eduGAIN Authentication Provider
XML Authentication Provider
Others: Custom Implementations
User Attributes Provider
Crowd User AttributesProvider
LDAP User AttributesProvider
XML User AttributesProvider
Others: Custom Implementations
Authentication Provider
ACL Service
XML ACL Service
Database ACL Service
Others: Custom Implementations

Figure 2. The architecture of AA Framework [20]

AutoBAHN can be configured to use XML or Atlassian Crowd [15] for Authentication and User Attributes Provider. It can also support the existing eduGAIN [12] infrastructure for authentication and authorization. In addition, LDAP [14] or Relational Databases can be used as Authentication and User Attributes Providers. They also have the additional capability of supporting Access Control Lists for flexible definition of authorization policies.

A. *User Authentication*

The AutoBAHN system has been designed in such a way so that multiple authentication methods may be used, in a modular way. The authentication mechanism is based on the Spring Security Java framework [13] that provides advanced authentication, authorization and other security features for enterprise applications.

The architecture of the authentication mechanism is based on a simple flow in which the main system components that are invoked during the authentication procedure are the Authentication Manager and the Authentication Provider. The user submits his credentials to the AutoBAHN Server and an Authentication Request is created at server-side. This request is sent to the Authentication Manager which is responsible for forwarding requests through a chain of Authentication Providers. The provider will request from the UserDetails Service to provide the granted authorities for this user. These authorities are later used during the Authorization phase. The Authentication Manager receives back the result of the described procedure and decides whether the authentication is successful or not.

TABLE I. IMPLEMENTATION CHOICES AVAILABLE FOR DEVELOPERS [20]

Provider	**Available choices**		
	Authentication Provider	***User Attributes Provider***	***ACL Services***
Attlasian Crowd	Yes	Yes	N.A.
eduGAIN	Yes	N.A.	N.A.
LDAP	Yes	Yes	N.A.
Relational Databases	Yes	Yes	Yes
XML	Yes	Yes	Yes

When a user connects to the graphical environment to submit a reservation request, his supplied credentials can be authenticated against the Authentication Provider which is currently used. The authorization mechanism is able to cooperate with interchangeable authentication modules as long as the authentication provider also supplies the necessary attributes that enable authorization decisions. As the eduGAIN scenario is the most complicated and interesting one, it will be described in more detail below.

In principle, when a user tries to access the AutoBAHN system, the user is redirected to the Single Sign-On (SSO) service of his/her federation. Then the user is authenticated through the federation software which sends the SSO response and SAML 2.0 authorization back to the AutoBAHN system. The response contains both authentication and authorization information as SAML 2.0

attributes. Finally, the AutoBAHN system checks the SSO response and SAML 2.0 attributes and responds to the user with a permission or denial to access the resources. The attributes that are transmitted are the following:

1) *Identifier*: A unique id of the user that wants to make a reservation. This could be either the name or the email of the user or a combination of both.
2) *Organization*: The organization/domain/federation of which the user is a member.
3) *Project Membership*: This attribute contains a specified value (e.g. AUTOBAHN) that demonstrates that this user is an authorized AutoBAHN user.
4) *Project Role*: This attribute offers granularity in terms of the subset of available actions that the user is allowed to perform and can contain values such as Service user, AutoBAHN administrator, etc.

The various project roles currently supported are:

1) *Service User*: People from e-science communities, other BoD systems, external client applications that become “service owners”.
2) *Network Administrator*: People responsible for the data plane e.g., the underlying data network infrastructure.
3) *Autobahn Administrator*: People responsible for the control plane software.

In AutoBAHN system, the above attributes are considered to be equivalent of granted authorities meaning that based on the policy that is defined by the administrator, those attributes also define the appropriate jurisdictions and capabilities that a user can have during the usage of the system.

B. *Multi-domain User Authorization*

Authorization is the function of specifying whether a user has the access rights to perform an action on the system resources. AutoBAHN implements multi-domain user authorization, which means that the above procedure is done on every single Domain Manager.

After the authentication phase, the user is able to request access to the available resources. The authorization procedure takes place at the Domain Manager which determines whether the user is authorized to access the requested resource. This decision is based on the access policies that each DM has defined.

For operations that are decided along a multi-domain path (

Figure **3**), there are multiple Domain Managers. Thus, the decision has to be taken in every domain along the reservation path based on user attributes that have to be transmitted with the reservation request and mapped to the policies implemented by each domain.

As described earlier, the user attributes are retrieved during the authentication phase. These attributes are then forwarded to each domain of the reservation path at server-side. Before a request is examined by the system at each domain, the attributes are compared against the policy module to check whether the user has the required privileges. The policy is based on logical operations among the user attributes: identifier, organization, project membership and role.

Figure **3** shows the whole procedure for authentication and authorization when a user wants to create a service request. In Step 1, the user (through a web browser) tries to access the service submission interface and the request is intercepted by the eduGAIN filter.

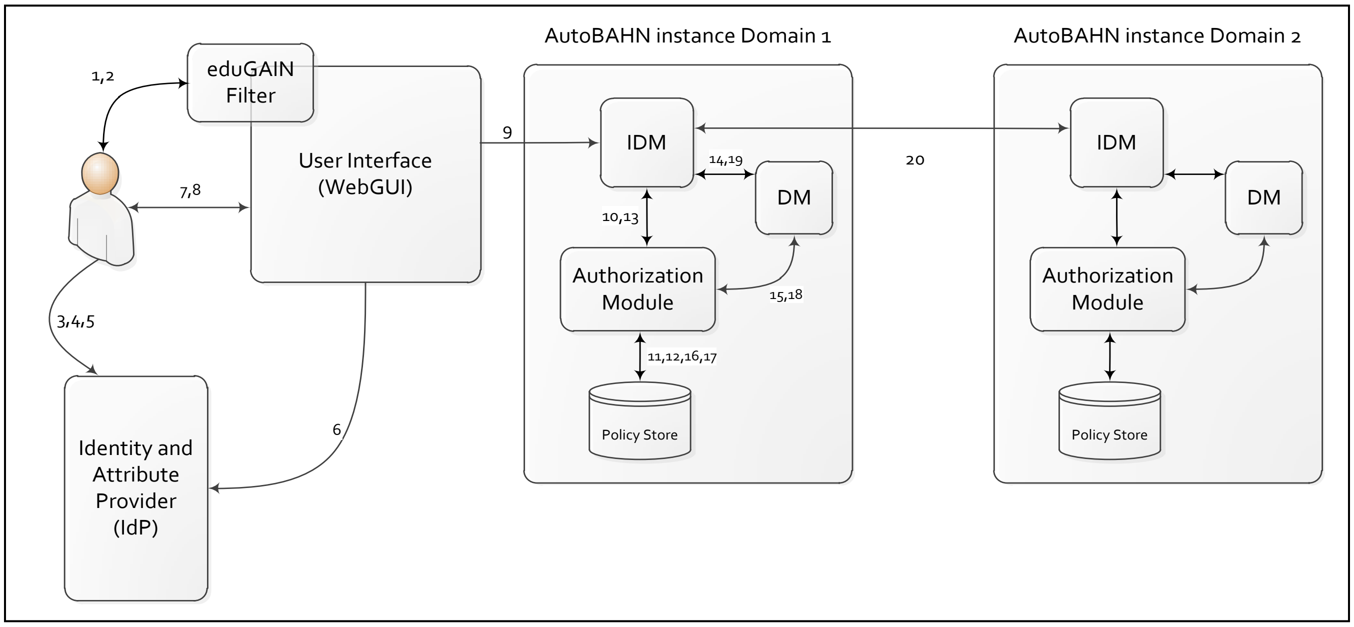


Figure 3. The user creates a reservation between different domains

The eduGAIN filter redirects the user to his local AAI (Step 2). The user's web browser sends an http request to the IdP server (Step 3). The IdP server sends to the web browser a page to authenticate the user and the user submits his credentials to the IdP server (Steps 4 and 5). The IdP server redirects the user to the WebGUI request page, and associated attributes are also sent (Step 6). The user fills in necessary parameters and submits the service request which may bundle multiple circuit reservation requests (Steps 7 and 8). The WebGUI forwards the service request and user attributes to the initiating IDM (Step 9).

The IDM deals with each reservation in the service separately. For the first reservation, it forwards the user attributes and reservation parameters to the AuthR module and the AuthR module constructs a policy evaluation query (Steps 10 and 11). The query is checked against the existing policies stored in the Policy Store and the AuthR module returns an answer (Steps 12 and 13). Assuming the response allows such a request, the IDM forwards it to the DM for intra-domain checking (Step 14). The DM calculates possible paths and forwards the reservation parameters to the AuthR module (Step 15). The AuthR module constructs a policy evaluation query and the query is checked against the existing policies stored in the Policy Store (Steps 16 and 17). The AuthR module returns an answer and the DM replies to the IDM about the feasibility of the reservation (Steps 18 and 19). The IDM forwards the request and the user attributes to the next domain along the path for further processing (Step 20) and finally, this process is repeated for all domains until the first reservation request has been evaluated. If a service request contains more than one reservation, this process is repeated for all reservations within this service request.

VI. TRUSTED AND SECURE COMMUNICATION BETWEEN SYSTEM COMPONENTS

To ensure a secure and trusted communication between system components, WS-Security standard [23] is used in addition with Edugain PKI infrastructure [24].

WS-Security (Web Services Security, short WSS) is a flexible and feature-rich extension to SOAP to apply security to web services. It is a member of the WS-* family of web service specifications and was published by OASIS.

The protocol specifies how integrity and confidentiality can be enforced on messages and allows the communication of various security token formats such as SAML, Kerberos, and X.509. Its main focus is the use of XML Signature, XML Encryption and XML Timestamp to provide end-to-end security and in our case all three options are available for use.

WS-Security makes heavy use of public/private key cryptography [16]. With public key cryptography, a user has a pair of public and private keys. A central problem for use of public-key cryptography is confidence (ideally proof) that a public key is correct, belongs to the person or entity claimed (i.e., is 'authentic') and has not been tampered with or replaced by a malicious third party. The usual approach to this problem is to use a public-key infrastructure (PKI) in which one or more third parties, known as certificate authorities, certifies ownership of key pairs.

AutoBAHN makes use of eduGAIN PKI for validating the identity of the components. The trust establishment process is enabled by means of using TLS connections for each eduGAIN interaction and including XML-Sig digital signatures for the appropriate protocol elements and assertions.

eduGAIN inter-component trust is based on X.509 certificates. It is rooted at a set of Certification Authorities (CA) created and maintained within the project. This set will be referred to as the eduGAIN truststore and all AutoBAHN components accept any of the CAs contained by the truststore as valid roots of trust. CAs in the eduGAIN truststore conform to the eduGAIN Certificate Policy, a document defining the rules and procedures agreed by the eduGAIN participants to rely on digital public certificates issued to the components of the infrastructure.

At least one of these CAs will be specifically established and run by the project. This root CA is referred as the eduGAINCA. The self-signed certificate of the eduGAINCA is the minimum content of the eduGAIN truststore.

A. *PKI Structure*

The structure of the eduGAIN PKI is shown in Figure 4. Each CA inside the eduGAIN truststore (shown as "Acc CA" in the figure) is accredited to issue certificates for components in a particular branch of the eduGAIN component identifier namespace (shown as "CId" in the figure). Certificates for components outside these branches are under the eduGAIN CA. The eduGAINCA issues certificates only to other CAs and these subordinated CAs will in turn be responsible for issuing certificates to the individual components. The eduGAIN infrastructure provides at least one of these subordinated CAs, known as the eduGAINSCA.

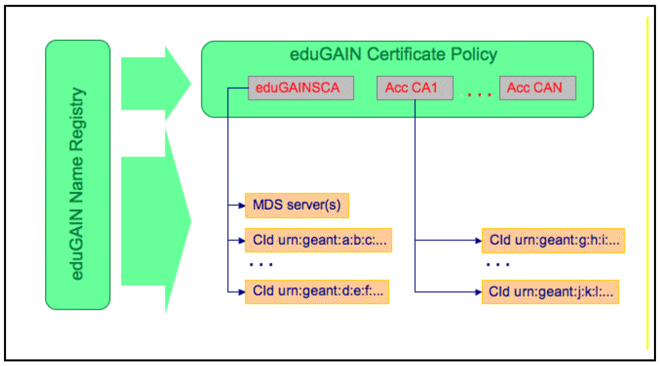


Figure 4. eduGAIN PKI Structure

The eduGAINSCA is able to provide a set of separately managed Registration Authorities (RA), according to the management procedures of the different eduGAIN namespaces under its responsibility.

B. *Trust Validation Procedure*

Trust validation is performed by eduGAIN components according to a two-step procedure:

1. The received certificate shall be evaluated to check whether the whole trust path correctly resolves to the eduGAIN root of trust.
2. The eduGAIN component identifier contained in the Subject Alternate Name extension of the received certificate shall be evaluated against the metadata available for this interaction. It must match with the component identifier as stored in these metadata.

A failure in any of the verifications above causes a rejection of the requested operation with a TrustError result. This procedure implies that for a proper trust evaluation, all metadata exchange through the MDS must contain the eduGAIN component identifiers applicable in each case.

Unless otherwise specified in the corresponding profile, all connections between any two eduGAIN components uses TLS and perform two-way certificate validation (both initiator and responder) according to the above procedures.

Validation of the certificates associated with XML Signatures follow the procedures described above.

In principle, when the client module wants to communicate with another module (the resource), it sends its request to the required resource along with its X.509 certificate signed by eduGAIN CA. The resource authenticates the client by validating its certificate using eduGAIN API. The certificate contains identification information that allows the resource to authenticate only designated clients.

The detailed procedure in the context of the AutoBAHN system for the trusted communication between AutoBAHN modules is as followed.

1) The AutoBAHN module that wants to communicate (client) must have a certificate so no interaction for credentials is needed. The X.509 certificate is issued by a Certificate Authority (CA) subordinated to one of the eduGAIN roots of trust.
2) The client module sends its request and the certificate to the resource.
3) The resource module performs trust validation by checking that the whole trust path of the certificate correctly resolves to the root(s) of trust defined by eduGAIN.
4) The resource checks that the client module is allowed to access it.
5) The resource provides the requested answer to the client module.

VII. Quantitative Measurements

In order to verify the scalability of the system, we have taken some quantitative measurements using Apache JMeter [25] as benchmarking tool. The test scenario specified the submission of 200 requests in time frame of 50 seconds resulting in 4 requests per second. The following graph presents the response time against the number of active threads. The lines refer to various steps during the submission of a reservation request in Client Portal.

The response time peaks at 330 milliseconds when the greatest number of requests is being processed.

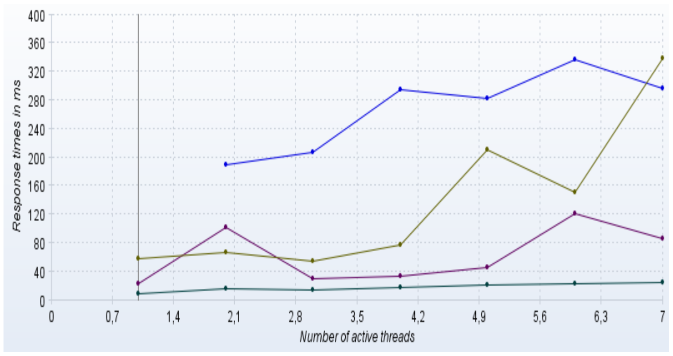


Figure 5. Response Time vs Active Threads

VIII. Conclusions and Future Work

In this paper, we presented an Authorization and Authentication Infrastructure that is used in AutoBAHN project to ensure a secure and trusted communication among its components.

AutoBAHN is considered a distributed system since each NREN deploys an instance based on their specific network technology and their needs in general. Because of that, it was important to emphasize in multi-domain user authorization ensuring that this communication is secured by WS-Security specification.

Also, a strong advantage of AutoBAHN's authentication and authorization mechanism compared to approaches of similar projects is the fact that it can support third party providers such as LDAP or Atlassian Crowd increasing the flexibility and the interoperability of the project. Furthermore, the AA process takes place in each single domain that is involved to a reservation path making the allocation of network resources more secure and robust. It's not a one-time procedure that takes place only in the originating domain.

AutoBAHN is designed in a highly scalable manner based on modules that perform specific tasks and communicate with one another. AuthR module handles AA tasks and then communicates with local DM, which later on sends the appropriate messages through the IDM to the rest of the AutoBAHN instances. This separation allows for a more convenient administration of the AAI in order to keep it up to date by adopting latest technology standards.

Currently AutoBAHN has been deployed in 6 NRENs creating a pan-european bandwidth on demand service. Several more NRENs have expressed interest or are already in the process of joining the service. This service is currently fully operable and being offered to NREN staff and clients.

The main lesson learnt was that the utilization of existing frameworks and open standards enabled us to implement our own custom AAI solution for a complex multi-domain environment and at the same time ensure extensibility and flexibility. In addition, due to the importance of the security mechanisms to the operation of the service, we consider it essential that the specification of the AAI architecture needs to be an integral part of the system design from the start.

Next step is the full integration of the latest AA Infrastructure that is a result of SA2/Task5 Common Framework efforts [17]. The WS-Security standard is based on WSS4J, a well-known and commonly used Java library for securing Web Services. In our case, we can additionally use AA Framework's two way SSL communication as an extra security layer above WS-Security, which is based on X.509 certificates that are issued by eduPKI CA, the next-generation CA of GÉANT's project [18].

ACKNOWLEDGMENT

The authors would like to acknowledge the contribution of their GN3 partners and especially the JRA2 Task 2 and SA2 Task 5 groups.

REFERENCES

[1] "GN3 European Project," [Online]. Available: http://www.geant.net/pages/home.aspx. [retrieved: June, 2012]

[2] M. Campanella, R. Krzywania, V. Reijs, A. Sevasti, K. Stamos, C. Tziouvaras, and D. Wilson, "Bandwidth on Demand Services for European Research and Education Networks," in 1st IEEE International Workshop on Bandwidth on Demand, San Francisco (USA), pp. 65-72, 2006.

[3] F. Leung, J. Flidr, C. Tracy, X. Yang, T. Lehman, B. Jabbari, D. Riley, and J. Sobieski, "The DRAGON Project and Application Specific Topologies," in Broadnets, San Jose, California, USA, pp. 1-10, 2006.

[4] Xi Yang, Tom Lehman, Chris Tracy, Jerry Sobieski, Shujia Gong, Payam Torab, and Bijan Jabbari, "Policy-Based Resource Management and Service Provisioning in GMPLS Networks", IEEE INFOCOM, pp. 1-12, 2006

[5] C. Guok, "ESnet On-Demand Secure Circuits and Advance Reservation System (OSCARS)," in Internet2 Joint Techs Workshop, Salt Lake City, Utah, 2005.

[6] Chin Guok, Robertson, D., Thompson M., Lee J., Tierney B., and Johnston W. "Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System", Broadband Communications, Networks and Systems, pp. 1-8, 2006. BROADNETS 2006. 3rd International Conference

[7] L. Gommans, C. de Laat, and R. Meijer, "Token based path authorization at interconnection points between hybrid networks aind a lambda grid," in Proceedings of IEEE GRIDNETS 2005

[8] L Gommans B van Oudenaarde F Dijkstra, C. de Laat, T. Lavian I Monga, A Taal F Travostino, and A Wan "Applications drive secure lightpath creation across heterogeneous domains,' IEEE Communications Magazine, vol. 44, no. 3, pp. 100-106, 2006.

[9] Y Demchenko, L. Gommans, C. de Laat, A. Tokmakoff, and R. van Buren, "Policy based access control in dynamic Grid-based collaborative environment," in International Symposium on Collaborative Technologies and Systems, pp. 64-73, 2006.

[10] J. Vollbrecht, P. Calhoun, S. Farrell, L. Gommans, G. Gross, B. de Bruijn, C. de Laat, M. Holdrege, and D. Spence, "AAA authorization framework," IETF RFC 2904, Aug. 2000.

[11] J.J van der Ham, F. Dijkstra, F. Travostino, H.M.A. Andree, and C.T.A.M de Laat "Using RDF to describe nfetworks"' iGrid 2005 special issue, Future Generation Computer Systems, vol. 22, no. 8, pp. 862-867, 2006.

[12] "Deliverable DJ5.2.3,3: Best Practice Guide - AAI Cookbook - Third Edition", [Online]. Available: http://www.geant2.net/upload/pdf/GN2-08-130-DJ5-2-3-3_eduGAIN_AAI_CookBook-1.pdf [retrieved: June, 2012]

[13] "Spring Security Framework," [Online]. Available: http://static.springsource.org/spring-security/site/ [retrieved: June, 2012].

[14] "LDAP," [Online]. Available: http://en.wikipedia.org/wiki/Lightweight_Directory_Access_Protocol [retrieved: June, 2012].

[15] "Crowd," [Online]. Available: http://www.atlassian.com/software/crowd/ [retrieved: June, 2012].

[16] "Public Key Cryptography," [Online]. Available: http://en.wikipedia.org/wiki/Public-key_cryptography [retrieved: June, 2012].

[17] G. Forge, "GÉANT AA Framework," [Online]. Available: https://forge.geant.net/forge/display/AAI/Home [retrieved: June, 2012].

[18] "eduPKI", [Online]. Available: http://www.geant.net/SERVICES/ENDUSERAPPLICATIONSERVICES/Pages/eduPKI.aspx [retrieved: June, 2012].

[19] "GN2 Project", [Online]. Available: http://www.geant2.net/ [retrieved: June, 2012]

[20] "GEANT AA Framework", [Online]. Available: http://www.geant.net/Services/NetworkPerformanceServices/Pages/GEANT_Framework.aspx [retrieved: June, 2012]

[21] "TERENA", [Online]. Available: http://www.terena.org/ [retrieved: June, 2012]

[22] "DANTE", [Online]. Available: http://www.dante.net/ [retrieved: June, 2012]

[23] "WS-Security", [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss [retrieved: June, 2012]

[24] "eduPKI", [Online]. Available: http://www.geant.net/Services/UserAccessAndApplications/Pages/eduPKI.aspx [retrieved: June, 2012]

[25] "Apache JMeter" , [Online]. Available: http://jmeter.apache.org/index.html [retrieved: June, 2012]

Identification and Implementation of Authentication and Authorization Patterns in the Spring Security Framework

Aleksander Dikanski, Roland Steinegger, Sebastian Abeck
Research Group Cooperation & Management (C&M)
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
{ a.dikanski, abeck }@kit.edu, roland.steinegger@student.kit.edu

***Abstract*—In the development of secure applications, patterns are useful in the design of security functionality. Mature security products or frameworks are usually employed to implement such functionality. Yet, without a deeper comprehension of these products, the implementation of security patterns is difficult, as a non-guided implementation leads to non-deterministic results. In this paper, the Spring Security framework is analyzed with the goal of identifying supported authentication and authorization patterns. Additionally, a best practice guide on implementing the identified patterns using the framework is presented. A real world case study is presented, in which the findings are employed to implement security requirements in a web application. With this approach it is possible to overcome the gap between pattern-based security design and implementation to implement high quality security functionality in software systems.**

Keywords - security patterns; security framework; security engineering; authorization; authentication

I. INTRODUCTION

Security engineering aims for a consecutive secure software development by introducing methods, tools, and activities into a software development process [1]. As such, each phase of the software development needs to consider security aspects: in the analysis phase security requirements are identified, in the design phase security functionality is modeled in conjunction with the main business functionality and finally, security solutions are realized in the implementation phase.

Security patterns are an agreed upon method to describe best practice solutions for common security problems [2]. When designing security functionality for an application such patterns can be instantiated in the design model to cover a certain security requirement.

The reuse of existing security functionality, i.e., in the form of security components, frameworks or products, is considered best practice as well, as they usually cover a great percentage of existing security requirements. Their maturity can usually not be achieved by implementing it completely new, so self-made solutions should extend it as well. By doing so, the quality of the security functionality of the developed application is increased. Also, as the main focus of software development lies upon the implementation of the business functionality, the reuse of existing functionality increases the efficiency of the implementation process.

Implementing security patterns using existing security functionality is complicated. For one, their built-in flexibility to support many different application contexts leads to a high complexity, requiring a deep understanding of the internal workings. This often raises the question, if and how the required security patterns can be implemented with the selected product. In such a case, the security functionality needs to be analyzed by security experts to determine the supported patterns.

Such an analysis is especially useful, if a model-driven approach is used to automatically generate security-related artifacts from design models. The identified and supported patterns of the framework or product can be used to describe the target platform and to generate framework artifacts from design models. Such an approach is part of a reuse-based security engineering approach, which we outlined in earlier works [3].

In this paper, the capabilities of the popular open source authentication and authorization framework Spring Security [4] are examined. The goal thereby is to identify support for common pattern by Spring Security and provide a reusable catalog of best practice advice on how to implement them in a high quality fashion. Theses informal description can be used by developers in the need to evaluate security frameworks as well as a guide to implementation. Also, they can be used to describe formal transformation rules for a model-driven approach.

The rest of this paper is structured as followed: Section 2 introduces the Spring Security framework and discusses related work. In Section 3, the relationship of the pattern-based framework description to our reuse-based security engineering approach is described. The identified security patterns and their equivalent implementation using Spring Security are covered in Section 4. In Section 5, a real-world case study is presented, which shows the security pattern implementation using Spring Security. A conclusion and outlook on future work closes the body of this paper.

II. BACKGROUND AND RELATED WORK

The following section provides a background on the Spring Security framework and discusses related works.

A. *Background on Spring Security*

Spring Security is an open source Java framework, providing highly flexible and extensible authentication, authorization, and access control solutions [5][6].

The modular framework consists of loosely coupled components, which are connected using dependency injection. The core classes and their dependencies are shown in Figure 1. The *Authentication* class stores user information. It is part of a *SecurityContext* class for every authenticated user in an application. An *AuthenticationManager* loads this data and which verifies the authenticity of users using offered credentials and information from a user store [5].

To intercept secured resource access, classes extend the *AbstractSecurityInterceptor* class, which is the central class in terms of authorization. Thereby, the *SecurityContext* and *SecurityMetadataSource* classes offer information about the current user and the secured object respectively. Access decisions are performed by the *AccessDecisionManager*, which is also called by the *AbstractSecurityInterceptor*. The *AccessDecisionManager* calls voters, which decide whether access is granted or not and which can be added dynamically to the application. Thus, the voter system abstracts from an access control mechanism.

Although it can be used for desktop applications, the main purpose of Spring Security is to secure web applications based on the Java Platform Enterprise Edition (JEE, [26]). The framework integrates with many authentication technologies and standards, e.g., Lightweight Directory Access Protocol (LDAP), Central Authentication System (CAS), OpenID and OAuth. Spring Security also provides support for basic role-based access control [6]. Due to its flexible architecture the framework can easily be adapted and extended to support other forms of authentication and authorization and access control as well.

B. *Related Work*

Due to the identification of security patterns, the work is based on common security pattern literature. A comprehensive catalog of abstract and context-specific security patterns for, e.g., operating systems, can be found in [2]. Identity management as well access control patterns are discussed in [7] and [8]. Patterns specific to the JEE platform are described in [9]. Authorization patterns for the Extensible Access Control Markup language (XACML) are discussed in [10]. An excerpt from the patterns presented in these works is used in this paper to show their support by the Spring Security framework. Pattern based security engineering processes are discussed in [11] and [12], yet they do not consider the implementation of patterns using security platforms.

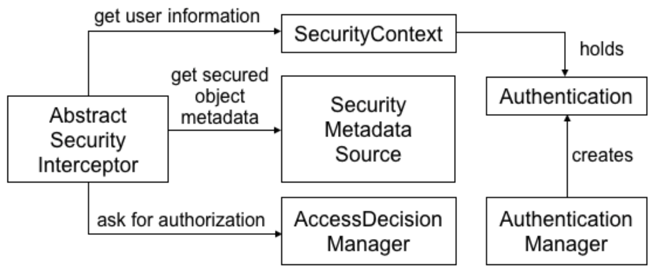


Figure 1 The main classes of the Spring Security Framework

An automated retrieval of security patterns in existing software, such as discussed in [13] and [14], would be useful in the identification process. Unfortunately, the retrieval rate of the approaches is still to low to be useful for our goals. Applying them would only show the patterns implemented in the software not all possibilities of the security framework. This is why a manual approach was applied.

The pattern-based platform description presented here is a feasible enhancement to model-driven security approaches, which is not considered by other such approaches, e.g., [15][16][17]. We aim at describing the target security platform using security patterns, to simplify the transformations and easily adapt them to new platforms.

Background information on the Spring Security framework, its inner relations and concepts as well as its usage can be found in the community documentation as well as in [5] and [6]. These descriptions are not based on security patterns and do not show all possible applications of the framework.

III. REUSE-BASED SECURITY ENGINEERING

The pattern-based identification and description of security functionality in existing frameworks is part of a reuse-oriented security engineering approach, presented in [3]. We argue for reuse of existing security functionality as well as knowledge throughout the phases of development processes to increase the quality and the development efficiency of the implemented software artifacts. Security problems, which can not be covered by existing models and functionality, can benefit from a reuse approach by extending or adapting them to a new context.

For one, the reuse of knowledge about possible threats and attacks against information resources, as well as appropriate countermeasures, is feasible in the analysis of security requirements of an application.

The topic discussed in this paper covers the design and implementation phase of the engineering process. In the design phase existing security knowledge should be used to determine possible solutions for security problems. Security patterns offer a proven method for describing such best practice solutions and can be integrated with common design patterns [2]. The implementation of security solutions should be based on existing security functionality, e.g., provided by products, frameworks or components. These are more mature and field tested, than a new implementation and usually offer support for existing security standards and technologies.

Yet, to support the security engineering process, there is a need for knowledge of the frameworks used for securing the software product. During the design phase, knowledge about patterns that are supported by a framework is needed in order to avoid incompatibilities between design and implementation. When implementing the design it is beneficial to know how to implement a pattern with a framework. This leads to the need of pattern identification in security frameworks.

IV. AUTHENTICATION AND AUTHORIZATION PATTERN IDENTIFICATION

The following section describes the pattern identification and implementation process using the Spring Security framework. A focus was put on authentication and authorization patterns, as these are the focus of the framework as well. Thereby a distinction is made between the format of security guidelines describing policy patterns, and architectural patterns, describing components using and evaluating the policies.

The patterns were identified manually by using practical experience on securing applications with the framework, its openly accessible source code and reference documentation as well as a book about the framework [5]. The selected patterns to identify in the framework cover several areas within authentication and authorization. Another reason in favor for the selection is their publicity. Commonly known patterns were selected from [2] and [9].

A. Authentication Patterns Description

The patterns described in this chapter are supporting decisions in the software development process concerning authentication.

1) Authentication Policy Patterns

We have not found an abstract authentication pattern description in the aforementioned literature, which we deem relevant. The *Authentication Information* pattern defines, that a subject has to deliver some sort of information to prove an association to an identity in an application.

In [9], several mechanisms to authenticate a subject are specified, which offer three specializations of the pattern. The distinction is made on what kind of prove has to be presented, i.e., the subject deliver information it knows, e.g., a username and password, it owns, e.g., from a smartcard, or intrinsically has or is, e.g., finger prints. Lastly, the fourth specialization of the *Authentication Information* pattern is the combination of any two or more of these three concretions, which is called multi factor authentication.

2) Authentication Architectural Patterns

Information about known identities needs to be stored for comparison with user input. The abstract *User Store* pattern [9] defines, that user information is stored in some kind of repository. Depending on the type of authentication mechanism different implementations of the *User Store* are required. A LDAP directory or a database, containing usernames and passwords, are examples of User Store pattern implementation.

Enforcing the authentication needs specification of the required components in the software architecture and their interplay. The *Authentication Enforcer* pattern [9] describes these components and their interaction in a web-based application. The pattern abstracts from the applied authentication mechanism, defined through the policies, to enhance reuse. Another aim of the pattern is to centralize authentication functionality and therefore to reduce redundancy.

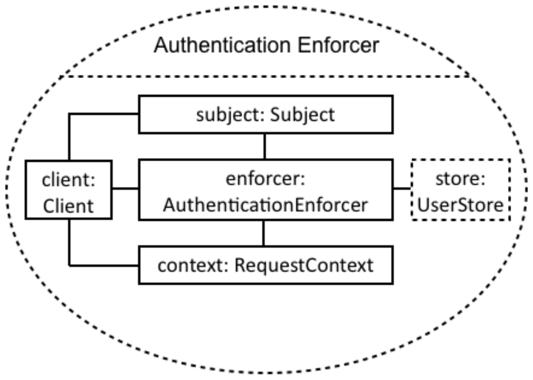
The main component is the eponymous *Authentication Enforcer*, to which authentication requests of the client are sent to. It takes the information offered by the clients from the request context and compares it to data in the user store. On successful verification, a subject containing information gained from the user store on the subject is created.

B. Authentication Patterns Identification

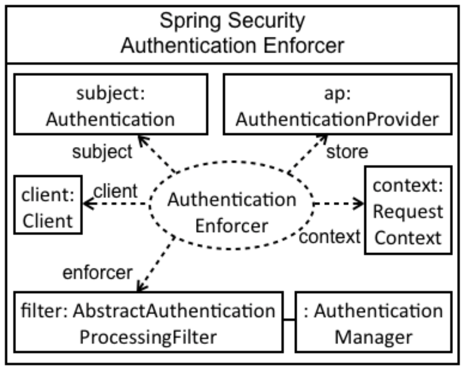
The main interface for implementing the *Authentication Information* pattern is the Spring Security *Authentication* interface, as its implementation offers information depending on the authentication mechanism. The *Authentication* interface is closely coupled to the *AuthenticationProvider* that loads the user information.

Accessing storages with the Spring Security framework, as required by the *User Store* pattern, is achieved through different implementations of the *AuthenticationProvider* interface. Each implementation represents a different *User Store* and uses varying *Authentication* concretions, e.g., the *OpenIDAuthenticationProvider* offers OpenID authentication by creating an *OpenIDAuthenticationToken* that implements the Authentication interface. The *AuthenticationManager* uses the *AuthenticationProvider* to verify authenticity of users. An *AuthenticationManager* and its *AuthenticationProviders* can be configured using XML. An example configuration is shown in Figure 2. The default authentication manager is used and the custom authentication provider class can be inserted.

In Spring Security, the *Authentication Enforcer* pattern is implemented using the filter chain mechanism introduced by



(a) Authentication Enforcer Pattern



(b) Spring Security Implementation of Authentication Enforcer Pattern

Figure 2 Authentication Enforcer Pattern and Implementation with Spring Security

TABLE I. SUPPORTED AUTHENTICATION PATTERNS

Authentication Patterns	Spring Security Implementation
Authentication Information	• Single and multi-factor authentication using username-password, OpenID, X.509 certificates, HTTP Basic and Digest authentication (native) • adaptable to other authentication methods using 3rd party frameworks
User Store	• XML configuration, LDAP, Database, properties file (native) • adaptable to 3rd party user store
Authentication Enforcer	• Authentication filters for Java Servlet filter mechanisms

the Java Servlet Specification [18]. The *DefaultLoginPage-GeneratingFilter* is executed if the login URL of the application is called and renders a login page to the client. When the client sends the rendered login form, the *UsernamePasswordAuthenticationFilter* tries to authenticate the client using the configured *AuthenticationManager*. Another example is the *BasicAuthenticationFilter*, which gets the username and password from the request according to RFC 1945 [19] and verifies authenticity. There are also filters for CAS or OpenID authentication, because they depend on an external user store and therefore must be treated differently. Due to the statelessness of HTTP, the *SecurityContextPersistenceFilter* is needed, which persists the security context including the authentication in the HTTP session before responding to a request and recovers the security context at the beginning of the next request.

Writing an own filter for supporting, e.g., biometric authentication is possible, too. For each filter specified in the filter chain, there must be a Java class with the same name. The filter chain and authentication provider offers flexibility in adding new authentication mechanisms and user stores needed to support the Authentication Enforcer pattern.

C. Authorization Patterns Description

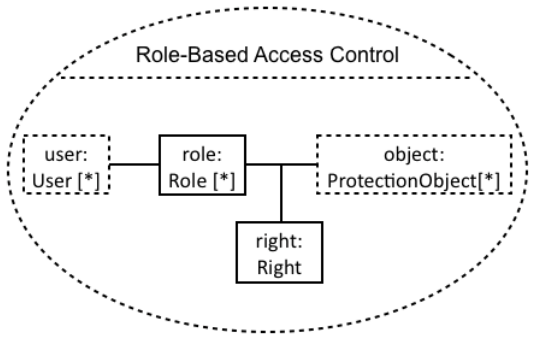
This section introduces patterns that can be used to describe or enforce authorization. Because there is a close relationship between authentication and authorization, some architectural patterns require authentication or even offer it.

1) Authorization Policy Patterns

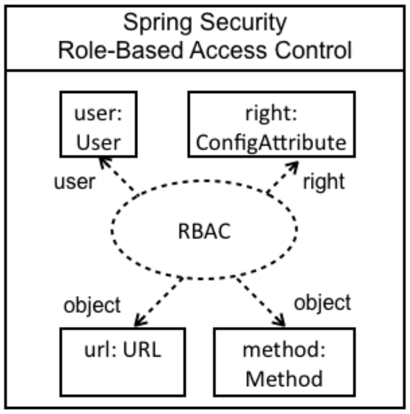
The *Authorization* pattern [2] is used to define access control for resources at a high level of abstraction. A subject is assigned a right for a resource. High level of abstraction means, that subject, right and resource are not specified concretely and can be of any kind.

The direct interpretation of the *Authorization* pattern is called *Identity-Based Access Control* (IBAC [2]). Due to the structure, the concrete *Subject* gets directly assigned a *Permission* to access a *Resource* in a specific way. Thus a fine-grained definition of access control is established. Usually IBAC is implemented using access control lists (ACL).

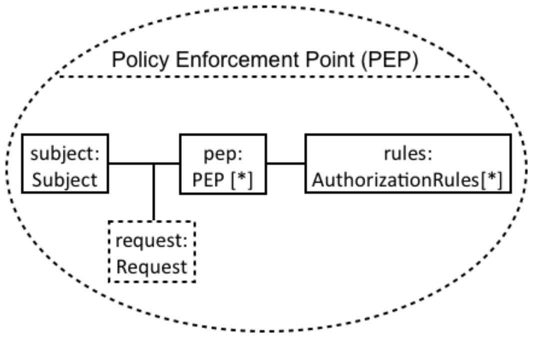
Role-Based Access Control (RBAC), described as a pattern in [2], is a specialization of the *Authorization* pattern, which refines the right assignment. Instead of directly assigning rights, a *Subject* gets assigned a *Role*, which



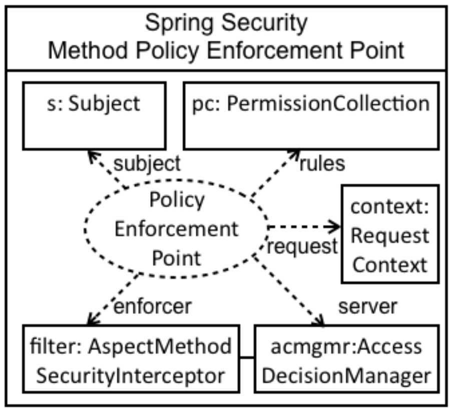
(a) Role-Based Access Control Pattern



(b) Implementing Role-Based Access Control with Spring Security



(c) Policy Enforcement Point Pattern


Spring Security Method Policy Enforcement Point
s: Subject
pc: PermissionCollection
subject
rules
Policy Enforcement Point
request
context: Request Context
enforcer
server
filter: AspectMethod SecurityInterceptor
acmgmr:Access DecisionManager

(d) Spring Security Implementation of Policy Enforcement Point for Method Access

Figure 3 Authorization Patterns in Spring Security

represents a set of *Permissions* to access a *Resource*. Thus, it is possible to assign *Subjects* with the same access rights using *Roles* among a system reducing the complexity of rights management.

Another concretion of the *Authorization* pattern is *Attribute-Based Access Control* (*ABAC* [20]). In contrast to RBAC, *Permission* can be defined through expressions using all available *Attributes* of *Subject*, *Resource* or *Environment*.

2) Authorization Architectural Patterns

Besides defining authorization policies, there are patterns describing their enforcement, i.e., access control. An abstract example for enforcement of access control is the *Policy Enforcement Point (PEP)* pattern, also known as *Reference Monitor* [2][21]. The PEP defines components and flows needed to control access to a resource in an abstract way. Requests to a *Protected Resource* shall be intercepted by the *PEP*. According to *Authorization Rules*, which consist of *Authorization* items, access is granted or denied.

Another concretion of the *PEP* is the *Authorization Enforcer* pattern [9]. The purpose of the pattern is to control access in a JEE application. Due to this circumstance, there are several variations of the pattern using different Java specifications. Requests from a *Client* are intercepted and redirected to the *Authorization Enforcer*, which uses the *Authentication Provider* to set the *Permissions* to the already loaded *Subject*. Thus the pattern needs an authenticated *Subject*, e.g., set by the *Authentication Enforcer* pattern. With the *Permissions* of the *Subject*, the *Authorization Enforcer* decides, whether the access is granted or rejected.

The *Intercepting Web Agent (IWA)* pattern [9] helps in separating application logic from authorization and authentication logic. It can also be used to add access control and authentication after the development of an application. The name already suggests that the patterns operational area is web application development. *Client* requests are intercepted by the eponymous *IWA*. Either the *Client* authenticates itself and its authentication information is persisted through a cookie or the *Client* tries to access a *Resource* directly, in which case the *IWA* loads the previously persisted information of the *Subject*. The request is forwarded by the *IWA*, if the *Subject* is authorized.

D. Implementation of Authorization Patterns

The following section discusses implementing the authorization patterns with the Spring Security framework.

1) Policy Pattern

Due to the voter mechanism used for access decisions, the framework can be enhanced to support several access control patterns, thus it supports the *Authorization* pattern. The sections about ABAC, RBAC, and IBAC show different voters supported by the framework and indicate the flexibility. By implementing an *AccessDecisionVoter*, it is possible to access external frameworks or software and to gain extra information needed for the decision or to ask for the decision from external software.

RBAC raises the need for defining *Roles* of *Users*. In Spring Security *Roles* are called (*Granted-*) *Authorities* [5]. *Authorities* can be assigned to Users *via* configuration or loaded from a *User Store* [5]. A documented best practice is the arrangement of *Authorities* into hierarchies [6]. *Roles* are assigned to *Users* and *Rights* are assigned to *Roles*. Thus a hierarchy is built and *Users* are assigned several *Rights* through their *Role*.

Rights are assigned to *Roles* to access a *Resource*. Spring Security supports the protection of methods and URLs as *Resources* [5]. In the configuration or annotation the corresponding *Right* is used, as can be seen in Figure 2 (c) and (d). Thus only *Users* with a *Role* having the *Right* to modify a resource are allowed to access it. When implementing a web application based on the REST (Representational State Transfer) paradigm [22], the approach of protecting URLs is preferred. Otherwise, method security and the use of annotations according to the Java Specification Request (JSR) 250 should be used. Thus, the flexibility in changing the security framework is saved.

ABAC is not directly supported by Spring Security, but can be easily implemented as shown next. Spring Security offers the Spring Expression Language (SpEL [5]) to describe access control. Instead of annotating a right to methods or to a URL, expressions can be used. When evaluating to *True*, access is granted. In SpEL expressions, *Attributes* of the *Subject* or the *Resource* can be used and compared, e.g., "*authentication.id=#resource.ownerId*", which evaluates to *True*, if the users owns the resource.

These expressions can be combined with "and" and "or". In general, the SpEL fulfills the requirements of the application. When using more complex ABAC expressions, SpEL in combination with *PermissionEvaluators* can be used. For that, the expression "*hasPermission*" can be used [5], for each of which the processing *AccessDecisionVoter* calls appropriate *PermissionEvaluators*. Implementing a *PermissionEvaluator* closes the gap between the needs of ABAC and the Spring Security access control implementation. The implementation of the *Permission-Evaluator* interface can access any *Attribute* of the *Subject*, *Resource* and *Environment*.

Spring Security offers the use of *Access Control Lists (ACL)*, which are commonly used to implement IBAC [2]. In [5], the set up of a database, holding the *ACL* and the configuration of Spring Security to use a database, is shown. For each *Resource* an *Access Control Entry* can be added to the database, giving specific *Permissions* to a *Subject*. Built-in permissions are read, write, create, delete and administer. These *Permissions* can be enhanced or replaced [5]. Besides *ACL* and its *Entries*, the protected URLs have to be configured or methods have to be annotated. This is done using the "*hasPermission*" SpEL expression [5].

2) Architectural Patterns

The previous section showed the definition of authorization policies with Spring Security and merely parts of their enforcement. The framework uses a concrete *PEP* for URLs and for method access control respectively. The *PEP* has to handle all requests on a *Protected Object*. A filter (*FilterSecurityInterceptor*) is used to intercept requests on URLs and to control access on the URL. The filter implements the *AbstractSecurityInterceptor*. Thus requests on URLs are handled as described in Section II.A.

Requests on methods are intercepted using the Spring Aspect-Oriented Programming (AOP) [23] feature. The Spring *AnnotationSecurityAspect* enhances security annotated methods. The advice of the aspect redirects method calls to the *AspectMethodSecurityInterceptor*, which is an implementation of the *AbstractSecurityInterceptor* interface, as well.

Thus, requests to URLs and methods are intercepted by the Spring Security framework and processed to enforce access control. The *AuthorizationRules* are described by the *AuthorizationPolicy* that is used. Method annotation and expressions in configuration for URLs describe the concrete *Authorization* for a *Resource*. The *PEP* pattern is used with Spring Security, if the *Authorization* pattern is set up and the *FilterChain* is configured or method security is activated [5].

The *Authorization Enforcer* pattern is the concretion of the *PEP* for Java EE applications. Thus, the mentioned protection of methods and URLs is an implementation of the pattern. The Spring Security *AuthenticationManager* takes the role of the *Authentication Provider* and the several authentication filters as well as the *AuthorizationManager* represent the *Authorization Enforcer* role. Thus, the *Authorization Enforcer* pattern can be implemented by using Spring Security access control. The *Intercepting Web Agent* pattern cannot be applied to the method protection, because the pattern defines application execution after access control. Thus the implementation of the pattern is applied through configuration of the *Authentication Enforcer* pattern, the *Authorization* pattern and a configured URL protection.

E. *Discussion*

The examination of the Spring Security framework revealed support for most known security patterns but failed to offer developers guidance on their implementation. This handicap has been overcome, as the proposed security pattern implementation templates enable the efficient mapping of pattern-based security design in future development processes. Thus, it allows security knowledge reuse as proposed by our security engineering approach described in Section III.

The identification process was thereby laborious as an intensive black box as well as white box examination of the framework was performed. This was only possible due to the excellent documentation and access to the framework's source code, which is not always the case, e.g., with proprietary frameworks, and makes the identification more difficult.

We tried to document the templates as independent of any application context as possible and in the implementation case study, discussed in the next section, we found that the templates are well crafted and suitable. But we do not claim completeness or efficiency. In fact, the templates as well as the pattern to implementation mapping may need to be adjusted to fit a specific context as well as future versions of the framework.

V. IMPLEMENTING CASE STUDY

The knowledge described in the previous sections combined with, e.g., use cases, misuse cases and component diagrams has been applied to the development of the security functionality of a web application. Spring Security was used as the security platform used to protect the application.

TABLE II. SUPPORTED AUTHORIZATION PATTERNS

Authorization Patterns	Spring Security Implementation
Role-Based Access Control	• Hierarchical roles using *GrantedAuthorities*
Identitty-Based Access Control	• Access Control Lists
Attribute-Based Access Control	• Simplified implementation using Spring Expression Language
Authorization Enforcer	• Aspect interceptor for method access
Intercepting Web Agent	• Filter mechanism of Java Servlets for URL access

A. *KITCampusGuide Scenario Descriptions*

The KITCampusGuide application is a navigation tool supporting students, teachers and staff in finding and navigating to points of interest (POI), i.e., any kind of landmark, such as a canteen, an auditorium or offices. Due to restricted areas on the campus and several other requirements, the search for and display of POIs has to be restricted. Users should be able to create private POIs, which can only be seen and modified by themselves. As such, management of POIs is the most relevant to security.

B. *Secure Development of a POI Manager Component*

A POI Management component was developed by modeling the requirements using UML use cases. Security analysis resulted in a need for user authentication and authorization, when creating private POIs. An architectural decision was made to use a single factor authentication using username-password pairs and RBAC for authorization policies. The security functionality is independent from the

```
<user name="student1"... authorities="ROLE_STUDENT" />
<user name="admin" ... authorities="ROLE_ADMIN" />
```

(a) User definition and role assignment

```
<bean id="rightsToRoles"
    class="oss.access.hierarchicalroles.RoleHierarchyImpl">
    <property name="hierarchy">
    ROLE_ADMIN >ROLE_STUDENT
    ROLE_STUDENT > PERM_DELETE_POI
    ...
    </property>
</bean>
```

(b) Role definition and permission assignment

```
@RolesAllowed("PERM_DELETE_POI")
public void delete(PointOfInterest poi) { ... }
```

(c) Configuring access control on a method using annotations

```
<http use-expressions=”true”>
    <intercept-url pattern=”/poi/*/delete”
        access=”hasRole(PERM_DELETE_POI)”/>
</http>
```

(d) Configuring access control on URLs

Figure 4 Implementing Role-Based Access Control in Spring Security (unnecessary information is stripped with “…”)

functional logic and supports access control to restrict access using an IWA. The architecture model was enhanced using the appropriate pattern descriptions.

Using the previously acquired knowledge about security patterns supported by the Spring Security framework, the security functionality was implemented by providing appropriate configurations to the framework and applying annotations to relevant methods. Figure 2 shows the necessary configurations to implement RBAC for a delete operation on POIs. Thereby two roles are defined and assigned to two different users. The role "ROLE_ADMIN" inherits the permissions of the role "ROLE_STUDENT", which in the shown example includes the permission to delete a POI. This is controlled using an annotation for the "delete" method as well as an authorization filter for the URL-based "delete" operation.

C. *Problems and Experiences*

Finding the level of abstraction needed for the application is an important issue during design phase. In the case study the whole development process was traversed by a single person and the application size was manageable. But as the size of the application grows, this could lead to problems. A hierarchy of patterns indicated in the previous chapters would close the gap between a high level of abstraction and a level close to implementation. This is helpful in concretizing the design step by step.

VI. Conclusion and Future Work

In this paper, the open source security framework Spring Security was examined in its support for common security patterns for authentication and authorization. Patterns for RBAC and ABAC as well as for username/password-based authentication were identified and appropriate best-practice implementation templates for Spring Security were provided. These templates can be used as a reference to implement the mentioned patterns in other projects. Further, the benefits of a pattern-based security framework description for a model-driven approach were discussed and its role in a reuse-based security engineering process was briefly explained.

In continuation of this work, the possible security design and implementation decisions need to be captured in flexible variation models to provide a decision support. Also, the relationships between the patterns will be determined and specified to identify mandatory or optional dependencies between the design and implementation patterns. In future research, we focus on completing the different parts of our reuse-based security engineering process.

References

[1] R. J. Anderson, Security Engineering, 2nd ed. Indianapolis, Ind.: Wiley, 2008, p. 1040.

[2] M. Schumacher, E. B. Fernandez, D. Hybertson, F. Buschmann, and P. Sommerlad, Security Patterns. Chichester, England: John Wiley & Sons Ltd, 2005, p. 565.

[3] A. Dikanski and S. Abeck, “Towards a Reuse-oriented Security Engineering for Web-based Applications and Services,” Proc. Seventh International Conference on Internet and Web Applications and Services (ICIW 2012), Stuttgart, June 2012, pp. 282–285.

[4] “Spring Security.” SpringSource Community, p. Apache License, Apr. 2008.

[5] P. Mularien, Spring Security 3. Birmingham, Packt Publishing, 2010.

[6] M. Wiesner, “Introduction to Spring Security 3 /3.1,” SpringOne 2GX. Chicago, Oct.-2010.

[7] N. A. Delessy, E. B. Fernandez, and M. M. Larrondo-Petrie, “A Pattern Language for Identity Management,” International Multi-Conference on Computing in the Global Information Technology, Guadeloupe City, March 2007, pp. 31–31.

[8] E. B. Fernandez, G. Pernul, and M. M. Larrondo-Petrie, “Patterns and Pattern Diagrams for Access Control,” Proc. Trust, Privacy and Security in Digital Business (TrustBus 2008), Turin, Italy, Sept. 2008, pp. 38–47.

[9] C. Steel, R. Nagappan, and R. Lai, Core Security Patterns, 1st ed. Upper Saddle River, N. J.: Prentice Hall International, 2005, p. 1088.

[10] N. A. Delessy and E. B. Fernandez, “Patterns for the eXtensible Access Control Markup Language,” Proc. 12th Pattern Languages of Programs Conference (PLoP2005), Monticello, Illinois, USA, Sept. 2005, pp. 7–10.

[11] E. B. Fernandez, “A Methodology for Secure Software Design,” Proc. International Conference on Software Engineering Research and Practice (SERP’04), pp. 21–24, 2004.

[12] N. A. Delessy and E. B. Fernandez, “A Pattern-Driven Security Process for SOA Applications,” Proc. ACM Symposium on Applied computing, pp. 2226–2227, 2008.

[13] M. Bunke and K. Sohr, “An Architecture-Centric Approach to Detecting Security Patterns in Software,” Proc. Third International Conference on Engineering Secure Software and Systems (ESSoS'11), Madrid, 2011, pp. 156–166.

[14] R. K. Keller, R. Schauer, S. Robitaille, and P. Page, “Pattern-based Reverse-Engineering of Design Components,” Proc. International Conference on Software Engineering, Los Angeles, 1999, pp. 226–235.

[15] T. Lodderstedt, D. Basin, and J. Doser, “SecureUML: A UML-Based Modeling Language for Model-Driven Security,” LNCS, vol. 2460, pp. 426–441, 2002.

[16] M. Alam, R. Breu, and M. Hafner, “Modeling Permissions in a (U/X) ML World,” Proc. First International Conference on Availability, Reliability and Security (ARES), Vienna, April 2006,pp. 685–692.

[17] C. Emig, S. Kreuzer, S. Abeck, J. Biermann, and H. Klarl, “Model-Driven Development of Access Control Policies for Web Services,” Proc. 9th International Conference Software Engineering and Applications (IASTED), vol. 632, pp. 069–165, 2008.

[18] “Java Servlet 2.3 Specifications,” Palo Alto, 53, Sep. 2001, last accessed March 2012.

[19] T. Berners-Lee, R. Fielding, U. C. Irvine, and H. Frystyk, “Hypertext Transfer Protocol (HTTP 1.0),” 1945, May 1996.

[20] E. Yuan, J. Tong, B. Inc, and V. McLean, “Attributed Based Access Control (ABAC) for Web Services,” Proc. IEEE International Conference on Web Services (ICWS), Orlanda, Florida, July 2005.

[21] T. E. Fægri and S. O. Hallenstein, “A Software Product Line Reference Architecture for Security,” in Software Product Lines, no. 8, T. Käkölä and J. C. Dueñas, Eds. Berlin, Heidelberg: Springer, 2006, pp. 276–326.

[22] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, Irvine: University of California, Irvine, 2010.

[23] G. Kiczales, J. Lamping, A. Mendhekar, C. Maeda, C. Lopes, J.-M. Loingtier, and J. Irwin, “Aspect-Oriented Programming,” in Lecture Notes in Computer Science, vol. 1241, no. 10, M. Akşit and S. Matsuoka, Eds. Berlin/Heidelberg: Springer-Verlag, 1997, pp. 220–242.

[24] Oracle, Java Platform, Enterprise Edition (Java EE) 6, <http://www.oracle.com/technetwork/java/javaee/overview/index.html>, last access March 2012.

Securityware: Towards an Architecture for User-Centric Approach

Ali Hammami, Noëmie Simoni
TELECOM ParisTech - LTCI - UMR 5141 CNRS
46, rue Barrault F 75634 Paris Cedex 13 - France
e-mail: ali.hammami—noemie.simoni@telecom-paristech.fr

***Abstract*— With the rapid evolution of next generation networks, telecommunication field are evolving towards user-centric approach. In fact, users demand the access to any service without any technical, temporal or geographical barrier. This approach permits user to personalize their services and have a unique, dynamic and seamless session despite heterogeneity and mobility of his environment. In this context, many security issues arise. In order to overcome security challenges while respecting user-centric requirements, we propose in this paper a new security architecture that is based on a set of service components. These components ensure a secured and simplified service access in a unique and seamless session. They should be reusable, interoperable, autonomous, mutualizable, interconnected and self-managed in order to participate in a secure, personalized and dynamic service composition. We describe at the end of this paper an application scenario and show the feasibility of our proposition.**

Keywords-User-centric; Security as a service; Security architecture; Secure dynamic service composition; Secure seamless and unique session.

I. INTRODUCTION

Nowadays, end-users demand to be constantly connected anywhere, anytime and anyhow. They want to have a dynamic session according to their preferences and the desired QoS (Quality of Service). In this user-centric context, end-users require a unique, personalized and secure session. Furthermore, this session should permit them to access their services in a continuous and seamless way. And, it should offer them an end-to-end QoS. The main goal of this user-centric approach is to make the whole system serve the user without any constraints such as treatment and hardware constraints (system-centric), application constraints (application-centric) or connection constraints (network-centric).

Moreover, with the emergence of next-generation networks, heterogeneous environment (multiple terminals, access networks, operators and service providers) and mobility requirements (user, terminal, network and service mobility) present two challenges that face to ensure a user-centric session. Thus, user-centric solutions should harmonize service personalization, heterogeneity and mobility aspects.

In this context, security represents an important concern. In fact, this user-centric session should obviously be secured. However, many challenges arise. In such environment, system boundaries, which were well delimited, become increasingly open. Indeed, there are multiple services which are unknown in advance and multiple communications between services and with users.

Besides, heterogeneity of involved resources (terminals, networks and services) in the user session increases the complexity of security tasks. In addition, the different types of mobility (user, terminal, network and service mobility) affect the user-centric session that should be unique, secure and seamless and ensure continuity of services. Mobility impacts strongly on an end-to-end secured service delivery.

The main questions we need to answer in this step are: How security should be conceived to be efficient in service-oriented environments? How can security support a personalized and flexible service composition for the user who desires a unique and seamless session? How can we have a secure user-centric session in a dynamic and mobile context?

This open, heterogeneous and mobile environment presents significant risks in terms of security and becomes increasingly vulnerable. Therefore, a new security solution, which responds to user-centric requirements in NGN environment, should be provided. For this purpose, we propose a novel security architecture that respects user-centric approach, called Securityware. We adopt new criteria in Service Oriented Architecture (SOA) which ensure a dynamic and seamless service composition. This architecture is organized in four visibility levels (equipment, network, service and user) and each resource (terminal, network and service) is considered as a service component. Our solution does not afford security mechanisms in a static and centralized way. Security is provided as a service due to a set of service components (Securityware).

The remainder of this paper is organized as follows. In Section 2, we discuss some existing approaches regarding security as a service. In Section 3, we present our proposed security solution which is based on security as a service approach. First, we describe the major security service characteristics that are based not only on simple SOA principles, but also on new features to provide a secure, personalized, dynamic and seamless service composition. Second, we detail the components of our security architecture. Third, we describe how to ensure a secure service composition in user-centric context. In Section 4, we present an application scenario and show the feasibility of our proposition. Finally, conclusion and perspectives for future work are presented in Section 5.

II. RELATED WORK

Before presenting our proposal, we analyze the content of some related technologies and research works, which deal with security in Service Oriented Architecture. We will focus on existing approaches related to security architectures, and we will discuss important research activities on the notion of security services.

P. Qi-rui et al. propose in [1] a unified authentication and authorization solution which supports the SOA-based distributed systems. This proposal is based on Service-Access-Agent architecture. It uses a mechanism of two authentication levels to separate in-domain and inter-domain authentication for more efficiency and configuration simplicity. The general framework of this solution is composed of GAAC (Global Authentication and Authorization Centre), SAA (Service Access Agent) and Web services. GAAC ensures inter-domain authentication by authenticating SAAs and provides centralized access authorization control to Web services. SAA is responsible to control and manage the Web services within local security domain. This solution provides authentication and authorization at the service level. This is an interesting contribution but the centralized adopted approach is monolithic and vertical and has a strong coupling. This raises the following questions: Is this sufficient in our cross-organizational and ubiquitous context? How can we have more flexibility, dynamicity and extensibility in security functions?

E. Bertino et al. [2] propose a reference architectural framework based on security services such as identification, authentication and authorization services. They discuss how each component of the security pipeline is considered as a service according to SOA principles. They discuss also mechanisms for coordinating different security services. Then, an event-based approach is used to disseminate relevant security events to all the potentially affected services. To this end, a notification service is integrated in the architectural framework to notify security services of relevant events. This work deals with security as a service approach which is particularly promising for Service Oriented Architecture. Nevertheless, it does not specify how security functions could be ensured in a transparent and simplified way to user in a dynamic and user-centric service composition. This raises the question: How to provide a secure and seamless session in a dynamic environment according to user requirements and preferences?

The service-oriented security architecture which is described in [3] and [4] consists of three layers. The first layer contains the well-defined and stable service interfaces to be used by other services. The authentication interface provides operations to authenticate a user (entering username and password) and generates a temporary security token. An access control decision can be delegated to the authorization verification service interface. In functional layer, the secure token service validates the claimed identity of the service consumer. A policy decision point (PDP) evaluates policies and makes authorization decision. These functional components store and retrieve their data from components placed in the data layer below. This work focuses only on interoperability criterion while many others criteria should be specified in order to satisfy new approaches such as user-centric approach that requires personalization, dynamicity, and transparency within a secure and seamless session.

All the cited works above present the good efforts to provide a conceptual views and approaches about security as a service to be managed in a Service Oriented Architecture. According to SOA, the service layer is like a big service pool with components of different functions. This approach brings more flexibility and allows SPs to respond more quickly to users requirements. Could the current solutions actually ensure flexibility if they guard vertical constraint between client and server and if we want to take into account mobility effects? We think that is not so because it is missing some criteria such as mutualization and self-management to have effective flexibility and dynamicity. In fact, mutualization consists of having not only reusable but also shareable components in different contexts of different users. Then, security components should be mutualizable to be used in every context by many users without any constraints. Current SOA approach does not specify also self-management criterion of service components that permits to have a seamless secure session with the desired level of security and QoS requirements despite mobility and changes due to dynamic service composition.

Meanwhile, service composition only on the service layer is not enough. Nowadays, with the existence of heterogeneous networks and terminals, which can greatly affect user experiences, we should opt for a new perception able to homogenize the different resources; the components of the network layer as well as the terminal layer should be perceived as services. Thereby, the security components should consider the terminal as a service. We demonstrate, then, in our proposals, how we can manage and ensure the security and continuity of user session.

III. PROPOSITION

The challenge is how to guarantee security, continuity and uniqueness of end-users sessions in a user-oriented and dynamic context. To overcome this problem, we present in this Section a security solution which is based on security as a service approach. These security service components should satisfy a set of criteria to be used in a dynamic service composition (Section 3.1). Then, we detail the components of our security architecture (Section 3.2). Finally, we describe how to ensure a secure service composition in user-centric context (Section 3.3).

A. Security service component characteristics

The security services should have basically the same criteria as specified in SOA. The characteristics of our security services are based not only on simple SOA principles but also on new features to provide a secure, personalized, dynamic and seamless service composition.

A security component is an element of the system that performs a predefined service and is able to communicate with other components. Among the basic characteristics, a security service component should be:

- *Reusable*: it must be a unit of reuse. An adaptation phase may be necessary depending on the execution context.
- *Autonomous*: this feature means that a service component is functionally independent. It is self-sufficient and it is able to perform its functionalities without needing another component.

In order to ensure flexibility, dynamic service composition and personalization regarding the user-centric context, our proposition is not limited to SOA characteristics. On the contrary, the added value of our solution consists in adding the following criteria:

- Mutualizable: this feature consists of sharing, simultaneously and between several users, resources that a service can offer. This permits to optimize use of these resources. This aspect requires that all constraints related to the preservation of the service state (activation, deactivation, etc.) or to particular user data be eliminated. This means that services should be stateless. Therefore, a service should perform generic tasks to all users without considering their contexts and maintaining their specific data. This criterion helps to have a dynamic replacement of a service in a loosely coupled service composition.
- Interconnected: this feature is just added to the interoperability one and ensures thereby interoperation between several service components in order to create a global service. This approach enables the cooperation of service components while avoiding treatment conflicts and deadlocks. Interconnection between services leads to define imperatively generic interfaces and links.
- Self-managed: this feature is set by the role of the QoS agent that is integrated on each service component to supervise and control its own QoS parameters. It permits to verify whether a service behavior respects the QoS agreement.

B. Security Architecture

To ensure security during user-centric session, it is necessary to apply security mechanisms on all service components involved in this session. For this purpose, we conceive all mechanisms as services. Thus, these services should be designed with specific features as mutualization, autonomy, interoperability and self-management. Moreover, we consider elements that constitute the different layers (equipments, networks and applicative-services) as service components.

In our context, the user-centric session is a juncture of the different session parts. In order to ensure security, continuity and uniqueness of the session and to best answer the end-user's preferences, we propose a security architecture whose components are involved in the different phases of session establishment. First, in the terminal access phase, the terminal must be able to authenticate the user as the legitimate owner or user of the device. This phase is ensured by services offered by the Userware in the terminal. Second, in the network access phase, most possible attacks should be identified because remote platforms are trying to be connected across mistrusted actors. To overcome the locks of trust, we use encryption and authentication mechanisms. The final phase involves the service (application) access. Regarding user rights, authorization service performs secure access to the required services.

The Next Generation Network/Next Generation Service (NGN/NGS) context (convergence, heterogeneity, mobility, mutualization, etc.) requires new mechanisms and techniques which can simplify the different layer access. Thereby, we adopt a distributed architecture that guarantees a loose coupling in order to overcome constraints arising in centralized and monolithic architectures. Moreover, our architecture is based on the two key concepts: security as a service and visibility levels. Then, due to a set of security components, we assure to the user a secured and simplified access to his services in the terminal level and the service level.

Our proposed security architecture, as shown in Figure 1, is based on three main components: the Securityware, the Security Agent and the Security Data store.

The Security Service Provider (Securityware) is responsible of security management and control. Thus, it gathers all the security components as following:

- *Identification Service*: it enables the recognition of a user by the system. It provides an identifier for each user to be recognized by the system. According to his roles, preferences, or service providers, user can have one or more identifier types;
- *Authentication Service*: it determines if an identity is actually what it claims to be. It aims to authenticate user only one time per session (unique authentication) for all requested services;
- *Authorization Service*: it occurs when a user requests access to a service to allow (or deny) him to use service component. Indeed, authorization service evaluates the effective rights. Permission is granted according to rights associated to the user role;
- *Token Service*: it generates and updates the token that permits a unique authentication and ensures consequently a continuous session. The token is regenerated if the session lasts a long time to prevent from session hijacking attempts.
- *Session Service*: it ensures the session creation and activation and the session identifier (Session_ID) generation (upon successful authentication). This service is responsible for managing and maintaining an end-to-end secure session;
- *VPDN and VPSN Services*: the VPDN (Virtual Private Device Network) service manages the set of user terminals while considering each terminal as a service component. The VPSN (Virtual Private Service Network) service manages the network of services which interact together to offer the required global service. These services generate, respectively, the VPDN and VPSN identifiers. These identifiers are kept unique during the session, despite token regeneration, to maintain a unique session.

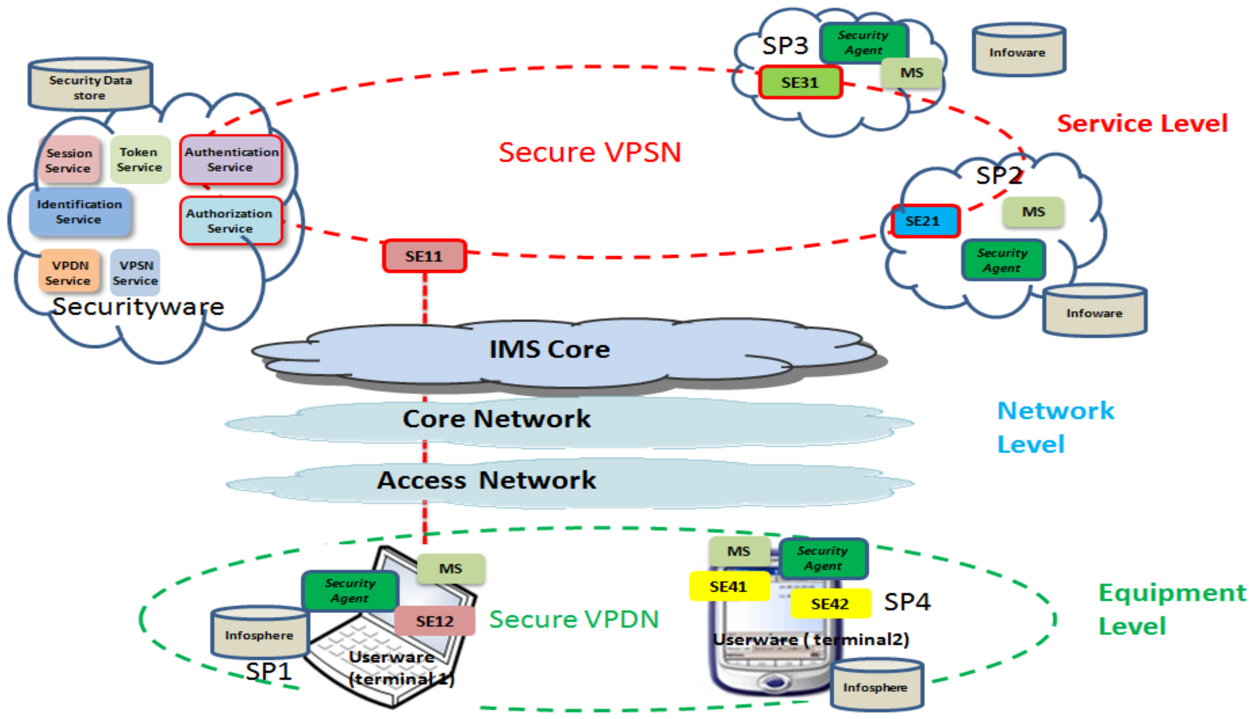


Figure 1. Security Architecture.

In order to facilitate and simplify the management of security and access control at the service and terminal levels, we propose also a Security Agent that is deployed in each service platform and terminal which is considered simultaneously as a service platform and as a service. The Security Agent intercepts requests for the protected resource (service or terminal). It sends then a request to the Securityware asking for decisions regarding the required services. So, the Security Agent secures access to service platforms by checking authorizations due to the Securityware.

Finally, the Security Datastore contains all the information needed by the Securityware, namely, user profile, device profile, VPDN profile, VPSN profile, session profile and service profile.

Our security architectural model will permit a secure vertical aggregation between terminal or equipment components and applicative-service components. Actually, the vertical view represents the user's session that must respect the User-Centric approach. The main objective of this approach is to make the whole system at the service of user unlike other approaches where user must comply with various constraints related to connection (Network Centric) or to treatment (Application Centric). So, the session must be dynamically established according to the user preferences and to services that can be offered by its environment during his movement. We should secure this session without complicating the task to the user each time he wants to add a service to his session.

In fact, our security provider regroups a collection of security components such as identification, authentication and authorization. It represents the Securityware that permits to manage the security of a set of service platforms including the terminal. We deploy also Security Agents that ensure a vertical aggregation of service and terminal levels involved in the session.

On the other hand, our security architectural model will also provide a secure horizontal composition between service components at the service level as the terminal level. Indeed, in a horizontal view, the Securityware and the Security Agent intervene during the service composition to build the VPSN from terminal or applicative services while ensuring a secure access to these services.

C. A Secure Service composition

In order to meet the user-centric needs, to ensure all kinds of service personalization and to follow the user's mobility, we must conceive services and manage and secure session otherwise. Thus, our proposed architecture offers a secured service composition to end-users, with dynamic changes following mobility, QoS and security requirements and depending on available service offers.

A secure service composition in service layer represents a VPSN. It includes the set of service components that are used by a user along his session which is based on a user-centric approach. So, services are composed dynamically according to user preferences. VPSN manages interaction between these components via links while respecting a logic of service. Our security services are among management services in the VPSN that are transparent to the user and aim to secure his session.

During a User Centric session, the user is asked to dynamically compose services he needs for a period of time. For example, he can add a videophone service to its voice call or transfer a TV program from his TV to his PDA when he moves. All these changes are ensured in a seamless way within a unique and secure session. Security is provided by a set of service components that we find in a specialized platform (service provider), called Securityware. We consider in our solution two key points: first, the Userware (terminal) is considered as any service platform (service provider); second, in each service platform, a Security Agent is deployed to manage its security. We describe below how to have a secure service composition within a Virtual Private Service Network (Figure 2).

1) *Session Initialization:*

When establishing a user session via his terminal, the Userware (Security Agent) sends a request to the Securityware to activate the identification service and then the authentication service in order to validate the user's credentials saved in his profile (Figure 2: (1), (2) and (3)). Once the user is successfully identified and authenticated, session initialization is performed by the Session Service. Then, a token is generated by the Token Service. It contains the necessary session and user information. Afterward, the token is transmitted by the Securityware to the Security Agent in the terminal.

2) *VPDN Creation:*

Next, we have to verify that the user is authorized to use this terminal and to create a Virtual Private Device Network VPDN [8]. So, an authorization request regarding the use of terminal is sent by the Security Agent to the Securityware. The Authorization Service evaluates the effective rights considering information provided by the Authentication Service. Once the user is authorized, the VPDN Service generates the VPDN-ID that is the identifier of the set of terminals deployed by the user during his session forming the VPDN. The Token Service retrieves the VPDN identifier and updates the token. The terminal is registered in the VPDN profile (Figure 2: (4) and (5)).

3) *VPSN Creation:*

Now, the user wants to create his VPSN via his terminal. For that the Security Agent checks user's rights (user's authorizations) with the Securityware asking for decision regarding the VPSN creation. When the Securityware grants a positive response, the VPSN Service generates a VPSN-ID and the Token Service updates the token. This latter contains then all the attributes related to the VPSN, the VPDN, the session and the user. Consequently, the service session is initiated by the terminal, which is considered as a service platform, after the authorization and the VPSN-ID generation (Figure 2: (6) and (7)).

Thereafter, the user is able to compose exposable services that he wishes to have in his session according to his preferences and needs. He can select these services from a service catalog stored in his profile or after a discovery of services that are available in his environment. Then, the terminal generates a service composition workflow and a logic of service based on the selected exposable services. Next, this information are sent via the signaling protocol SIP+ (Session Initiation Protocol) [9] in a SIP+ Invite message passing through IMS (IP Multimedia Subsystem) to the service platforms called Serviceware. The SIP+ Invite message body contains also the token that is transferred to each security agent that protects a service platform visited by the user in order to ensure a single authentication during user session (Figure 2: (8) and (11)).

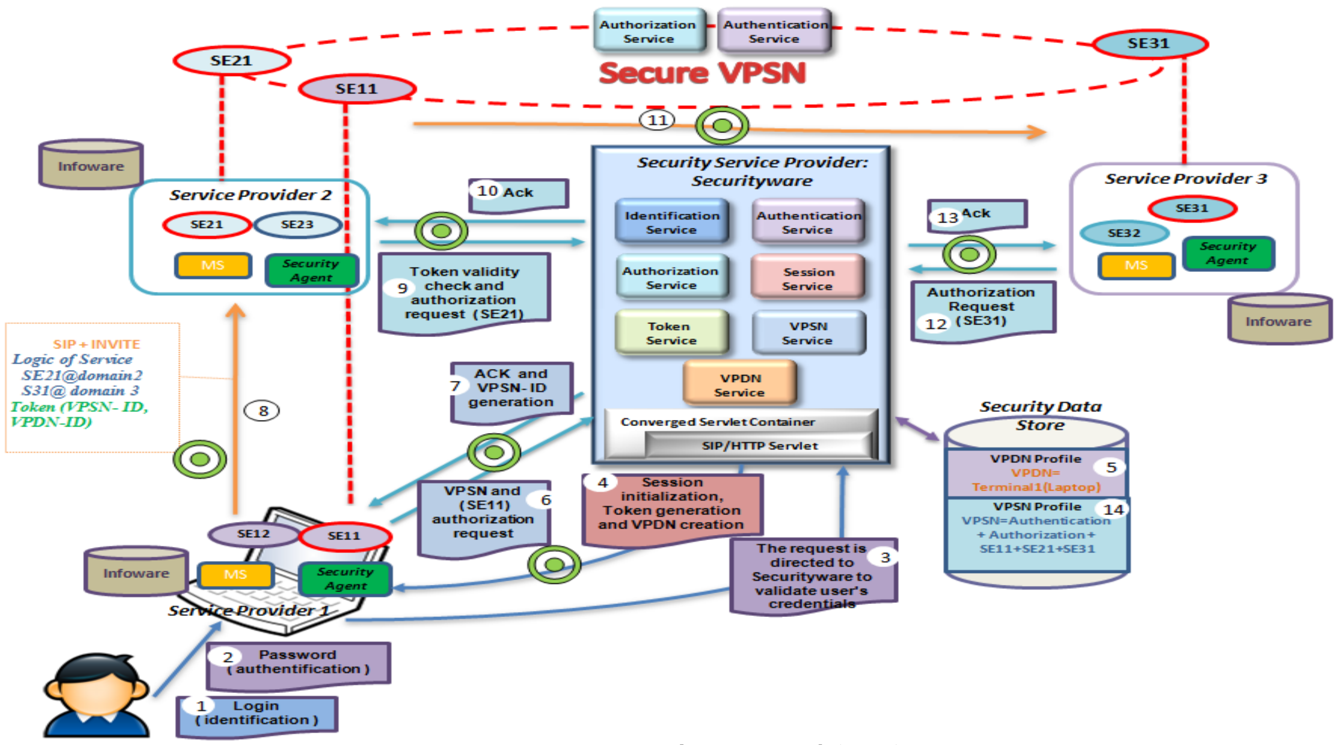


Figure 2. A Secure Virtual Private Network (VPSN).

At the reception of this message, the Serviceware translates exposable services into basic services and builds the VPSN. The Authorization service is included in this VPSN as a management service to perform access authorization to basic services. It interacts with the security agents of the different requested service platforms to verify if privileges related to the desired services correspond to user rights related to his role. In each passage from one platform to another, the security agent checks the validity of the token (Figure 2: (9), (10), (12) and (13)).

The Authentication service is also included in the VPSN in case of a higher authentication is required. For example, when one of selected services in the VPSN is a banking service (payment), a strong authentication is required.

The authentication and authorization components are basic management services which participate in pre-provisioning and provisioning of the services that compose a VPSN. These services perform their roles transparently, without user intervention. Finally, the created VPSN is saved in the VPSN profile (Figure 2: (14)).

IV. FEASABILITY

A. Scenario

For more clarity, we present in this section an application scenario that explains how our security service architecture guarantees security, seamlessness, dynamicity and uniqueness of the end-user session. To achieve automated and distributed control and management of security, we will apply our proposed vision of security as a service called Securityware based on generic, mutualizable, stateless and self-managed service components.

This scenario, as shown in Figure 3, takes place in a cross-organizational context that has multiple service platforms (terminal is also considered as a service platform). Therefore, the creation of VPSN is shared between the different involved platforms.

Considering the fact that our mobile and heterogeneous environment exposes significant risks in terms of security, it becomes increasingly vulnerable. For this reason, our scenario brings out the case when a service becomes unavailable following a malicious attack or a security fail (denial of service) in a user session.

Then, let us describe this scenario: Bob is at home. He uses his laptop. According to his needs and his preferences specified in his contract, he wants to display a movie with high resolution (SE11: Service Element 11), check his emails (SE21) and receive all SMS messages as Voice Messages (SE31).

1) *A secure service composition:*

As described in subsection 3-C, a secure service composition is performed and a VPSN is provisioned for this user. It contains all services that respond to Bob's request according to his preferences and the required level of security. We note that SE11 is provided by the service provider SP1 (Userware: terminal), SE21 is provided by the service provider SP2, SE31 is provided by the service provider SP3, and the security services are provided by the Securityware.

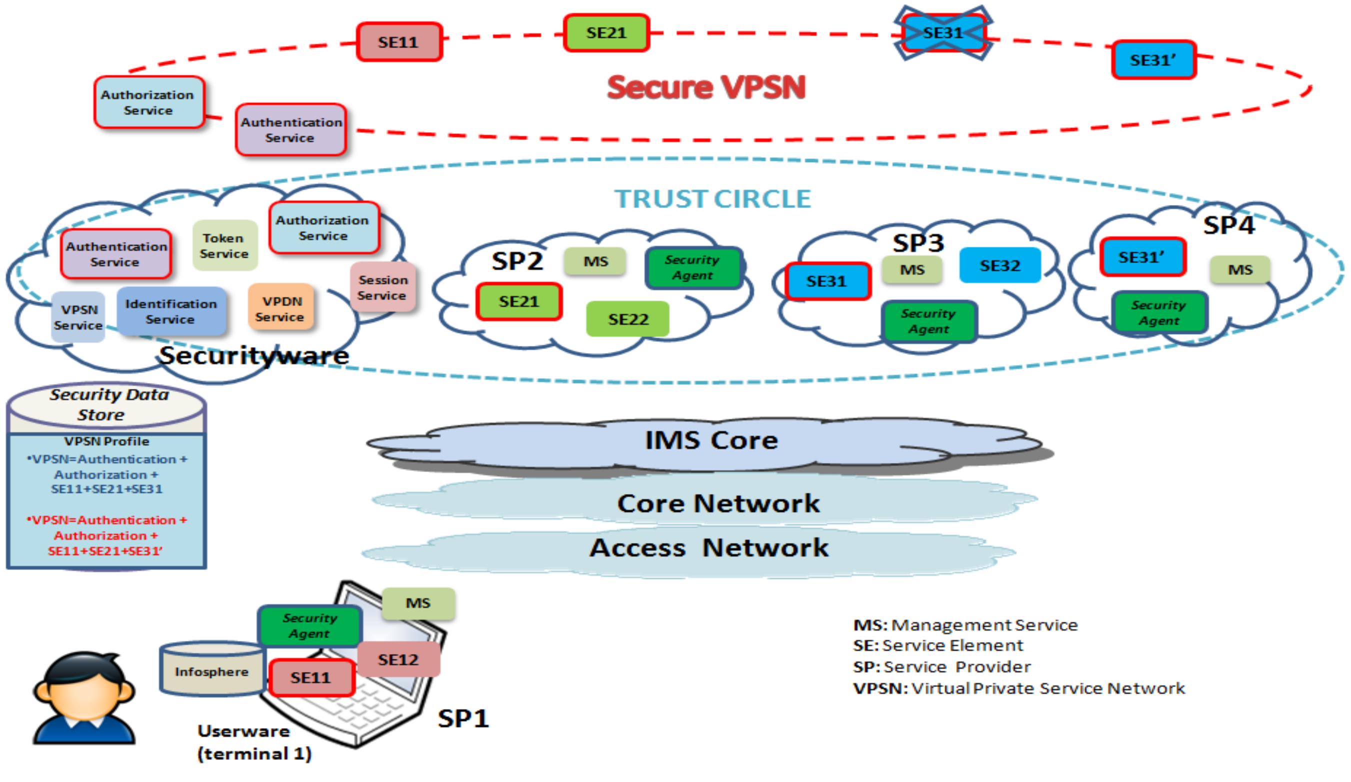


Figure 3. Scenario.

These five services are linked in the VPSN and executed according to a logic of service that presents a workflow of the chosen services and indicates in which order they should be executed.

VPSN (Bob) = Authentication Service + Authorization Service + SE11 + SE21 + SE31

The security services are imperatively interconnected and interoperable with other services in order to provide a secure global service. These security services interact also with Security Agents in each passage from one service platform to another to ensure a secure access to these service platforms using the token.

2) A continuous secure session:

Every change in the user's context as well as the changes related to a service component (availability, malicious attack, security fail) is recovered by adding, removing or replacing one or more component to the virtual service network (VPSN). To have an automated and decentralized recovery and service management, each service component is self-managed. This solution permits to detect the failure at the right time and to provision service resources without causing interruption during service use due to a periodic service discovery that maintains a set of ubiquitous services that are potentially equivalent to services involved in the active VPSN. This discovery is launched in all the federated service platforms which are in the same circle of trust. In fact, the search is based on the three following criteria:

i) Services which belong to the circle of trust,

ii) Services which have the needed functionality, and

iii) Services to which user have access authorization based on his role and privileges.

Thereby, a secure VPSN and a continuous end-user session are maintained.

For example, we suppose that SE31 undergoes a security fail (malicious attack). Thus, a new composition of services is needed to recover this fail. The unavailable component, which has been attacked, should be replaced by another that is functionally equivalent and able to provide the required security level. Consequently, the service SE31 is replaced by SE31', which has been already discovered. It belongs to the trusted service provider SP4.

As a result, a new service composition is established and the logic of service becomes as follows:

VPSN (Bob) = Authentication Service + Authorization Service + SE11 + SE21 + SE31'

3) Mutualizable, generic and stateless security services:

It can be proven that the same service component can be mutualized among different users and can be used in different contexts through the genericity of the service component. For example, the same authentication service resource can be shared between two users in two different contexts: the first user needs a strong authentication for banking transactions and the second user needs a simple authentication to watch a movie. This is possible if service components, particularly security services, are stateless. This means that service components should not require a state or information relative to clients when they are invoked. This feature leads us to have ubiquitous services that can be deployed in any environment. It also makes possible the use of different services provided by different service providers. Therefore, we can switch from a service to another even if they do not belong to the same SP or are not deployed on the same platform which makes the service composition more efficient and flexible.

B. Implementations

In order to prove the feasibility and to validate our proposition, we use our UBIS ("User centric": uBiquity and Integration of Services) project platform that contains principally Open IMS Cores and SailFin application server. The former ensures IMS session control and the latter represents control and applicative service container. We explain below how we implement and deploy our different security service components.

We use Java language to write the proposed security architecture components, namely, Securityware and Security Agent. For our Securityware, we use EJB (Enterprise JavaBeans) technology [10] to develop autonomous, loosely connected and stateless services. These services are deployed in SailFin [11] Application Server that supports various APIs (Application Programming Interfaces) such as JMS (Java Message Service), JNDI (Java Naming and Directory Interface), JDBC (Java DataBase Connectivity) and Sip servlet. Our proposed Securityware extends actually some security parts involved in OpenSSO project [12]. For our Security Agents, they are deployed in each terminal or service platform. In order to support SIP+, we deploy then Converged Application Container which is composed of SIP and HTTP (Hypertext Transfer Protocol) servlets and will permits to switch from HTTP to SIP. We note that all transactions between different components are secured using SSL (Secure Socket Layer) protocol. Our Security Datastore is an LDAP (Lightweight Directory Access Protocol) directory (OpenDS) that contains our used information and is connected to the Securityware.

Figure 4 describes how our security service provider Securityware is implemented.

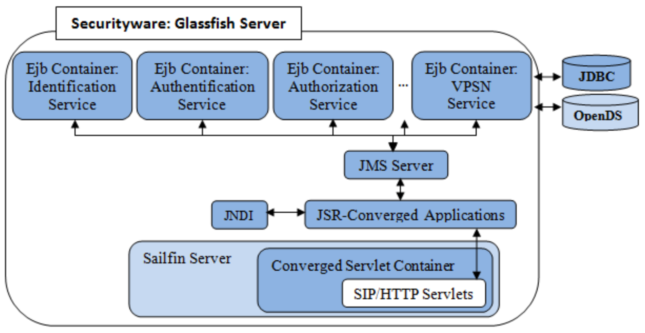

Securityware: Glassfish Server
Ejb Container: Identification Service
Ejb Container: Authentification Service
Ejb Container: Authorization Service
...
Ejb Container: VPSN Service
JDBC
OpenDS
JMS Server
JNDI
JSR-Converged Applications
Sailfin Server
Converged Servlet Container
SIP/HTTP Servlets

Figure 4. Securityware.

V. CONCLUSION

In this paper, we have presented our service-oriented security architecture solution, which is promising in dynamic, mobile and cross-organizational environment. We demonstrated how security can be provided as a service ensured by a set of security components offering the user a secured and a seamless access to his services. The security services are specified respecting a set of criteria, namely mutualization, autonomy, interoperability and self-management. These services intervene transparently to offer a secure and dynamic service composition within a unique and continuous session. To support our proposition, we have described an application scenario which explains the major contributions of our security solution, and we have proven its feasibility in practices. Future efforts will go into the design and implementation of federation audit and trust services.

ACKNOWLEDGMENT

The authors would like to thank all the participants in the UBIS project, financed by the French ANR VERSO 2008 in which is situated this work.

REFERENCES

[1] P. Qi-rui, W. Cheng, W. Jing, L. Jun, L. Qing, and S. Beien. An authentication and authorization solution supporting soa-based distributed systems. In Software Engineering and Service Sciences (ICSESS), 2010 IEEE International Conference on, pages 535–538. IEEE, 2010.

[2] E. Bertino and L.D. Martino. A service-oriented approach to security–concepts and issues. In Future Trends of Distributed Computing Systems, 2007. FTDCS'07. 11th IEEE International Workshop on, pages 31–40. IEEE, 2007.

[3] C. Emig, F. Brandt, S. Kreuzer, and S. Abeck. Identity as a service-towards a service-oriented identity management architecture. In Proceedings of the 13th open European summer school and IFIP TC6. 6 conference on Dependable and adaptable networks and services, pages 1–8. SpringerVerlag, 2007.

[4] C. Emig, F. Brandt, S. Abeck, J. Biermann, and H. Klarl. An access control metamodel for web service-oriented architecture. In Software Engineering Advances, 2007. ICSEA 2007. International Conference on, pages 57–57. IEEE, 2007.

[5] R. Kanneganti and P. Chodavarapu. Soa security. 2008.

[6] Z.B. Daho and N. Simoni. Towards dynamic virtual private service networks: Design and self-management. In Network Operations and Management Symposium, 2006. NOMS 2006. 10th IEEE/IFIP, pages 1–4. IEEE, 2006.

[7] C. Phan. Service oriented architecture (soa)-security challenges and mitigation strategies. In Military Communications Conference, 2007. MILCOM 2007. IEEE, pages 1–7. IEEE, 2007.

[8] A. Hammami and N. Simoni. Secure and seamless session management in mobile and heteregenous environment. In SECRYPT, ROME, 2012.

[9] H. Alaoui, P. Coude, and N. Simoni. User-centric and QoS-based Service Session. In ASPCC, Korea, 2011.

[10] http://www.oracle.com/technetwork/java/javaee/ejb/index.html

[11] Sailfin Project. https://sailfin.dev.java.net/.

[12] OpenSSO Project. https://opensso.dev.java.net/.

Realtime-Aware Control Unit Network Attachment Module

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

Abstract—**Information security is gaining increasingly more importance for real-time industrial automation networks. Protection is not only needed against attacks originating from external networks connected as remote data access, but also from potential attacks originating from locally connected devices connected during regular operation. Relevant use cases comprise local service access and plug-and-play of automation components. A conventional firewall filtering is not sufficient as it filters allowed network traffic depending on the contents of data packets, but it is not aware of specific real-time restrictions that have to be obeyed within an industrial automation network. For these reasons, a solution is required allowing to connect uncontrolled resp. untrusted devices with a real-time automation network segment in a secure way. This paper describes the general concept for a network filtering device that supports real-time criteria for making filtering decisions.**

Keywords-security; device authentication; real-time; network access authentication; firewall.

I. INTRODUCTION

Standard communication technologies as Ethernet and Internet Protocol IP are increasingly used in industrial environments down to the field level. Guaranteed real-time communication plays an essential role for many industrial control applications. In contrast to audio-video communication, industrial real-time requires that the real-time constraints are obeyed under all conditions as intermediate disturbances or delays are not acceptable. Different kinds of industrial automation may be distinguished, e.g., process automation for running a chemical process or a power plant, factory automation, e.g., a conveyor belt production line, or energy automation for the energy grid. For these time-critical types of applications, different real-time variants of Ethernet are available, see [1] for an overview. A basic mechanism is to reserve time slots for real-time traffic that is not allowed to be used for non-real-time traffic.

The correct operation of a shared real-time network segment depends on the fact that all attached network devices respect the real-time medium access protocol. Different usage scenarios require, however, that uncontrolled network devices are connected to a real time Ethernet segment. A network firewall filtering the data traffic according to defined filter rules depending on data content of frames/packets does solve the problem only partly as an uncontrolled device cannot be trusted to support or obey correctly the real-time constraints valid for the real-time network segment. Exemplary usage scenarios for direct connection of an uncontrolled device to a real-time field level network are maintenance and diagnosis, where a service notebook may be connected temporarily during operation. Also, standard network equipments, like a file server (NAS / network attached storage), video camera, a data recorder etc. shall be integrated within a real-time segment in a secure way.

This paper presents a firewall-like network traffic filtering security device that performs a time-dependent filtering of data traffic such that access to the communication medium is prevented during time-periods reserved for real-time traffic. It provides a control measure to enforce network access policy for a real-time field level industrial control network segment.

The remainder of this paper is structured as follows: Section II provides an overview on real-time control networks, depicting different topology types. Section III describes the problem statement and the solution approach for a real-time network attachment module allowing phasing in no-real-time capable device into real-time networks. Section IV gives an overview about related work and Section V concludes the paper and provides an outlook.

II. REALTIME CONTROL NETWORKS

Real-time systems typically consist of hardware and software that are subject to time constraints regarding execution of commands. This comprises the initiation of a command, the execution itself and the acknowledgement of the execution. Real-time in the context of this paper refers to systems with a deterministic behavior, resulting in a predictable maximum response time. These systems will handle all events at appropriate (context-dependent) speed, without loss of events.

In industry environments, especially in automation systems (robotics, motor control systems), there exist hard real-time constraints. Here, high performance system and network technologies are required to cope with the timing requirements [1]. Characteristic features of devices used in high performance communication networks are short communication bus cycle times, low device latency and high update rates of inputs and outputs of network nodes. To support strictest real-time requirements, industrial systems often apply an isochronous communication bus with very low bus cycle jitter. The process control systems synchronize measurement, control, actuation and communication, taking advantage of the isochronous bus, and providing a fix schedule for process control. Real-time requirements vary from response times < 100 ms for video and voice over IP (VoIP) to response times in the range of µseconds for motion control applications.

Characteristic for this type of networks is the engineering phase, in which all systems are configured according to their responsibility in the overall system for the support of a dedicated task. Besides the engineering phase there is also a maintenance phase, in which administrative tasks like Firmware updates or similar can be applied to the system components. This phase is typically scheduled to consider its impact in the common workflow.

History has shown that these maintenance windows may not be sufficient and that there is need for administrative actions, e.g., due to defect field devices or the need to measure certain data, e.g., during the production cycle. Especially if there are dynamic error situations in the network that need to be tracked over a certain period of time or detected security vulnerabilities, which may be leveraged by malware, it may be necessary to introduce an additional component in the network, like a service component. It is very likely, that such a service component may not cope with the strict real-time requirements of the rest of the system. Nevertheless, the introduction of these service components shall not interfere with the real-time communication of the original industrial system.

The next subsections will provide more insight into typical network architectures used in industrial automation environments, which build the base for the problem and solution description in the next chapter.

A. *Network Architecture*

Industrial networks as used e.g., for energy automation are typically shared networks connected in a ring, star, or bus topology or a mixture of these [2]. Most often, the time critical part is performed on a dedicated network, while the rest of the communication supporting the industrial systems is performed on networks with lower performance or latency requirements. An example may be the connection of the industrial network to the office network. The real-time specific part is contained in a so called industrial automation cell as shown in Figure 1. The cell connects to the outside world through the connectivity module, which may be a switch or similar. To achieve security, this connectivity module often features a firewall to filter incoming and outgoing data communication. When, however, uncontrolled equipment is connected directly to a real-time network segment, e.g., for maintenance or diagnosis, or when plug and play of automation equipment is supported, the filtering firewall at the border of the automation cell is not sufficient when it cannot be excluded that the connected device may be compromised or malicious.

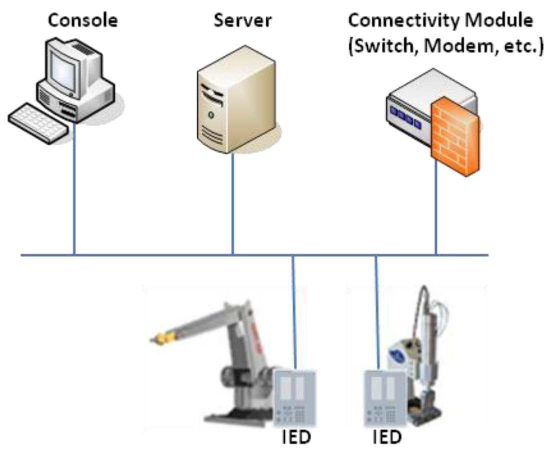


Figure 1. Example for an industrial automation cell

A bus topology connects IEDs (Intelligent Electronic Devices) via a shared communication line. There may be no further switches or hubs involved. The bus topology typically utilizes IEEE 803.3 based Carrier Sense Multiple Access with Collision Detection (CSMA/CD) mechanism [3]. Protocols deployed in this setup are for instance Profinet or Ethernet in use cases like process automation. The IEDs may belong to more complex systems like robots or melding machines. More complex setups may also provide a reason to use a substructure of the network, e.g., a robot may communicate with the outside via one dedicated connection switch, while the robot itself is build using several IEDs thus building its own automation network or automation cell. This is depicted in the following Figure 2.

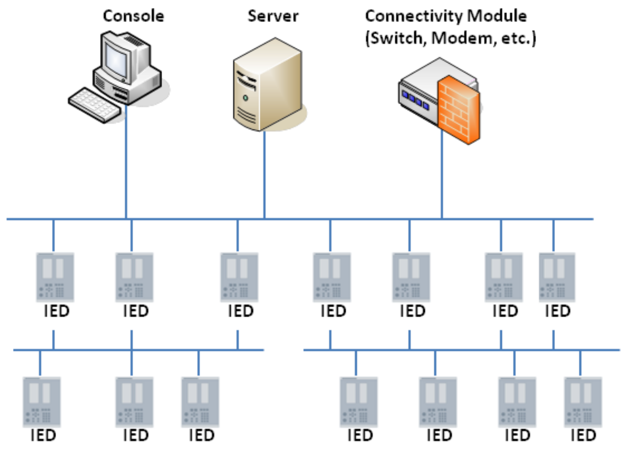


Figure 2. Bus Topology in an Industrial Network

Besides the shown bus topology, there are further network topologies used in industrial communication cells like ring or star topologies as stated above. The selection depends on the automation task to be performed. Common to all of them is the strict separation of real-time and non real-time traffic.

Besides these pure topologies, also mixture resulting in combinations of star and bus or star and ring are deployed in the field.

Automation network cells often realize the concept of a security cell. Here, a plant network is divided into so called security cells, which span a dedicated part of the network belonging to a distinguished administrative domain and most importantly, belong to a physically separated part of the network in which real-time requirements have to be met. A firewall at the border of a security cell does not address the case when uncontrolled and potentially corrupted or malicious devices are connected to a real-time network segment directly.

B. *Real-time Communication*

A real-time system can be described as showing deterministic behavior, meaning that the maximum response time of a system can be predicted and guaranteed. To describe the latency resulting from communication, the transfer time is typically used, which includes the communication processing on the sending and the receiving node as well as the network transport time. Another property often used is the cycle time, which describes the time need to complete a defined set of actions from one start to the next.

For real-time communication there are several protocols on different layers of the OSI communication stack defined, which provide the functionality stated above. In industrial communication this is often realized using field busses. For field bus communication there exist a complete family of protocols, which has been standardized within ISO-IEC 61158 and IEC 61784. Among these protocols are protocols like PROFIBUS [4], Interbus [5], PROFINET IO [6], EtherCAT [7], and SERCOS [8]. An overview about these protocols can be found in [1].

To support real-time requirements, industrial Ethernet based systems use an isochronous communication bus with very low bus cycle jitter. Isochronous communication buses are typically applied when process control systems have with very high performance requirements to synchronize measurement, control, actuation and communication. Some of protocols stated above provide support for isochronous operation, such as PROFINET-IO. To enable synchronous operation of different nodes, even below 200 milliseconds, the Precision Time Protocol (IEEE 1588) [9] is used to synchronize the involved nodes.

The basic idea of many real time Ethernet variants is to have time periods that are reserved for real time traffic, see Figure 3. These periods are called also channels or phases.

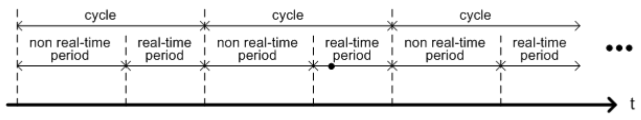


Figure 3. RT and Non-RT Periods of a Real-Time network Segment

Within these real-time periods, a strict scheduling is being applied. Non real-time capable devices should not interfere with these periods to avoid functionality loss through communication disturbances.

III. Real-Time Network Attachment Module

In the following, the problem statement and a solution is described targeting the connection on uncontrolled and potentially malicious non real-time capable devices to real-time networks. The reason for a device being malicious may be an intentional attack, but as well the fact that the device may be infected unintentionally by malware.

A. *Problem Statement*

The correct operation of a real-time network segment depends on the fact that all attached network devices conform to the real-time protocol. This is an additional requirement to the known network traffic filtering as performed by packet filters or application layer firewalls. It is not sufficient to limit the maximum data transfer rate, but to allow or discard data traffic depending on whether it is sent ensuring a period reserved for real-time data.

Different usage scenarios require that network devices shall be connected to a real time Ethernet segment that is not supporting the real-time enhancements. Examples comprise a service notebook connected temporarily during operation for maintenance or diagnosis. Further examples are provided through attachment of standard network equipment like a file server (NAS / network attached storage), video camera, a data recorder, etc., to be used within a real-time network segment. For the future, connecting new devices in a plug-and-play manner during the operation of a real-time automation system is envisaged. In particular when a device is connected to different networks over time or used as general PC, it is not possible to guarantee that the device has not been infected by malicious software.

In all these scenarios, it shall be ensured – in addition to ensuring that only allowed network traffic is occurring – that the real-time communication in the network segment is not affected. A real-time aware filtering of data communication will in practice be combined with well-known firewall-like filtering depending on the contents of data packets and on the connected device.

B. *Real-Time Aware Network Access Control*

This section first describes the new concept of real-time aware network access filtering. This targets different properties as there are the adherences to the real-time requirements, the detection of unauthorized components on the network segment as well as unexpected traffic induced by one of the network components.

To solve the described problem, a real-time network attachment module (RTAM) is proposed that prevents medium access to the real time network segment during time periods that are reserved for real-time traffic. While being a mechanism that could be used as such, preferably RTAM will realize filtering depending on contents of data packets and depending on an authentication of the connected device described in the following subsections.

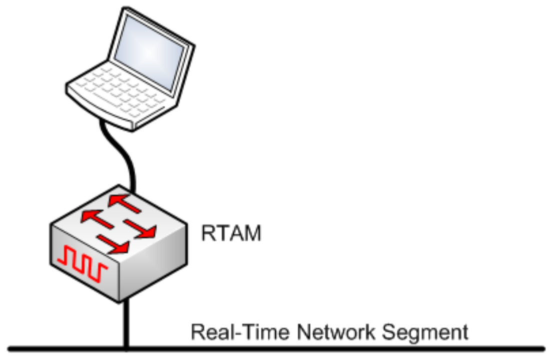


Figure 4. Real-Time Attachment Module

The real-time attachment module (RTAM) allows connecting an arbitrary network device with a real-time network; see Figure 4. It is ensured that the real-time traffic is not disturbed by the connected device. Towards the network device, a standard network interface is provided so that the network device does not require any adaptations or be aware of real time traffic.

There exist different realization options to prevent disturbance of real-time traffic:

- The simplest option it to interrupt physically the connectivity to the network segment during real-time periods. This has however the disadvantage that frames that happen to be sent during a real-time period are lost.
- A pseudo-carrier may be provided by RTAM towards the connected device during real time period independently of whether there is traffic on the real-time network segment or not. This prevents that the connected device transmits data during these time periods.
- A more complex option would be to buffering frames transmitted by the connected device and resend them during an allowed non-real-time period. So basically, a network switch is realized.

It depends on the target use case, which of the realization options is the most appropriate. While the first option is relatively simple, it has the problem of packet loss, which is addressed by the latter two options. The third option is the most flexible one, which basically works in a store and forward operation, ensuring that the real-time periods are obeyed and avoiding packet loss by buffering communication from the attached device to be sent during the non real-time periods. As typically the real-time network access will be combined with packet filtering that requires packet processing anyhow, the third option of buffering frames is the preferred option.

A preferred realization is a stateless or stateful packet filter that performs filtering decisions depending on both the packet contents the real-time period. The information whether a packet has been received during a real-time period or a non-real-time period is used as a filter criterion to allow or discard a packet. Furthermore, the actions to allow or discard a packet may be extended with a “delay” action so that traffic not permitted for a real-time period can either be discarded or delayed.

The configuration of filter rules can be organized in different ways:

- Configure separate filter rule sets for contention period and non-contention period.
- Single filter rule set with filter criteria “contention period” / “non-contention period” and actions extended from simple “allow/deny” to “delay-non-contention”. Allow traffic, but delay it to a non-contention period.

Configuring the filter rules appears to be simpler with a single filter rule set supporting additional filter criteria (real-time filter period) and actions.

In an extension, the connected device may be identified and authenticated, e.g., based on the IEEE 802.1x standard [10]. This is required in particular to support plug-and-play of network devices of different kind. After the device has been authenticated, a device-specific filtering policy is determined locally or using an authentication, authorization, and accounting (AAA) server [10]. While a real-time capable device may be granted direct, unlimited access to the real-time network segment, a generic device as an office notebook gets only limited access to non-real-time periods. Unknown or unauthenticated devices can be rejected, i.e., they do not get access to the field level network.

In a further extension, the RTAM module may provide access to the real-time network based on priorities of the non-real-time devises. The priority of a non-real-time device may be a matter of policy. As an example, non-real-time devices acting on the detection of security breaches may have a higher priority for changing the configuration of the real-time network to ensure reliable operation. Monitoring, e.g., queries from the non-real-time network (like MES systems) regarding process monitoring data recorded from the devices on the real-time network may get a lower priority.

C. *Real-Time Aware Intrusion Detection / Intrusion Prevention*

Intrusion prevention systems (IPS) or intrusion detection systems (IDS) monitor network traffic to detect anomalies [11]. RTAM functionality may be integrated into the IDS / IPS monitoring a real-time automation network.

The IDS/IPS analyzes communication on the network segment. In addition to the contents of data communication, it analyzes whether a data packet has been observed during a real-time period or a non-real-time period. A security event is detected when data traffic allowed only for a non-real-time periods is observed during a real-time period.

Since industrial automation networks are typically engineered, the communication behavior of the connected nodes under normal circumstances is known. This information can be used to support the analysis, which can either increase the efficiency or decrease the requirements on the hardware of an IDS/IPS system.

D. *Real-Time Aware Network Access Control*

Network Access Control (NAC) technology provides the functionality of verifying that a network accessing host complies with a dedicated policy [12]. Typical application examples are given through enterprise networks, were mobile clients like laptops, tablet PCs, or personal digital assistants, are frequently connected for local or remote access to the network.

Using NAC functionality it may be checked, if the client (device) has a certain operating system and what the patch status of this operating system is. Moreover, additional features like the availability of a virus scanner and the pattern status may be checked as well. If the client complies with the given policy, access to the network is granted. If not, the client may be put into a quarantine zone, in which he can be updated with the appropriate software packages. This can be depicted as multi-step approach consisting of:

- Detection of devices connecting to the network
- Identification and authentication of clients/devices
- Validation of device compliance to a given policy
- Authorization of compliant devices or
- Remediation of clients to comply to policy

The NAC functionality is standardized by the Trusted Network Connect Working Group (TNC WG) of the Trusted Computing Group (TCG) [11]. The IETF also addresses this functionality in the Network Endpoint Assessment (NEA) Working Group [13].

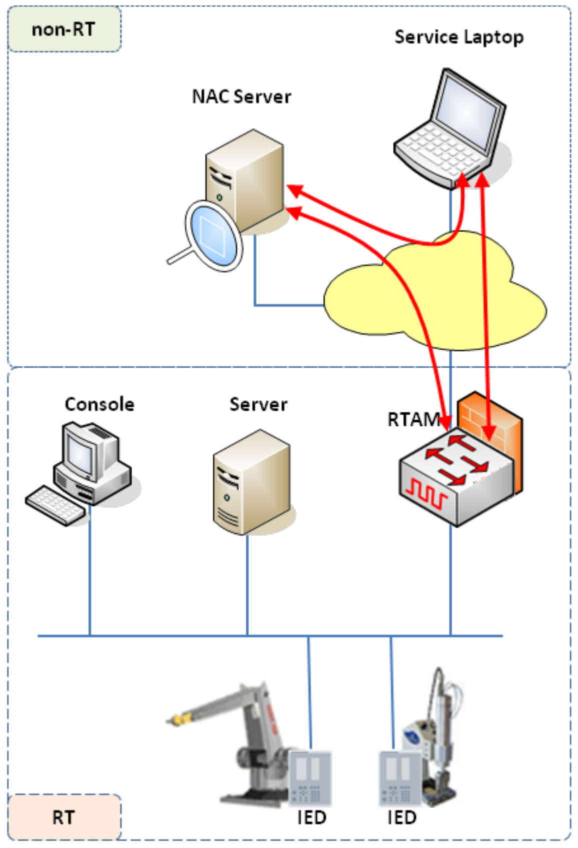


Figure 5. RTAM-NAC-FW Integration

While the network setup in industrial automation used to be static, future deployments, as currently discussed in ongoing funded projects, may be more dynamic. Providing the opportunity in industrial automation for quick replacements of IEDs, attachments of service laptops, remote administration and also agile production requires the compliance of the device compliance to a local policy to ensure reliability of the industrial automation. Moreover, this approach may also be used to realize a secure inventory management, which collects information about all nodes in a network, targeting the observation and maintenance of system integrity. Hence, NAC functionality is increasingly required in industrial automation, too.

RTAM may benefit from a NAC to ensure that only devices complying with a dedicated policy connect to the module. One realization option here is to combine the RTAM module with the NAC functionality of a switch as depicted in Figure 5. Before granting access to the industrial automation cell, the service laptop has to be checked regarding compliance to the operator’s policies. Once this process has finished, the RTAM gets the confirmation from the NAC server. Depending on the confirmation, the RTAM may provide the service laptop with an unrestricted or restricted access option to the real-time network. The restricted access variant combines the firewall approach additionally with RTAM to, e.g., check the commands intended for the real-time network before actually submitting them.

IV. RELATED WORK

Firewalls for filtering network traffic are a widely deployed security technology. While firewalls are available supporting real-time applications like VoIP (i.e., SIP and RTP), they do not support industrial real-time Ethernet layer 2 communication [14]. Also protection mechanisms against denial-of-service attacks are known that limit the allowed data rate, but these do not address specifics of real-time Ethernet as well [15].

Intrusion Detection and Prevention functionality is already known from and successfully used in telecommunication networks to detect and defeat unexpected traffic, e.g., through network externally launched denial of service attacks or network-internally deployed malware. An overview about IDS and IPS systems is given in [16].

They are typically not used in industrial automation networks. As stated in section III, the engineering of industrial automation networks can be considered in the definition of rules for IDS/IPS systems. This can even be automated if the engineering data is provided to these systems.

V. SUMMARY AND OUTLOOK

This paper described a new real-time security mechanism that filters network communication depending on whether the data affects a time period reserved for real-time communication. The mechanism allows connecting different kinds of network devices to a real-time control network segment securely by ensuring that the real-time communication within the network segment is not affected.

It goes far beyond well-known protections against denial of service attacks that do not respect the specific operation characteristics of real-time Ethernet networks, and filtering of data communication by a firewall depending on the contents of data packets.

This is a basic security feature enhancing known network security mechanisms to support various use cases in which network devices that cannot be trusted to respect real-time restrictions can anyhow be connected securely to a real-time network segment. Examples include temporarily connected monitoring, maintenance and service devices as well as dynamic plug-and-play of different automation devices.

REFERENCES

[1] M. Felser, "Real-time Ethernet – industry prospective," Proc. IEEE, vol. 93, no.6, June 2005, pp. 1118-1128, http://www.felser.ch/download/FE-TR-0507.pdf [retrieved: June 2012].

[2] T.S. Sidhu, M.G. Kanabar, and P. Palak, "Implementation issues with IEC 61850 based substation automation systems," Proc. Fifteenth National Power Systems Conference (NPSC), Dec. 2008, http://romvchvlcomm.pbworks.com/f/p274.pdf [retrieved: June 2012].

[3] IEEE, "IEEE standard for information technology-specific requirements - part 3: carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications," IEEE standard 802.3-2008, 2008, http://standards.ieee.org/about/get/802/802.3.html [retrieved: June 2012].

[4] Profibus Nutzerorganisation, "Profibus system description – technology and application," Nov. 2010, http://www.profibus.com/nc/downloads/downloads/profibus-technology-and-application-system-description/download/12821/ [retrieved: June 2012].

[5] Interbus Club, "Interbus basics," http://www.interbus.de/get.php?object=497 [retrieved: June 2012].

[6] Profibus Nutzerorganisation, "Profinet system description - technology and application,“ http://www.profibus.com/nc/downloads/downloads/profinet-technology-and-application-system-description/download/11864/ [retrieved: June 2012].

[7] EtherCAT Technology Group, "EtherCAT the Ethernet fieldbus", May 2012, http://www.ethercat.org/pdf/english/ETG_Brochure_EN.pdf [retrieved: June 2012].

[8] Sercos international, "Sercos the automation bus," April 2012, http://www.sercos.de/sites/default/files/sercos_en_april_2012_v1.pdf [retrieved: June 2012].

[9] IEEE, "IEEE Standard for a precision clock synchronization protocol for networked measurement and control systems," IEEE standard 1588-2008", 2008, http://standards.ieee.org/findstds/standard/1588-2008.html [retrieved: June 2012].

[10] IEEE, "Port based network access control", IEEE standard 802.1x-2004, Dec. 2004, http://standards.ieee.org/getieee802/download/802.1X-2004.pdf [retrieved: June 2012].

[11] A. N. Mahood, C. Leckie, J. Hu. Z. Tari, and M. Atiq, "Network traffic monitoring: application to SCADA security," Springer Handbook of Information and Communication Security, Springer, 2010, pp 383-405.

[12] TNC – Trusted Network Connect, http://www.trustedcomputinggroup.org/developers/trusted_network_connect/ [retrieved: June 2012].

[13] IETF NEA WG, http://datatracker.ietf.org/wg/nea/ [retrieved: June 2012].

[14] R. Falk and S. Fries: Voice Security: Sichere Sprachkommunikation in Unternehmen unter Benutzung aktueller Technologien, vde Verlag, 2008.

[15] John Ioannidis and Steven M. Bellovin, "Implementing pushback: router-based defense against DDoS attacks," Proc. Internet Society Symposium on Network and Distributed System Security, 2002.

[16] Karen Scarfone and Peter Mell, "Guide to intrusion detection and prevention systems," NIST SP 800-94, Feb. 2007, http://csrc.nist.gov/publications/nistpubs/800-94/SP800-94.pdf [retrieved: June 2012].

Improving Online Account Security: Implementing Policy and Process Changes

Pankaj Goyal
MicroMega Inc.
Denver, USA
e-mail: Pgoyal@micro-mega.biz

***Abstract*— User authentication, a difficult problem, suffers from various shortcomings with the prevalent use of passwords as an authentication method. Requirements for password memorability and usability make them easy to break. Password reuse across sites, including insecure sites, phishing and spoofing attacks, requires that financial institutions examine security by analyzing end-to-end processes and identities involved. This paper presents an approach for intrusion tolerance, and the necessitated changes to processes and policies.**

Keywords - online security; transaction security; identity security; security policy; financial systems; intrusion tolerance

I. Introduction

User authentication is one of the most difficult problems in Internet security [19]. Passwords are ubiquitous in their use for authenticating users and pose the biggest security challenge. Users frequently forget passwords, necessitating expensive customer support calls or automated backup authentication schemes often involving challenge questions, that are even weaker forms of authentication. Users also can have their passwords stolen through phishing [5, 11], social engineering, man-in-the-middle (MITM) [13], and key-logging attacks, or they may share [32]. Most of the issues related to passwords have been well documented and numerous efforts to alleviate some of these issues have been proposed and implemented; a complete survey of the literature is beyond the scope of this paper but some of the methods are surveyed in Section II. Passwords because of their usability, ease of implementation, etc., are unlikely to be dislodged as the user authentication method of choice for online services; Herley et al. [19] lists a number of barriers to move away from passwords with a major obstacle being the diversity of requirements.

People today, are subscribers of an increasing assortment of online services: news, social networking, shopping, financial and medical. Unfortunately, users have a difficulty in creating unique, easily memorizable, secure passwords for all the services that they use. Thus, users tend to use something from their background (family, friends, pets, history, likes, dislikes, etc.) for creating passwords and overwhelmingly tend to reuse passwords across services; it should be noted that the security mechanisms of these service sites is as varied as their nature and content. Users password entropy is likely to remain constant, while the number of services that a user subscribes to increases and as does the adversary's computational power) and access to users personal information.

Attempts to address some of the weaknesses in password security include multi-factor authentication to increase security of a password-based system. Multi-factor authentication involves the use of more than one mode in the authentication processes, for example, two-factor authentication can be implemented by coupling the use of passwords with (1) a security-challenge or (ii) a code sent over another communication channel (example, mobile phone) that the user is then required to enter. These schemes still suffer from vulnerabilities, in particular, the real-time man in the middle attack. Password management, including password retrieval and resets, are major issues and areas of vulnerability. The efficacy of security-challenge questions, however, in the era of easy availability of personal data is in serious doubt.

In this paper, we present an approach for intrusion tolerance of a security system for a financial institution (FI). As part of the analysis, many best practices, their advantages and risks were examined and attempts made to find robust solutions to address these risks. In addition, the design attempted to protect victims of phishing or other attacks where the victim's password has been compromised. The analysis led to definition of policies and processes to ensure *authorization* of all sensitive actions – actions that can negatively impact the legitimate user and/or FI. Authorization is implemented using only an out-of-band communication channel, so that it is not susceptible to a real-time man in the middle attack over the primary interaction channel.

Section II presents an overview of the current approaches and their limitations. Passwords (or their predecessor watch-words) and associated vulnerabilities have existed for millennia. It is a given that attacks on authentication schemes will continue and get more sophisticated. It is, thus, not enough to attempt to secure passwords, use processes to manage password change and use of a separate channel for on-time passwords (OTP) or authorization codes. If a password change requires authentication but, say, the email address in the profile does not, then an attacker on gaining entry would first alter the email address thereby by blinding the actual owner of all changes to its account. In the natural world, a financial transaction involve multi-channel interactions and authentications. Thus, what is required, in the online world, are methods and processes that minimize damage in case of password compromise. In Section III, this paper presents a systems approach to identify at risk items and activities from a compromise of authentication systems, and methods and processes for intrusion tolerance so as to mitigate financial losses. The proposed schemes use bi-directional multi-channel interactions, use methods and processes to authenticate profile changes.

II. BACKGROUND AND RELATED WORK

The increased use of computer systems is accompanied by an increased, and increasingly well-organized, attempts to breach their security. One of the most vulnerable elements of security is the *password*. This is also the most common method of user authentication. Once authenticates, the user is restricted to perform only authorized activities.

Authentication: is a means of verifying that a user is who they claim to be. In the process the user presents a *user-identifier*, representing the user's identity, and a set of *credentials* (e.g., password or certificates). The user is granted access to the service only on verification of the user-identifier and the credentials.

Authorization: determines whether the client is allowed to perform certain tasks or operations. Authorization enforces policies that controls access to activities, resources or services.

Identity management is concerned with users' credentials and their access to services. Identity management systems consist of an authentication part which is used to verify the correctness of an entity's claim to identity, and an access control part, authorization, which grants access to applications and resources. User-identifiers represent our digital identities, who we are, when engaging in online activities and transactions. Ideally, users should choose different user-identifiers (user-ids) for each service, however, this would require users to memorize user-id's in addition to passwords; this is not practical as analysis of passwords use has shown that users tend to reuse passwords [14]. Identity management is also a major issue of concern but is not the focus of this paper; for an introduction into issues related to usability and privacy, see [20].

A. Password Management

A big inconvenience in password based authentication is that ideally a user should memorize and use different passwords for different services and that these passwords should not be easily breakable. If a common password is used across all services then a service (or an attacker with knowledge of the common password) can impersonate a user to another service or a service can impersonate another service to the user. Password reuse across multiple services increases their vulnerability as compromising a single password allows an attacker access to multiple accounts. Gaw and Felton [14] conducted an experiment and found that password reuse rates increased over time because people subscribed to more services but did not create more passwords as reusing passwords made passwords easier to manage. Also sharing passwords among relatives and close friends has been found to be a sign of trust and intimacy [32].

Forcing people to choose and remember strong passwords - those that tend to be long character strings including both Roman letters and digits - is unworkable because such passwords are also unmemorable [12, 23]. Bellovin and Merritt [3, 4], Gong et al. [17], and Lomas et al. [25] have proposed authentication protocols that are resistant to password guessing attacks but with the added cost of additional messages; Gong [16] optimized the number of messages and cycles required.

Passwords are frequently deployed in an insecure manner. The large inconsistency of implementations between sites and the frequency of simple mistakes show an unanticipated level of insecurity and confusion [2]. While a site may not hold critical information and, thus, use simple security measures, a compromise of its security can allow the attacker to gain access to a compromised user's critical account on another site because of the reuse of user-id's and passwords. Given this threat of cross-domain password attacks, insecure sites collecting passwords have the potential to impose a costly externality on more careful sites.

Mobile phones and various other types of trusted mobile devices have been suggested as a means of achieving a two-factor authentication through devices that we routinely carry. Alternatives to passwords for authentication include Public-key infrastructure (PKI) but PKI security has been successfully compromised. Sotirov et al. [33] identified a vulnerability in the Internet Public Key Infrastructure (PKI) used to issue digital certificates for secure websites. They executed a practical attack scenario and successfully created a rogue Certification Authority (CA) certificate trusted by all common web browsers. This certificate would have allowed them to impersonate any website on the Internet, including banking and e-commerce sites secured using the HTTPS protocol; successfully used in a number of high profile cyber-attacks including compromise of Microsoft Update program (allowed 'Flame' malware to spread). It should be noted that the CA issuing authorities GlobalSign, Comode and DigiNotar have all been hacked. Biometrics suffer from deployment scaling, privacy and authentication from untrusted hardware.

Weir et al. [37] compared three different authentication processes, a 1-factor and two 2-factor methods. They found that convenience and personal ownership as some of the most important criterion in user preference of authentication methods and majority of the study participants perceived the 1-factor method as being the most secure and most convenient option. In their study experienced users gave higher usability scores to the 1-factor method they currently use.

Over the years, a variety of solutions have been proposed to the issue of password management. Bonneau and Preibusch [7] provide a broad overview of password history and management research. Anderson [2] provides a comprehensive overview of security engineering.

B. Password Management Systems (PMS)

Typical PMSs allow password resets online, including when a user runs out of the allowed number of retries, and some inform the user by email that a password change was made. In the era of email overload (and phishing attacks), by the time a user reacts to this informational message the damage might be done.

For example, certain sites that do not engage in any financial transactions may implement loose security policies and controls, such as, easy retrieval of passwords (clear text). But with the prevalence of passwords use across cross-sites this is a serious vulnerability. Some sites use security challenge questions to allow a user to reset their passwords; this is also allowed when a user runs out of password tries. These challenge questions, and their answers, are typically based on the user's background. With the increasing availability of online personal data, including through social networking sites of even such information as friends and pets, the efficacy of these security challenges is seriously in doubt. Secondly, they are vulnerable to real-time man in the middle attack.

FIs are starting to deploy dynamic challenge questions and two-factor authentication; the term FI includes online shopping, gaming, etc., as the issues faced by these online systems are similar. Challenge questions are commonly used to authenticate users who have lost their passwords. The PMSs that allow password resets on successful answer to a set of challenge questions suffer from the fact that increasing amounts of personal data is widely available online. Patterns have been found to exist in the security questions [21, 28]. An analysis of user-generated challenge questions found that 34% of user questions asked for a human name, 15% asked for a pet name and 20% asked for a place name [21]. Of the remainder, 22% asked for a user's favorite item amongst films, singers, car brands, etc., 5% asked for a time, date, or number, and the remainder were ambiguous. Thus, a few simple categories of proper names cover roughly 70% of real-world questions. While personal security questions may have had their use when there was scarce online or easy availability of personal information, however, with the ubiquitous availability of personal information online, the security provided by such questions is doubtful.

C. *Password Managers*

The objective of password managers is to be able to use complex, not easily breakable, unique passwords for user authentication by service providers while the user utilizes a simple easily memorizable password for password manager access.

One approach is to create centralized, trusted authentication services, such as Microsoft's Passport initiative [26] and its security has been analyzed [24]. Such centralized services require both users and service providers to place their trust onto a centralized service ("Big Brother") and every service that would use the centralized system for authentication would need to make changes to their systems.

Another proposal lets users choose their own passwords and then store them in a safe, for example, Password Safe application [31] stores the data in an encrypted database on the user's machine, secured with a user-chosen master password.

Another method assigns fixed passwords for each site or service that can be computed whenever they are needed. For example, the Lucent Personal Web Assistant [15] operates as an HTTP proxy server that users access with a master username and password. They can then tag web site password fields to be automatically filled in with values derived from a hash-based function of the user's master password and the domain name of the web site.

PwdHash [29], a user web browser plug-in, applies a similar hash-based technique. Password Multiplier [18], is also a user web browser plug-in, extend the approach of Kelsey et al. [22]. In Kelsey et al's method [22], the password is derived by repeatedly iterating a hash function on the original master password. Halderman et al. [18] compute the site-password in two steps. In the first step, an interim value is derived by applying the secure hash function is iterated k_1 times on the concatenation of the username and master password. In the second step, the site password is derived by applying the secure hash function is iterated k_2 times on the concatenation of the site name, master password and the interim value from step 1. Chiasson et al. [9] evaluate PwdHash and Password Multiplier suffer from major usability problems that cause security exposures – exposures that the users are unaware of. Also, password managers cannot prevent MITM attacks.

D. *Single Sign On (SSO)*

Fundamentally, Web single sign-on (SSO) systems shift the functions of identity collection and authentication from the content servive provider (CSP) to Identity Providers (IdP); in the process the CSP becomes a relying party (RP) of the IdP. An IdP issues identities or credentials to users, while an RP depends on the IdP(s) to assert the user credentials before allowing access to its services. This enables users to leverage one identity across multiple RPs.

An inherent risk of usingWeb SSO is that one compromised account on an IdP can result in breaches on all services that use this compromised identity for authentication. The SSO introduces a single point of failure or compromise -- the IdP systems and infrastructure [27]. An inherent characteristic of web applications is that some of the internal information flows are inevitably exposed on the network and encryption is insufficient to safeguard against information leaks [8]. Users may become accustomed to being redirected to identity provider web-sites for authentication. To prevent phishing attacks, users must verify the authenticity of an identity provider before entering their credentials.

Users and the RP have to trust the integrity of the IdP. In web SSO ecosystems, the issue of liability becomes highly complex as the integrity of the ecosystem depends upon the quality of the implementation of the IdP and all RP clients of the IdP; Wang et al [36] list the issue of implementation complexity as a major issue for SSO's. A number of researchers have investigated flaws in the protocols and implementations of such systems and surveyed [8, 27, 34, 36].

E. *Man-In-the-Middle Attack (MITM)*

It is possible for an attacker to intercept the communication between the customer and the FI server and impersonate them both [13]. The attacker tricks the customer into logging into the attacker's website, and masquerade as the real FI. This can for example happen through the attacks commonly known as *web spoofing* and *pharming* [5, 13].

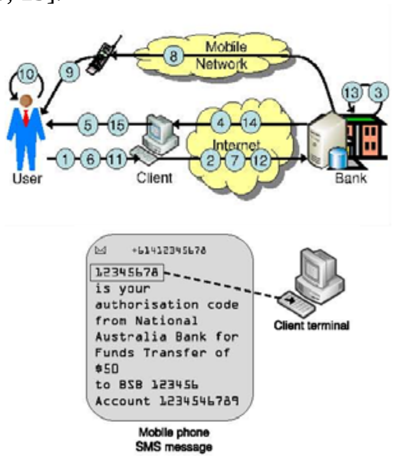

+61412345678
12345678
is your
authorisation code
from National
Australia Bank for
Funds Transfer of
$50
to BSB 123456
Account 1234546789

Figure 1. Authorizing bank transactions via SMS (from AlZomai [1])

To counter MITM, some FIs use an OTP (One-Time-Password) to authorize each transaction (Fig. 1); also, referred to as the authorization code. It is common to send the authorization code through an out-of-band channel, for example, email or a mobile SMS; out-of-band channels are also referred to as authorization channels while the primary interaction channel is known as the in-band channel. The transaction authorization method based on SMS messages was introduced by FIs in response to phishing attacks. However, an investigation of the security and usability of SMS-based transaction authorization method and found that it is vulnerable to stealthy attacks, such as minor changes to account numbers [1]. The scheme also suffers from the fact that the user has to enter the authorization code using the in-band channel – potential for mistakes and still subject to a sophisticated MITM attack.

F. Combining Text and Graphics

In an attempt to counter password theft through phishing attacks and to differentiate real sites from spoofed sites, mix of text and graphic passwords have been proposed [10, 35]. Some online FIs employ site verification schemes; for example, SiteKey [6] displays a user selected image back to them at login. But, an empirical study [30] found that users will still enter their passwords when presented with fraudulent messages claiming that the image authentication server is down.

III. Financial Institution (FI) Situation and Analysis

The FI allows users to perform financial transactions online, including deposits, bill and recurring payments, and also sells certain financial and FI-related items (coin collections, piggy-banks, etc.) that are then shipped to the user specified address.

A. Problem

Based on analysis of available literature and experimental results (some of these are referenced in Section II), it was clear that (a) all identity credentialing and authentication schemes are vulnerable, and that these credentials can become known to attackers; and (b) passwords have to be used for user authentication. It was also clear that in the current environment challenge questions are ineffective and, while they would be implemented for psychological comfort, will not play any role in user authentication. The FI was interested in intrusion tolerance mechanisms that could minimize impact of, say, MITM attacks, and in using transaction authorizations.

B. Challenges

Let us consider the situation where the attacker has gained access to the users' identifier and password; the attacker takes control of the account and make changes to, say, the email address and mobile phone number, or add payees, etc.

It is not typical of online sites to inform users if, say, their email address or phone number have been changed. The email address change prevents the victim from receiving notices of password change, including links that allow password changes to be made; instead the attacker will receive them. The mobile phone number change allows the attacker to receive the transaction authorization SMSs.

TABLE I. Terms

(…)	tuple of items
{…}	set of items
A_i	Attribute name
a_i	value for attribute A_i
Account	(Account-Type, Account-Identifier, {(Account-Actions, Limit), …})
Address	mailing, shipping, billing, primary/secondary email, SMS
Authorization Action Channels (AAC)	are associated with a ROC and user action
Channels	postal mail, email, Fax, Voice call, browser, smart phone app
Identity	{ (Identity-Attribute, value), ….. }
Identity-Attribute	Name, Address, Phone, Account, Identity-Category
Identity-Category	Self, Payee, Beneficiary
Name	User/Account Holder, Payee, Beneficiary [e.g., Family-Friends]
Phone	Home, Work, Mobile, Other, Fax
Request Originating Channel (ROC)	channel through which any user action is requested (for example, logging in)
User Actions	login, delete/add/change any profile information

A number of other challenges were analyzed. During the analysis a number of terms were identified (Table 1; not all terms identified) and it became clear that users needed to be informed of any identity changes. A more stringent and secure criterion is that any identity change needed to be authorized through an authorization channel distinct from the request originating channel; this is different from the currently used methods where the authorization is performed on the request originating channel. The primary purpose of the analysis and the formalization of terms was to identify at risk items – items that can contribute to circumvent security.

C. Interaction Channels

At the FI, the following channels can be used to originate requests, including changes to user profiles and accounts:

- Paper, typically signed forms and letters
- Telephone (only from registered phones)
- Web site
- *In person*

For each of these requesting originating channels, alternate authorization channels are specified (Table 2); it is possible for users to choose a preferred authorization channel. Emails are used as a notification method and contain instructions on how to perform the requested authorization. Originating requests in person allows the FI personnel to physically authenticate the user and the user authorizations the actions by signing documents (may be online).

TABLE II. AUTHORIZATION CHANNELS FOR REQUEST ORIGINATING CHANNELS

Request Originating Channel	**Authorization Channels**
Paper	Email, Paper (signature), SMS, Telephone
Telephone - Landline	Email, Paper (signature), SMS
Telephone - Mobile	Email, Paper (signature)
Web site	Email, Telephone, SMS
In Person (face to face)	At service/interaction point; receipts may be transmitted by Email

D. Policy Changes

- All changes, including add/delete, to identity require explicit user authorization
- New financial transactions require explicit user authorization
- On users' initiation of a change to identity or a new financial activity, the change/activity will be put in pending status and, thus, are effectively inactive.
- All authorizations must be made through a channel distinct from the request originating channel.
- Both authorization requests and user responses to these requests must be made on the authorization channel.
- A change in pending status is made active or inactive based on the user authorization response.
- Challenge questions shall play no role in making decisions about a users' identity
- In person (physical presence) on the presentation of valid photo-id shall allow the user to request immediate changes with authorization received in the form of users' signature.

E. Process Changes

Every process that deals with users (interactions), user profiles, and user accounts is affected. Some of the major process changes were to call center processes as the policy changes impacted a number of processes. For example, customer service representatives could no longer reset passwords on the phone, or make changes to emails – any change to identity; they should mark the requested changes to be pending status and await authorization through an alternate channel.

F. Impact of these Changes

1) Password compromise

The affect of these changes is that even with the attacker logged into the victims' account, the attacker is prevented from making changes to the victims' identity. Thus, for example, the attacker cannot make a change to the victims' registered email address or mobile number; any changes will need to be authorized by the victim before they can take effect and such authorization will occur through a different channel. Similarly, the attacker cannot add a new beneficiary or payee or make changes to their addresses or account details. Thus, the attacker is effectively prevented from causing any harm to the victim.

2) Man-in-the-Middle (MITM) Attack

Similar to the password compromise scenario (above), the attacker is unable to authorize its illegal gain activities, as all authorization will occur on a channel different from the channel that the attacker has spoofed.

It should be noted that there is no defense against an attacker who has total control over the victim, viz., the attacker has control over all possible communication channels.

G. Other Design Options

It is possible to construct channel use charts where for each request originating channel a set of approved authorization action channels are defined. During the design phase it was decided against including this extra level of complexity as the potential benefits were not clear.

There was considerable debate on whether authorization should be required for any transactions above a certain minimum amount. Since, any beneficiary or payee identity would be created only on explicit authorization any benefits of a transaction would accrue to members of the account holder's circle and that any benefit would be subject to the upper limit for that identity. The problem with creating exceptions is that there is a cost of implementation and ongoing costs as changes to exceptions are made. Secondly, any activity not initiated by the account holder is still a major inconvenience both to the account holder and the FI.

H. Usability

Users have the option to opt out of out-of-band authorization for certain profile changes and low-value financial transactions; users can not opt-out of financial transactions above a FI defined threshold, nor can they opt out of authorizing changes to their passwords, email address and authorization channel information. Given that an attacker would be unable to gain financially it is expected that attackers would concentrate their attention elsewhere.

Authorizations utilize automated systems, similar to those utilized to automatically activate credit and debit cards. Thus, there is no extra burden on call center representatives.

IV. Conclusions

Passwords are likely to retain their predominant position in user authentication. With the increase in the sophistication of phishing and man-in-the-middle attacks, cross site use of passwords by users, sharing of passwords and the wide availability of personal information, requires that FIs assume password compromise, including knowledge, and devise intrusion tolerant processes and policies to minimize adverse effect on their customers and themselves. Most of the methods proposed by researchers address one or two shortcomings of passwords, for example, password breakability. Security implementations at online sites do not consider end-to-end impacts of password compromise. This paper has presented some of the outcomes of a comprehensive analysis and methods incorporated in the design of an FIs security system tolerant to malicious intrusions. One of the characteristics of the design requires that any change to an identity/profile requires user authorization, say, using mobile networks as an authorization channel. With near ubiquitous use of mobile devices these methods are not onerous.

Identity credentialing and authentication methods, other than passwords, are also not immune from compromise. The complexity of some implementations [36] may introduce vulnerabilities that require new methods and tools to detect. In the absence of faultless systems and processes, it is incumbent on organizations to analyze intrusion impacts, and introduce intrusion tolerant systems, processes and policies. These intrusion tolerant schemes have to be designed by each organization based on their risk exposure.

References

[1] M. AlZomai, B. AlFayyadh, A. Jøsang, and A. McCullagh, “An Experimental Investigation of the Usability of Transaction Authorization in Online Bank Security Systems,” Sixth Australasian Conference on Information Security, pp. 65-73, 2008.

[2] R. Anderson, Security Engineering: A Guide to Building Dependable Distributed Systems, Second Edition, Wiley Publishing, 2008.

[3] S.M. Bellovin and M. Merritt, “Augmented Encrypted Key Exchange: A Password-Based Protocol Secure Against Dictionary Attacks and Password File Compromise,” 1st ACM Conference on Computer and Communications Security, pp. 244-250, 1993.

[4] S.M. Bellovin and M. Merritt, “Encrypted Key Exchange: Password-Based Protocols Secure Against Dictionary Attacks,” IEEE Symposium on Security and Privacy, pp. 72-84, 1992.

[5] H. Berghel, ‘Phishing mongers and posers’, Communications of the ACM, 49(4), pp. 21–25, 2006.

[6] Bank of America SiteKey. http://www.bankofamerica.com/privacy/sitekey/ [retrieved: March 2012].

[7] J. Bonneau and S. Preibusch, “The password thicket: technical and market failures in Human authentication on the web,” The Ninth Workshop on the Economics of Information Security (WEIS 2010), 2010.

[8] S. Chen, R. Wang, X. F. Wang, and K. Zhang "Side-Channel Leaks in Web Applications: a Reality Today, a Challenge Tomorrow," IEEE Symposium on Security and Privacy, pp. 191-206, 2010.

[9] S. Chiasson, P.C. van Oorschot, and R, Biddle, “A Usability Study and Critique of Two Password Managers.” 15th USENIX Security Symposium, 2006.

[10] S. Chiasson, Usable Authentication and Click-Based Graphical Passwords. PhD thesis, Carleton University, Ottawa, Canada, January 2009.

[11] R. Dhamija, J. Tygar, and M. Hearst, “Why Phishing Works,” SIGCHI conference on Human Factors in Computing Systems, pp. 581-590, 2006

[12] D.C. Feldmeier and P.R. Karn. UNIX Password Security - Ten Years Later. Crypt0 ‘89, volume 435, Lecture Notes in Computer Science, Springer-Verlag, pp. 44-63. 1989.

[13] E. Felton, D. Balfanz, D. Dean, and D. Wallach. Web Spoofing: An Internet Con Game. 20th National Information Systems Security Conference, also Technical report 540-96, Princeton University, 1997, http://www.princeton.edu/sip/pub/**spoofing**.html [retrieved: February 2012]

[14] S. Gaw and E.W. Felton, “Password management strategies for online accounts,” SOUPS '06 Proceedings of the second symposium on Usable privacy and security, pp. 44-55, 2006.

[15] E. Gabber, P.B. Gibbons, Y. Matias, and A.J. Mayer, “How to make personalized web browsing simple, secure, and anonymous” Financial Cryptography, pp. 17–32, 1997.

[16] L. Gong, “Optimal Authentication Protocols Resistant to Password Guessing Attacks,” Eighth IEEE Computer Security Foundations Workshop (CSFW '95), pp. 24-29, 1995.

[17] L. Gong, T.M.A. Lomas, R.M. Needham, and J.H. Saltzer, “Protecting Poorly Chosen Secrets from Guessing Attacks,” IEEE Journal on Selected Areas in Communications, 11(5).pp. 648-656, 1993.

[18] J.A. Halderman, B. Waters, and E.W. Felten, “A Convenient Method for Securely Managing Passwords,” 14th Intl Conf on World Wide Web (WWW’05), pp. 471-479, 2005.

[19] C. Herley, P.C. van Oorschot, and A.S. Patrick “Passwords: If We're So Smart, Why Are We Still Using Them?” Financial Cryptography and Security, pp. 230-237, 2009.

[20] M. Josang, S. AlZomai, and S. Suriadi, “Usability and Privacy in Identity Management Architectures,” Fifth Australasian Information Security Workshop: Privacy Enhancing Technologies (AISW 2007), pp. 143-152, 2007.

[21] M. Just and D. Aspinall, “Personal Choice and Challenge Questions: A Security and Usability Assessment,” 5th Symposium on Usable Privacy and Security (SOUPS’09), 2009.

[22] J. Kelsey, B. Schneier, C. Hall, and D. Wagner, “Secure applications of low-entropy keys” *Lecture Notes in Computer Science*, 1396, pp. 121–134, 1998.

[23] D.V. Klein, “Foiling the Cracker: A Survey of, and Improvements to Password Security,” 2nd USENIX Unix Security Workshop, pp. 5-14, 1990.

[24] D.P. Kormann and A.D. Rubin, “Risks of the Passport single signon protocol” 9th International World Wide Web Conference on Computer Networks, North-Holland Publishing Co., pp. 51–58. 2000.

[25] T.M.A. Lomas, L. Gong, J.H. Saltzer, and R.M. Needham, “Reducing Risks from Poorly Chosen Keys” 12th ACM Symposium on Operating System Principles, ACM Operating Systems Review, 23(5), pp. 14-18, 1989.

[26] Microsoft Passport service. http://www.passport.net [retrieved: July 2011].

[27] D. Pham and A. K. Sood, "An Intrusion Tolerance Approach to Enhance Single Sign on Server Protection," 2010 Third International Conference on Dependability, pp. 98-103, 2010.

[28] A. Rabkin, “Personal Knowledge Questions for Fallback Authentication: security questions in the era of Facebook,” 4th symposium on Usable privacy and security (SOUPS 2008), pp. 13-23, 2008.

[29] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J.C. Mitchell, “A browser plug-in solution to the unique password problem,” 2005. Technical report, Stanford-SecLab-TR-2005-1.

[30] S.E. Schechter, R. Dhamija, A. Ozment, and I. Fischer, “The Emperor's New Security Indicators,” IEEE Symposium on Security and Privacy, pp. 51-65, 2007.

[31] B. Schneier et al. Password Safe application. http://www.schneier.com/passsafe.html [retrieved: July 2011].

[32] S. Singh and A. Cabraal, C. Demosthenous, G. Astbrink, M. Furlong, “Password Sharing: Implications for Security Design Based on Social Practice,” SIGCHI conference on Human factors in computing systems (CHI ’07), pp. 895–904, 2007.

[33] A. Sotirov, M. Stevens, J. Applebaum, A. Lenstra, D. Molnar, D.A, Osvik, and B. de Weger, “Creating a Rogue CA certificate," 25th Chaos Communication Congress in Berlin 2008. Also: http://www.phreedom.org/research/rogue-ca/ [retrieved: July 2011].

[34] S. T. Sun, Y. Boshmaf, K. Hawkey, and K. Beznosov, "A billion keys, but few locks: the crisis of web single sign-on," 2010 Workshop on New Security Paradigms (NSPW'10), pp. 61-72, 2010.

[35] P.C. van Oorschot and T. Wan, “TwoStep: An Authentication Method Combining Text and Graphical Passwords,” 4th MCETECH Conference on eTechnologies, Lecture Notes in Business Information Processing, E-Technologies: Innovation in an Open World, pp. 233-239, 2009.

[36] R. Wang, S. Chen, and X. F. Wang, "Signing Me onto Your Accounts through Facebook and Google: a Traffic-Guided Security Study of Commercially Deployed Single-Sign-On Web Services," IEEE Symposium on Security and Privacy, pp. 365-379, 2012.

[37] C.S. Weir, G. Douglas, T. Richardson, and M. Jack, “Usable security: User preferences for authentication methods in eBanking and the effects of experience,” Interacting with Computers, 22 (3), pp. 153-164, 2010.

FIELDS: Flow Intrusion Extrusion Largescale Detection System

Nicolas Grenèche
Université Paris 13 — PRES Sorbonne Paris Cité
LIPN UMR CNRS 7030 / DSI
99 Avenue Jean-Baptiste Clément
93430 Villetaneuse (France)
Email: nicolas.greneche@univ-paris13.fr

Quentin Narvor, Jérémy Briffaut, Christian Toinard
ENSI de Bourges, LIFO — EA 4022
88 Bld Lahitolle
F-18020 Bourges Cedex, France
Email: {quentin.narvor,jeremy.briffaut, christian.toinard}@ensi-bourges.fr

***Abstract*—This paper presents an advanced pre-processing, called FIELDS, for analyzing the network traffic based on flow assessments. FIELDS is an extensible Network Security Monitoring that supports 1) advanced traffic pre-processing, 2) forensics and 3) existing Network Extrusion/Intrusion Detection Systems. FIELDS has been experimented during two months using a large real network thanks to its non intrusive nature. The results show the efficiency of different heuristics for pre-processing the traffic relevant of an intrusion. FIELDS provides an unified and efficient tool for pre-processing the network traffic and detecting/controlling the potential internal/external intruders. FIELDS solves the problem of scalability for the monitoring of the security of large networks. It can be easily extended to integrate other heuristics and correlate the different analysis.**

***Keywords*-Network flows; Intrusion Detection; Network Forensic; Network Capture; Attack Prevention**

I. INTRODUCTION

Network Security Monitoring is associated with different methods such as capturing the network traffic, analyzing the traffic, logging the traffic, forensics and network intrusion detection (NIDS). Classical NIDS has been studied since a long time [1]. They generally compute the ingoing and outgoing network traffic for detecting the intrusions. Studies mainly address how to learn [2] or modelize [3] a network threat. The major drawback is how to analyze the whole traffic in a scalable manner. FIELDS proposes a generic method for solving the problem of scalability. It provides a pre-processing of the outgoing traffic (pre-extrusion) as a mean to suspect the compromising of the local hosts (intrusion). Using the suspected list of hosts, FIELDS enables a safe Network Extrusion Detection System (NEDS) by reusing existing security tools (e.g. NIDS, anti-virus, anti-malware, HIDS, HIPS, ...) to measure the compromising. FIELDS provides an extensible and non-intrusive approach for 1) detecting potential intruders/victims based on efficient heuristics, 2) collecting the traffic of those eventually malicious hosts and 3) analyzing the corresponding flows and 3) reusing an existing packet filter while guaranteeing a low overhead. Despite FIELDS extrudes the outgoing traffic from the internal malicious hosts, it can provide the external malicious hosts. Thus, FIELDS can also improve the detection/prevention of intrusions coming from external hosts (pre-intrusion). Globally, FIELDS reduces the overhead of a NIDS/NEDS approach by permitting a consistent analysis only for a subset of the whole traffic associated with the internal/external suspected hosts. The second section describes the state of art related to heuristics for pre-processing the outgoing network traffic and to the network capture. A third section describes our major objectives for pre-processing the outgoing traffic, analyzing the suspected hosts and controlling/measuring the suspected hosts. The fourth section describes our solution. A fifth section describes our experimentation. Finally, the paper describes the perspectives and summarizes the FIELDS approach.

II. STATE OF THE ART

Classical NIDS approaches such as [2], [3] need pre-processing in order to limit the overhead and reduce the number of false positive alerts. Thus, our state of art considers only the pre-processing problem in order to have 1) an updated list of the suspected hosts and 2) an exhaustive capture of the abnormal traffic coming from those hosts. This section shows which heuristics enable to compute a list of suspects based on extrusion, i.e. abnormal outgoing traffic from internal hosts that is a consequence of a successful intrusion. Then, the section goes on with the methods available for capturing the traffic of the suspected hosts. Finally, it describes recent works related to extrusion.

IP blacklist: This is the most simple heuristic. Despite it's simplicity, this is currently massively used to detect malwares that contain hardcoded addresses or names. For malwares associated with a more sophisticated infrastructure containing dynamic hosts e.g. associated with fast flux based methods [4], the IP blacklist approach becomes inefficient. IP Blacklist can be used for both intrusion and extrusion monitoring i.e. external and internal malicious hosts.

DNS domain blacklist: This blacklist maintains a list of malicious DNS domain names. This technique is extrusion oriented because only the DNS requests from internal hosts are audited. The DNS resolver of an infected host can also responds with an unused IP address belonging to a sinkhole.

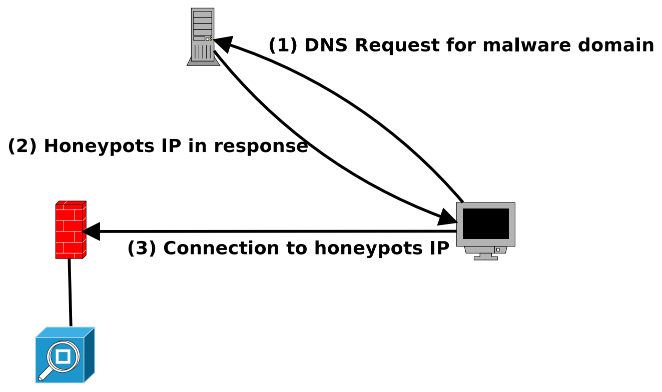

(1) DNS Request for malware domain
(2) Honeypots IP in response
(3) Connection to honeypots IP

Figure 1. DNS name blacklist

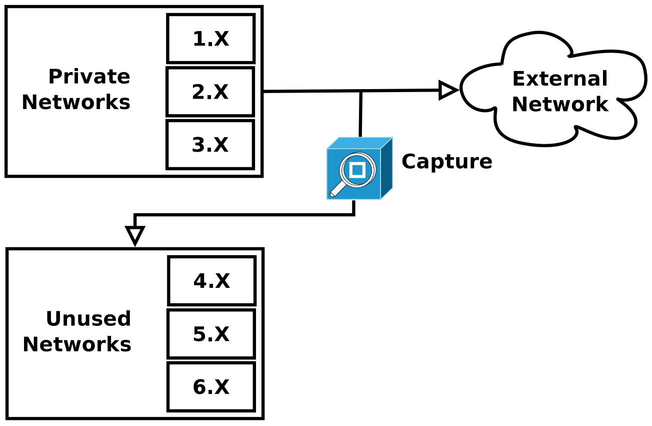


Figure 2. IP Sinkhole for 192.168.X.X/24

In this case, as shown in Figure 1, a honeypot can handle the connections of the malwares [5] to this address.

IP Sinkhole: ISP sinkholes are used at BGP level by Internet providers to redirect and log attacks against customers [6]. IP sinkholes are used by network administrators to monitor a traffic that tries to reach unused private networks [7], [8]. Private range of addresses are defined by the RFC1918. Addresses of the RFC1918, that are not used by the organization, are in the sinkhole. An implementation using a router is given in Figure 2. There are three real networks (192.168.1.0/24, 192.168.2.0/24 and 192.168.3.0/24). Routing tables are set up to route real private networks first. Then, RFC1918 networks are routed. Finally, a last route authorizes the Internet access. As a consequence, if a packet tries to reach a non existent (but belonging to RFC1918) network, it is captured by routing equipments. However, this solution is intrusive because it needs to modify the routing tables.

Network capture: As stated in [7], [8], [9] four different kinds of information can be stored for Network Security Monitoring (NSM): global statistics (how many data sent/received to/from a given network), flows (who talked to who and how), IDS alerts and comprehensive captures. For large organization, a comprehensive capture of each network flow is impossible but such level of detail is interesting in terms of network forensics. Common network sensors can be configured to perform a comprehensive capture (tcpdump) or only a capture of the flows (netflow, sflow or argus collector). Currently, there is no hybrid collector available in the literature. An hybrid collector would be able to switch from one capture mode to another one to zoom on supposed malicious flows.

Recent extrusion works: Solutions such as [10] proposes to install viruses and malwares for analyzing their behaviour. Others [11] reuses snort for analyzing outgoing traffic. However, those approaches consider analysis of viruses and do not solve the problem of scalability for analyzing the whole outgoing traffic.

III. MOTIVATIONS

The first objective is to suspect a host using a network extrusion. It is interesting since extrusion does not require to analyze the ingoing traffic reducing thus the overhead of the analysis. Moreover, it allows to detect threats coming also from usb or other removable resources and the hosts do not have to be monitored. The second objective consists in reusing simple tools efficient for capturing the traffic. It is a low cost approach since tools such as BSD Packet Filter are free and strongly supported. Third, our approach must support different types of heuristic for computing the list of the suspected hosts. Indeed, the IP/DNS blacklist and sinkhole can miss some infected hosts. Thus, the list of the malicious hosts can be computed using the the IP/DNS blacklist and sinkhole approaches and improved with other methods such as CERT notifications, OS fingerprinting or IDS alerts. Fourth, the approach must reuse existing security tools (e.g. anti-virus, anti- malware, HIDS, HIPS, NIDS, ...) to measure or recover the infected nodes. Fifth, the solution must solve the scalability problem by analyzing part of a network traffic that is limited to to the suspected nodes. Sixth, the network analysis must be non-intrusive and require only a limited modification of the capturing tool. Seventh, the monitoring solution must support not only internal hosts but also external hosts. The advantage is a common and extensible approach to manage and reduce the overhead of NEDS/NIDS tools.

IV. THE FIELDS APPROACH

FIELDS is a software that uses the BSD Packet Filter (PF) [12] tables and PCAP logging capabilities. It requires only a very limited modification of PF while supporting a large range of heuristics. In PF, a table provides a list of IP addresses matching a destination or a source. A table referred to as `<suspicious>` can be declared to store potentially infected hosts. The lookup operation on the table is very efficient. The logging infrastructure of PF enables to capture network packets that match a given rule in the PCAP format. For example, the following rule enables to capture all the traffic originating from the IP addresses available in the <suspicious> list:

```
pass in log on em0 from <suspicious> to any
```

The idea of FIELDS is to express malicious flows through a dedicated policy formalizing different heuristics. This way, FIELDS highlights malicious flows. When such a flow is detected, FIELDS switches into a comprehensive capture of the traffic. FIELDS can be considered as a kind of NEDS/NIDS pre-processor reducing the data size that has to be analyzed by existing NEDS/NIDS. In that sense, it is not concurrent of existing approaches but eases their management in a production environment in order to solve the scalability problem while reusing off-the-shelf NEDS/NIDS tools.

V. Packet Filter patch

FIELDS uses a modified Packet Filter. The proposed patch extends the capabilities of the PF rules. When a packet matches a rule, PF performs some actions: `pass`, `block` or `log`. Our patch adds a fourth action `add` which performs the addition of a source and/or destination address in a table. The Figure 3 describes our patch integration into PF. When a packet reaches the PF firewall, a lookup in the tables is preformed (step 1). A dedicated table can be declared for each malicious flows heuristic: a table `<blacklist>` to store the IP addresses that try to reach the hosts of the blacklist, a table `<sinkhole>` to store the addresses talking to the sinkhole addresses, etc. If the destination or source (depending of the meaning of the matching rule) is present in a table, the packet is logged (step 2a). If the corresponding address is not in a table, the packet is submitted to a set of filtering rules modelizing a set of malicious flows heuristics (step 2b). If the packet matches with a rule, its source and / or destination IP is added to the corresponding table (step 3a). For example, if a packet matches a rule describing a sinkhole heuristic, its source IP is added to the `<sinkhole>` table. Thus, any packet involving this IP address will be logged (step 2a). These logs are analyzed afterwards by different IDS tools (such as snort) in order to identify malwares. FIELDS extends the grammar of the PF rules. Our patch is about 300 lines of C, covering kernel code (100 lines) and userland `pfctl` command (200 lines). This patch is not intrusive since it does not modify the sensible parts of the kernel.

A. Extension of the PF grammar

A new option has been added in the rules syntax: the option `add`. This option must be used after all the usual options of pf. `add` have to be followed by at least one of these 2 options:

- `ipsrc <src_tblname>`: add source IP address of the matched packet to the table "`src_tblname`" ;

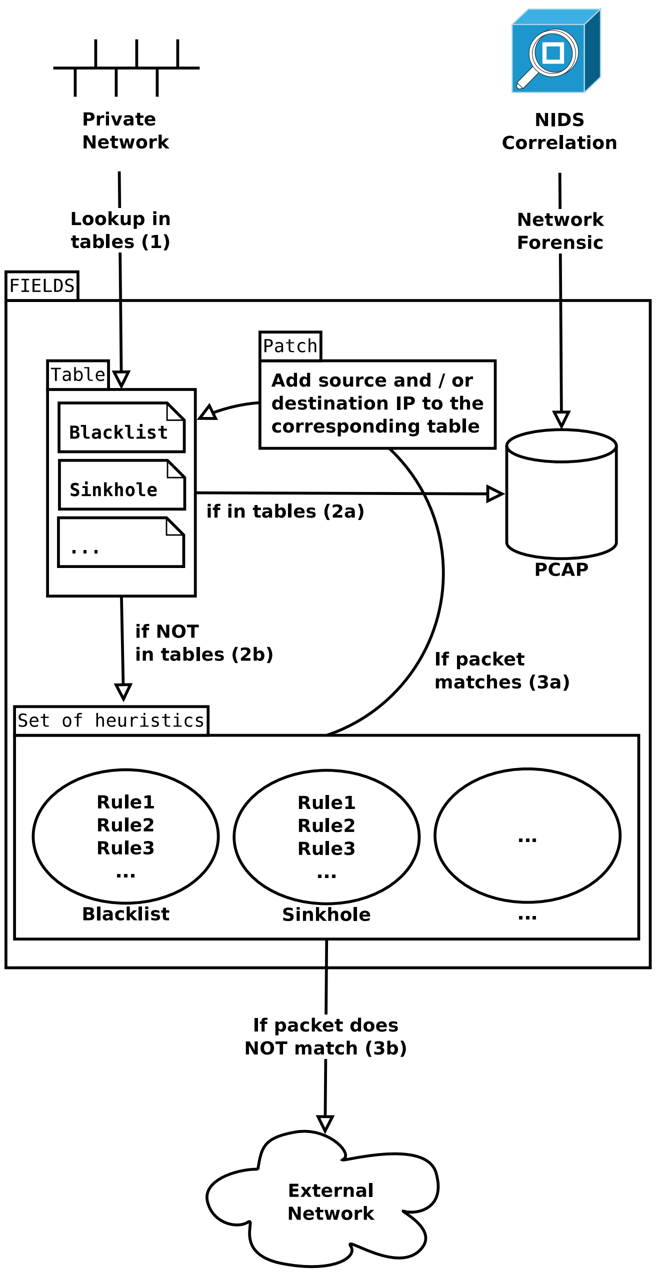


Figure 3. FIELDS architecture

- `ipdst <dst_tblname>`: add destination IP adress of the matched packet to the table "`dst_tblname`".

Let's give examples of such a rule:

```
pass in on em0 from 192.168.0.0/16 to any add ipdst <
    tableA>
```

In this rule, the destination addresses of all the packets coming on the network interface em0 from the subnet 192.168.0.0/16 will be added to the table named `tableA`.

```
pass in on em0 from 192.168.1.0/24 to 192.168.2.0/24 add
     ipsrc <tableA> ipdst <tableB>
```

In this rule, the source addresses of all the packets coming on `em0` from the subnet 192.168.1.0/24 to the subnet 192.168.2.0/24 will be added to the table named `tableA`. In the same way, destination addresses will be added to the second table named `tableB`.

B. Modifications on PF

A new structure named `add` has been added in the file `pfvar.h`. This structure stores data required by our new options:

```
struct add {
   u_int8_t check_add;
      struct {
      u_int8_t check_src;
      char src_tblname[PF_TABLE_NAME_SIZE];
      struct pfr_ktable *src\_tbl;
   } src;
   struct {
      u_int8_t check_dst;
      char dst_tblname[PF_TABLE_NAME_SIZE];
      struct pfr_ktable *dst_tbl;
   } dst;
} add;
```

The existence of the `add` keyword in a filtering rule is flagged with the `check_add` member. This keyword wait for source or destination IP addresses associated with the tables required to store those addresses. Those IP addresses can be assigned to different tables, i.e. the source address can be associated with a table and the destination address with another table. Thus, this structure allocates two sub structures `src` and `dst`. Each sub structure contains a variable `check_[src|dst]` (that flags the existence of source and / or destination IP address), `[src|dst]_tablname` (name of the table used to store the IP address) and `[src|dst]_tbl` (a pointer on the table data).

In the file `pf.c`, a function `pf_save_addrs` has been added:

```
void pf_save_addrs(struct pf_rule **rule, struct pf_pdesc
     *pd)
```

This function extracts the source and the destination addresses of the packet from the `pf_pdesc` structure and adds them into the tables specified on the `pf_rule` structure.

This function is called four times respectively in `pf_test_tcp()`, `pf_test_udp()`, `pf_test_icmp()` and `pf_test_other()` for covering the different levels of the IP protocols. Those four functions are called for the first packet of a connection. For the following packets, PF maintains a state for the corresponding flow. Here is an extract of `pf_test()` (`pf_test6()` for IPv6) function which handles the packet matching for TCP:

```
1 switch (packet_protocol)
2 case TCP: {
3    pf_test_state_tcp();
4    if (existing state)
5       update state
6    else
7       pf_test_tcp()
8    break;
9 }
```

The function `pf_test_tcp()` is thus called for the first packet of a TCP connection (line 7). This is where the instrumentation, i.e. the saving of the addresses, must be done. In `pf_test_tcp()`:

```
if (add option) {
  pf_save_addrs (&r, pd);
}
```

C. Modifications on pfctl

`Pfctl` is the userland tool to interact with PF. Its main function is to enable, disable and check a ruleset stored in a configuration file `pf.conf`. It also permits to interact with some PF objects on the fly (for example adding an IP address to a table, list IP addresses of a table, load an anchor etc.).

PF makes use of Lex and Yacc to check a ruleset. The Yacc grammar has been modified to take the `add` keyword and its options (`ipsrc` and `ipdst`) in account.

In the file pfctl_parser.c, the `add` keyword has to be taken into account in the function `print_rule()` that displays the current ruleset using the `pfctl` command:

```
1 if (r->add.check_add != 0) {
2    if (r->add.src.check_src != 0 && r->add.dst.check_dst
       == 0)
3       printf("add ipsrc <%s>", r->add.src.src_tblname);
4    if (r->add.src.check_src == 0 && r->add.dst.check_dst
       != 0)
5       printf("add ipdst <%s>", r->add.dst.dst_tblname);
6    if (r->add.src.check\_src != 0 && r->add.dst.check_dst
       != 0)
7       printf("add ipsrc <%s> ipdst <%s>", r->add.src.
            src_tblname, r->add.dst.dst_tblname);
8 }
```

Line 1, a check is performed on `r` (current rule) to see if the `add` keyword is present on the rule through the member `check_add`. If present, a check to see if the source IP (line 2), destination IP (line 4) or both (line 6) have been specified for logging.

VI. FIELDS: NEW NSM HEURISTICS

A. IP Blacklist

Some websites, such as `spamhaus`, `uce-protect` and so on, provide blacklists filled in by addresses that have been judged compromised: botnet, spam, etc... These lists can be used to block an incoming connection to the internal network hosts. Connections incoming from these addresses are blocked by the firewall but connections made from internal hosts to these addresses are not controlled. Logging these connections is interesting because it means that the host has a suspect behavior.

The FIELDS controller can detect these connections and then automatically launch a comprehensive capture of these internal hosts for a further analysis. Since those internal hosts have a suspect behavior, a deep analysis of their traffics might tell us if they are infected, and eventually, with which malware.

A simple bash script has been made to automatically download updated blacklists and load them into FIELDS. This is done without reloading the entire policy of FIELDS thanks to `pfctl`.

As pf can handle lots of distinct tables, the system is very modular. The comprehensive capture can be optional: the first suspect packet is always automatically logged, and the address added into the desired table. This table just can be used to know which host has match a given rule. Example:

```
1 table <blacklist>
2 table <compromised_hosts>
3 table <internals_hosts>
4
5 pass in log on $if from <internals_hosts> to <blacklist>
      no state add ipsrc <compromised_hosts>
6
7 pass in log on $if from <compromised_hosts> to any no
      state
8 pass in log on $if from any to <compromised_hosts> no
      state
```

These rules allow `pf` to launch a comprehensive capture on any internals hosts of our network which try to connect to a host from the blacklist. There are two stages is the FIELDS process:

- Addition of the suspected host that talks to a blacklisted address to the table `<compromised_hosts>` (line 5) ;
- Comprehensive capture of each packet from (line 7) and destined to (line 8) the suspects stored in `<compromised_hosts>`.

B. IP Sinkhole

FIELDS can modelize an IP sinkhole. The collection of internal subnets is `$FEDE`, RFC1918 addresses are stored in `$sinkhole`. The `em0` and `em1` interfaces are linked to the Network TAP. FIELDS is listening on them.

```
1 pass in log on { em0 em1 } from $FEDE to $sinkhole no
      state add ipsrc <univ_to_sinkhole>
2 pass in quick log on { em0 em1 } from <univ_to_sinkhole>
      to any no state
3 pass in quick log on { em0 em1 } from any to <
      univ_to_sinkhole> no state
4 pass quick on { em0 em1 } from $FEDE to $FEDE no state
```

The traffic from the internal network to the sinkhole is logged and the sources are added to `<univ_to_sinkhole>` (line 1). Traffic from (line 2) or destined to (line 3) hosts belonging to `<univ_to_sinkhole>` is dumped and the packet evaluation ends ("quick" keyword). Packets from internal to another internal network pass without any log (line 4).

C. DNS Sinkhole

FIELDS can detect when an internal host attempts to connect to a blacklisted address, as shown in Figure 4. But,

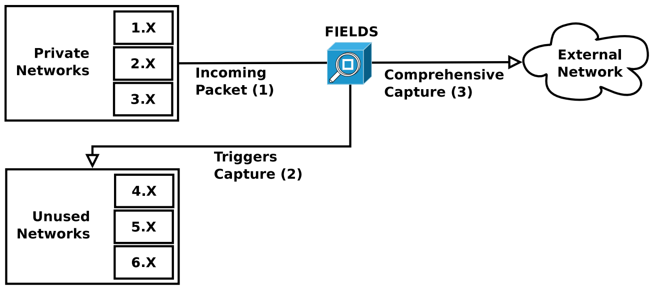


Figure 4. FIELDS IP / DNS Sinkhole for 192.168.X.X/24

the connections can be done with malware domains. To cover such connections a DNS Sinkhole is available. Thanks to the DNS Sinkhole, our DNS gives a false IP address as a response to a DNS request for all the domains which are considered as compromised. Our DNS Sinkhole lists the malware domains e.g. the domains provided by dedicated websites covering well-know malware domains. Thanks to our DNS Sinkhole, the host that will try to establish a communication with a suspected domain will fail, but it will generate an odd traffic which will be recognized by FIELDS. This false IP address has to be a private IP address (rfc 1918), and will be recognized as a part of the sinkhole. The rule concerning the DNS has been written after the rule of the sinkhole to be sure it takes the priority. Otherwise, we would not be able to see the difference between a host making a request for a blacklisted DNS and a host trying to contact a host in the sinkhole. A FIELDS policy enables to log all the hosts that attempt to connect to this false IP address. It probably means that this host is compromised by some malware and needs an antivirus/IDS analysis.

The DNS rules are close to the blacklist rules except we add our false IP address returned by the DNS to 1) the table `<compromised_hosts>` or 2) another table.

D. Using the transport layer: Service sinkhole

Another variant of sinkhole is to rely on transport to determine which service on non-existant hosts is tried to be accessed. A Windows oriented sinkhole has been modelized to catch hosts that try to access Windows ports:

- 137: NetBIOS name service (UDP) NetBIOS-sn lookups for "`gethostbyaddr()`" function;
- 138: NetBIOS datagram (UDP) non connected messages exchange;
- 139: NetBIOS Session (TCP) Windows File and Printer Sharing;
- 445: CIFS (TCP) used for the Common Internet FileSystem resources sharing protocol;
- 1433: SQLServer (TCP) Microsoft SGBD.

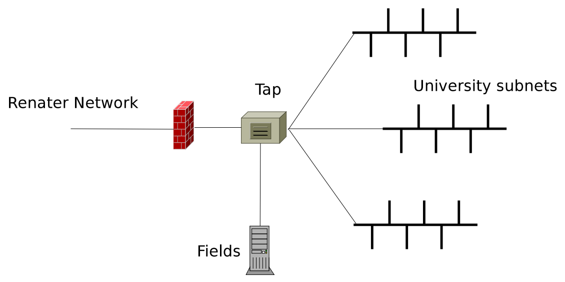


Figure 5. FIELDS integration topology

```
pass in log on { em0 em1 } proto udp from $FEDE to
     $sinkhole port { 137 138 139 } no state add ipsrc <
     to_winsinkhole>
pass in log on { em0 em1 } proto tcp from $FEDE to
     $sinkhole port { 445 1433 } no state add ipsrc <
     to_winsinkhole>
```

E. Bruteforce

The extensibility of FIELDS easily enables to cover the bruteforce traffic attempting to discover the network resources. A FIELDS policy enables to support the network bruteforce. Thus, FIELDS captures all the traffic of a host that does the network scanning. Example:

```
pass in log on $if from <internals_hosts> to <
     internals_hosts> port { 22 445 } keep state (max-src-
     conn-rate 3/10, overload <bruteforce> flush global)
```

This rules add the address of any internal host that attempts a bruteforce attack on ssh and samba ports into the table `<bruteforce>`.

VII. Experimentations

As shown in Figure 5, FIELDS is placed behind a Network TAP. This device copy traffic passing through the firewall and the external router in a non-intrusive way. As a consequence, in case of performance lack / bug, the real network is not affected. FIELDS processes all the traffic between 1) the Internet and the University of Orleans, 2) each university pole, and 3) university poles and the DMZ. FIELDS contains only rules relative to the EDS/IDS pre-processing VI. Remember that the idea is not to supply a complete EDS/IDS but an intelligent pre-processor for exhaustively collecting only the traffic related to suspected hosts (and not only the suspected traffic).

As explained in V, FIELDS captures the traffic from lists of IPs stored in tables. When an IP is added to a table, a timestamp of the current date is included. If the IP is already stored in the table, PF only updates the record with the current date. In order to avoid having growing only tables, FIELDS also must be able to delete a record from a table. A deletion can be carried out manually when the host has been checked as safe by the security team. But, PF also provides automation for the deletion. For example, if there is no malicious activity involving this address for a given period of time, then FIELDS deletes the entry from the table. In practice, FIELDS uses the program `expiretable` (which is shipped with PF) for removing an address after a period of time (2 days on this study) without any illegal activity.

`Pflog` is used to log all the traffic of the selected hosts. In the proposed experimentation, these `pcap` files are processed with argus in order to find flow anomalies first, then with snort in order to detect more advanced attacks. Obviously, other IDS and network analysis are supported by FIELDS.

FIELDS can detect outgoing attacks: if an intruder already owns an host in our internal network, his ignorance of the network structure, his attempts to discover the network, will make his host detected by FIELDS (`sinkhole`). If he is detected, all the traffic of the suspected host will be logged, which will make the attack a lot easier to analyze. The flexibility of the tables of `pf` allow us to isolate potentially harmful traffic such as an `nmap` scan from an internal host. It's possible to isolate these IPs in a table, which will be read regularly. As FIELDS detects the traffic which is not supposed to happen, a host misconfiguration can also be detected.

A. Detection results

This section presents the results of the FIELDS experimentation on a real network of about 30 000 users for a period of 2 months. The volumetric for such a network is about 1,5 TB per day of raw PCAP for the external link (roughly speaking, link from your network to the Internet). The analysis on each host is performed in two rounds: a scan with Snort on the captured PCAP and a local antiviral analysis (if Snort returns nothing). FIELDS has been installed on common hardware (Intel XEON processor / 4 cores and 8GB of RAM). Table I gives the number of IP caught by FIELDS. Blacklist has only 10 hosts caught. Those hosts are infected by `conficker` and tries to reach servers handled by Microsoft. No bruteforce attempts have been detected. It's a false negative because there are bruteforce attempts but they are too slow or they come from distributed sources. As a consequence, they do not raise an alert based on connection rate as exposed in VI-E. Majority of the hosts has been caught with sinkhole heuristics. Table I gives the proportion of:

- infected: malicious payload detected on captured PCAP or malicious programs detected on hosts ;
- misconfigured: bad settings at network level on host (WINS server, DNS settings, routing etc.) ;
- unconfirmed: traffic and antiviral analysis did not give any results ;
- unchecked: traffic analysis did not give any results and antiviral analysis has not been performed ;

Table I
DETECTION RESULTS

Heuristic	Number of IP caught	Infected	Misconfigured	Unconfirmed	Unchecked
Blacklist	10	90%	0%	10%	0%
Bruteforce	0	0%	0%	0%	0%
IP sinkhole	163	22%	11%	32%	35%
DNS sinkhole	45	82%	0%	18%	0%
Windows sinkhole	44	64%	15%	21%	0%

The only false positive for the blacklist is a GET request from the security team for testing purpose. IP sinkhole has numerous unchecked hosts because it is difficult to request local antiviral tests on those hosts. Moreover, the IP sinkhole is prone to false positive because there are a lot of situations that can lead to send packets to sinkhole destinations (essentially typos in command). The DNS sinkhole is very effective, false positive are performed by people accessing malicious websites (i.e. blacklisted DNS domain name) through their web browser. Snort did not detect malware on traffic of DNS sinkholed hosts, only antiviral analysis gave results.

B. Network forensic

The amount of the whole PCAP captured is 46 Gb for a day. FIELDS enables to retain a lot more traffic than an exhaustive capture on the outgoing link (about 1,5 TB a day). This is great help for a forensic purpose when a request comes from upper IT security authority (such as RENATER CERT).

VIII. CONCLUSION AND PERSPECTIVES

FIELDS provides some key features in network security monitoring:

- Modelization and improvements of malicious flows detection algorithms;
- Efficient pre-processor of flows based on a reduced improvement of a classical Packet Filter;
- Hybrid collector of network traffic helping network forensic process;
- Non intrusive monitoring ;

A large scale experimentation shows the efficiency of the approach and demonstrates the ability of FIELDS to take the best of existing security tools (antivirus, NIDS). Thus, the approach enables to confirm or to unconfirm the risk. The results encourage us to follow FIELDS in several directions. First, FIELDS eases the modelization of other heuristics. Second, it can help to prevent the propagation of the intrusions through firewall/network configurations since it provides relevant information about the network compromission. Finally, an efficient forensic could be proposed through correlations of different tools since FIELDS provides the malicious traffic.

REFERENCES

[1] M. Ranum, A. Lambeth, and E. Wall, “Implementing a generalized tool for network monitoring,” in *In Proc. 11th Systems Administration Conference (LISA)*, 1997.

[2] S. A. Mehdi, J. Khalid, and S. A. Khayam, “Revisiting traffic anomaly detection using software defined networking,” in *Proceedings of the 14th international conference on Recent Advances in Intrusion Detection*, ser. RAID’11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 161–180.

[3] R. Sommer and V. Paxson, “Enhancing byte-level network intrusion detection signatures with context,” in *In Proc, 10th Conference On Computer And Communications Security*. ACM, 2003, pp. 262–271.

[4] A. Caglayan, M. Toothaker, D. Drapaeau, D. Burke, and G. Eaton, “Behavioral analysis of fast flux service networks,” in *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies*, ser. CSIIRW ’09. New York, NY, USA: ACM, 2009, pp. 48:1–48:4.

[5] H.-G. Lee, S.-S. Choi, and Y.-S. L. H.-S. Park, “Enhanced sinkhole system by improving post-processing mechanism,” *Future Generation Information Technology*, pp. 469–480, 2010.

[6] B. Greene and D. McPherson, “Isp security: Deploying and using sinkholes,” *NANOG talk, http://www. nanog. org/mtg-0306/sink. html*, 2003.

[7] R. Bejtlich, *The Tao of network security monitoring: beyond intrusion detection*. Addison-Wesley Professional, 2004.

[8] ——, *Extrusion detection: security monitoring for internal intrusions*. Addison-Wesley Professional, 2005.

[9] C. Sanders, *Practical packet analysis*. No Starch Press, 2007.

[10] B. Sunny and K. Krishan, “An experimental analysis for malware detection using extrusions,” ser. ICCCT. IEEE, 2011, pp. 474 – 478.

[11] T. Ankita and S. R. /, “Optimization of snort for extrusion and intrusion detection and prevention,” *International Journal of Engineering Research and Applications, Vol. 2, Issue 3*, pp. 1768–1774, 2012.

[12] D. Hartmeier, “Design and performance of the openbsd stateful packet filter (pf),” [retrieved: 07, 2012], 2002.

Towards Assisted Remediation of Security Vulnerabilities

Gabriel Serme*, Anderson Santana De Oliveira*, Marco Guarnieri† and Paul El Khoury‡
*SAP Research
Sophia Antipolis, France
{name.lastname}@sap.com
†Dept. of Information Technology
and Mathematical Methods
University of Bergamo, Italy
0guarnieri.marco0@gmail.com
‡SAP AG
Walldorf, Germany
paul.el.khoury@sap.com

***Abstract*—Security vulnerabilities are still prevalent in systems despite the existence of their countermeasures for several decades. In order to detect the security vulnerabilities missed by developers, complex solutions are undertaken like static analysis, often after the development phase and with a loss of context. Although vulnerabilities are found, there is also an absence of systematic protection against them. In this paper, we introduce an integrated Eclipse plug-in to assist developers in the detection and mitigation of security vulnerabilities using Aspect-Oriented Programming early in the development life-cycle. The work is a combination of static analysis and protection code generation during the development phase. We leverage the developer interaction with the integrated tool to obtain more knowledge about the system, and to report back a better overview of the different security aspects already applied, then we discuss challenges for such code correction approach. The results are an in-depth solution to assist developers to provide software with higher security standards.**

***Keywords*-Security, AOP, Software Engineering, Static Analysis, Vulnerability Remediation**

I. Introduction

After a decade of existence, cross-site scripting (XSS), SQL Injection and other of types of security vulnerabilities associated to input validation can cause severe damage once exploited. To analyze this fact, Scholte et al. [1] conducted an empirical study that shows that the number of reported vulnerabilities is not decreasing.

While computer security is primarily a matter of secure design and architecture, it is also known that even with best designed architectures, security bugs will still show up due to poor implementation. Thus, fixing security vulnerabilities before shipment can no more be considered optional. Most of the reported security vulnerabilities are leftovers forgotten by developers, thought to be some benign code. Such kind of mistakes can survive unaudited for years until they end up exploited by hackers.

The software development lifecycle introduces several steps to audit and test the code produced by developers in order to detect the security bugs, such as code review tools for early detection of security bugs to penetration testing. The tools are used to automate some tasks normally handled manually or requiring complex processing and data manipulation. They are able to detect several of errors and software defects, but developers have to face heterogeneous tools, each one with a different process to make it run correctly, and they have to analyze the results of all the tools, merge them and fix the source code accordingly. For instance, code scanner tools are usually designed to be independent from the developers' environment. Therefore, they gain in flexibility but loose comprehensiveness and the possibility to interact with people having the experience on application code. Thus, tools produce results that are not directly linked to application defects. It is the case for example for code scanner tools triggering several false positives, which are not actual vulnerabilities.

The contributions of this paper are twofold. First, we focus on static code analysis, an automated approach to perform code review integrated in developer's environment. This technique analyzes the source code and/or binary code without executing it and identifies anti-patterns that lead to security bugs. We focus on security vulnerabilities caused by missing input validation, the process of validating all the inputs to an application before using it. Although our tool handles other kinds of vulnerabilities, here we discuss on three main vulnerabilities caused by missing input validation, or mis-validation of the input: cross-site scripting (also called XSS), Directory Path Traversal and SQL Injection. Second, we provide an innovative assisted remediation process that employs Aspect-Oriented Programming for semi-automatic vulnerability correction. The combination of these mechanisms improves the quality of the software with respect to security requirements.

The paper is structured as follows: Section II presents the overall agile approach to conduct code scanning and correct vulnerability during the development phase. Then, Section III presents the architecture we adopt to combine the static analysis with the code correction component. The Section IV describes the static analysis process with its integration in the developers' environment. Then, we explain techniques for assisted remediation along with pros and cons in Section V. Finally, we discuss the advantages of our approach compared to related work in Section VI, and we conclude in Section VII.

II. An agile approach

Agile approaches to software development require the code to be refactored, reviewed and tested at each iteration of the development lifecycle. While unit testing can be used to check functional requirements fulfillment during iterations, checking emerging properties of software such as security or safety is more difficult. We aim to provide each developer with a simple way to do daily security static analysis on his code. That would be properly achieved by providing a security code scanner integrated in the development environment, i.e., Eclipse IDE in this case, and a decentralized architecture that allows the security experts to assist the developers in any of the findings. Typically, that would include verifying false positives and correspondingly adjusting the code scanner test cases, or assisting in reviewing the solutions for the fixes. It brings several advantages over the approach in which the static analysis phase stays only at the end. The expertise of the context in which the code was developed lies in development groups. Therefore, the interaction between development team and security experts is faster with less effort in finding and applying corrections on the security functionalities. The experts provide support on a case basis for a better tuning of false positive detection across teams and reducing final costs of maintenance: solving security issues into the development phase can reduce the number of issues that the security experts should analyze at the end.

Maintaining the separation of roles between the security experts performing the code scanning and the team members developing the application raises a critical complication, typically, from a time perspective, due to the human interaction between security experts and developers. If such an approach would have to scale to what most of the agile approaches describe, the amount of iteration between developers and experts would need to be reduced. That could be reduced by up-skilling the developers and reducing the interaction between them and the security experts for the analysis of the security scans of the project, which is simplified by the introduction of our tool.

Our incentive is to harvest the advantages acquired by using our approach in an agile and decentralized static analysis process early in the software development lifecycle. It raises security awareness for the developers at the development time and reduces maintenance costs. A tool covering the previous needs should fulfill several requirements:

- *easy-to use* for users non-experts in security
- *domain specific* with integration into developers' daily environment, to maximize adoption and avoid additional steps to run the tool
- *adjustable* to maximize project knowledge and reduce false positives and negatives
- *reflexive* to adjust accuracy of the scan over time, with collaborative feedbacks for example
- *supportive* to assist developers in correcting and understand issues.
- *educative* to help developers understanding errors, steps to correct existing error, and techniques to prevent future vulnerability

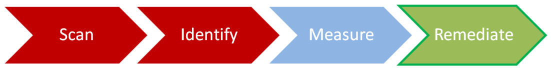


Figure 1. Vulnerability remediation process. The red corresponds to the static analysis component. The green one corresponds to the remediation component. The blue one corresponds to assisted processing

We have developed an Eclipse Plugin, presented in [2], made of components leveraging decentralized approach for static analysis. It gives direct access to detected flaws and global overview on system vulnerabilities. The developer analyzes its code and review vulnerabilities when necessary.

Figure 1 presents the interaction between the two phases: the static analysis phase allows scanning the code in order to identify and classify the different vulnerabilities found. It is described in details in Section IV. The measurement is performed directly by developers who decide what to remediate by undertaken actions, with support from our second component. The full remediation process is given in Section V .

III. Architecture

Figure 2 represents the architecture of our prototype. First of all, we consider two main stakeholders involved in the configuration and usage of the prototype. Security experts and developers regroup different profiles whose goal is to provide and configure the knowledge database in order to avoid false positives and negatives, and to provide better accuracy during the analysis phase. They have two main tasks. First, they update the knowledge base, adding to its classes or methods that can be considered as trusted for one or more vulnerabilities. Second, the knowledge database receives feedback from analysis on possible trusted objects for one or more security vulnerabilities; they must analyze them more in detail and, if these objects are really trusted they tag them as trusted into the knowledge base. We better explain the different concepts and tasks in Section IV.

The second role is the developer, interacting directly with the static analysis engine to verify vulnerabilities in application, code and libraries under its responsibility. The

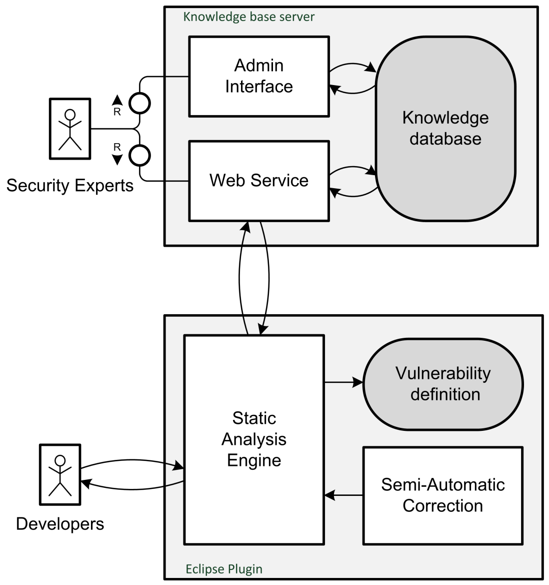


Figure 2. Architecture

developer at this stage can be naive, therefore with no focus on complexity of security flow. The knowledge base is shared among developers. It contains all the security knowledge about trust: objects that do not introduce security issues into the code. Security experts and developers with understanding of security patterns maintain and keep under control the definitions used by all developers in an easy way using one admin web application or some web-services. In this way the code scanner testing rules are harmonized for the whole application or even on a project-basis. The knowledge base allows developers to run static analysis that is perfectly adapted to the context of their project.

In industrial scale projects, daily scans are recommended. In order to facilitate this task, we provide a plugin for Eclipse that uses an abstract syntax tree (AST) generated by the JDT compiler - the compiler that Eclipse provides as part of the Java Development Tools platform, to simplify the static analysis process. The plugin accesses the knowledge database via web-services making it possible to each developer to run independently the code scanner. We detail its components in the next section.

IV. Static analysis

Static analysis can report security bugs even when scanning small pieces of code. Another family of code scanners is based on dynamic analysis techniques that acquire information at runtime. Unlike static analysis, dynamic analysis requires a running executable code. Static analysis scans all the source code while dynamic analysis can verify certain use cases being executed. The major drawback of static analysis is that it can report both false positives and false negatives. The former detects a security vulnerability that is not truly a security vulnerability, while the latter means that it misses to report certain security vulnerabilities. Having false negatives is highly dangerous as it gives one sensation of protection while vulnerability is present and can be exploited, whereas having false positives primarily slows down the static analysis process. Modern static analysis tools, similarly to compilers, build an abstract syntax tree that represents the abstract syntactic structure of the code from the source code and analyze it.

A. Static Analysis Process

In a nutshell, our process allows developers to run a check on their code to uncover potential vulnerabilities by checking for inputs that have not been validated. It finds information flows connecting an entry point and exit point that does not use a trusted object for the considered vulnerabilities. The algorithm uses an abstract syntax tree of the software in conjunction with the knowledge base to identify the vulnerable points. The Figure 3 presents the different analysis steps performed from the moment developer presses the analysis button to the display of results.

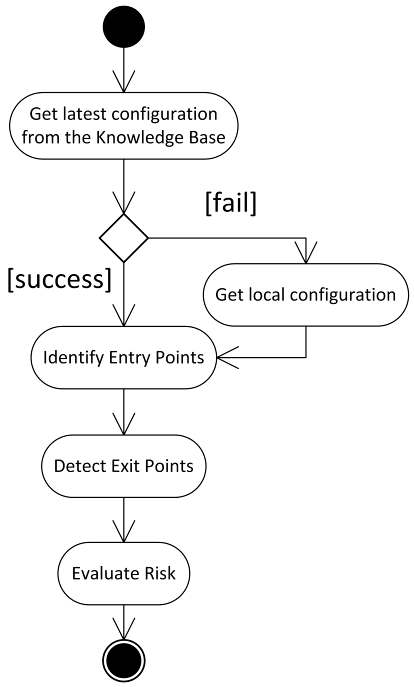


Figure 3. Static Analysis Activity Diagram

The static analysis works on Document Object Model generated by the Eclipse JDT component able of handling all constructs described in the Java Language Specification [3]. The static analysis process is described as follows:

- The engine contacts the knowledge database in order to retrieve the up-to-date and most accurate configuration from the shared platform. If the developer cannot retrieve the configuration, it can still work independently with the latest local configuration.

- The process identifies all *entry points* of interest in the accessible source code and libraries. The analysis is based on the previously mentioned AST. We are gathering the different variables and fields used as well as the different methods. We apply a first filter with pattern-matching on the potential *entry points*: a method call or a new object instantiation might be tagged as returning trusted inputs.
- For each *entry point* the control flow is followed to create the connections between methods, variables and fields to discover all the *exit points*. For instance, the engine visits assignments, method invocations and construction of new objects with the variables and fields detected during the entry point gathering.
- Once the different *exit points* have been collected, we evaluate the risk of having security vulnerabilities in the code. We check for an absence of validation in the flow for the different kinds of vulnerabilities. For instance, if the flow from an entry point to an exit point passes through a method or a class, which is known to validate SQL input, the flow is tagged as trusted for this specific vulnerability. Of course, the tag runs from the moment where the method validates for the vulnerability to the moment of a novel composition with potential vulnerable code, or until an *exit point*.

B. *Multiple vulnerability analysis*

In the previous section, we have presented the global analysis process. In this section, we discuss more in-depth the notion of trusted object and vulnerability propagation for the different vulnerabilities we address. The Listing 1 presents some source code vulnerable to cross site scripting. The vulnerability propagates from the request parameter to the object *query*, which is then written in the response. The problem of identifying security vulnerabilities caused by errors in input validation can be translated to finding an information flow connecting an *entry point* and an *exit point* that does not use a *trusted object* for the considered vulnerabilities.

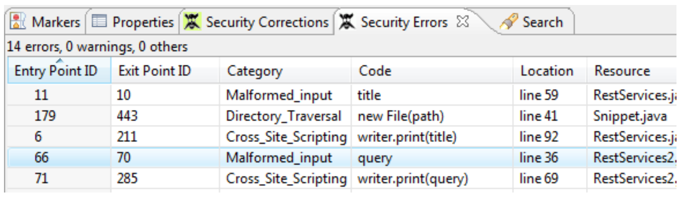
```
/** This servlet proposes XSS example. */
public class EchoServlet extends HttpServlet {
  protected void doGet(HttpServletRequest req,
      HttpServletResponse resp) {
    PrintWriter writer = resp.getWriter();
    String query =
        req.getParameter("query");

    resp.setContentType("text/html");
    writer.print("<html><h1>Results for ");
    writer.print (query);
    writer.print("</h1></html>");
    writer.flush();
    writer.close();
    }
}
```

Listing 1. Vulnerability propagation of a cross site scripting

We define an *input* as a data flow from any external class, method or parameter into the code being programmed. We also define as *entry point* any point into the source code where an *untrusted input* enters to the program being scanned, like the *query* input from Listing 1. In an analogous way we define as *output* any data flow that goes from the code being programmed into external objects or method invocations. Our approach relies on our *trusted object* definition, which impacts the detection accuracy. A *trusted object* is a class or a method that can sanitizes all the information flow from an *entry point* to an *exit point* for one or more security vulnerabilities. We implemented the trust definitions into the centralized knowledge base presented in the previous section. The knowledge database represents the definitions using a trusting hierarchy that follows the package hierarchy.

Security experts can tag classes, packages or methods as trusted for one or more security vulnerabilities, accordingly to their analysis, feedbacks from developers or static analysis results. Obviously defining a trusted element in the trust hierarchy also adds all the elements below it: trusting a package trusts all the classes and methods into it and trusting a class trusts all the fields and methods in it. A trusted object can sanitize one or more security vulnerabilities (e.g., sanitization method can be valid for both SQL Injection and cross site scripting). This approach enables developers and security experts to define strong trust policies with regards to the system they are securing.

Defining a *trusted object* is a strong assertion as it taints a given flow as valid and free for a given vulnerability. The definition process to trust a class, a package or a method must be rigorous: it influences the risk evaluation accuracy. The object must not introduce a specific vulnerability into the code. This is the reason why developers report feedback and security experts take the decision. The experts can also analyze, manage and update the base, if the class, package or method is considered trusted. This phase allows system tuning that is related to a given organization and leads to fewer false positives while ensuring no false negatives.



14 errors, 0 warnings, 0 others

Entry Point ID	Exit Point ID	Category	Code	Location	Resource
11	10	Malformed_input	title	line 59	RestServices.j
179	443	Directory_Traversal	new File(path)	line 41	Snippet.java
6	211	Cross_Site_Scripting	writer.print(title)	line 92	RestServices.j
66	70	Malformed_input	query	line 36	RestServices2.
71	285	Cross_Site_Scripting	writer.print(query)	line 69	RestServices2.

Figure 4. Code Analysis result

The detected vulnerabilities (Figure 4 gives an example of analysis result in the tool) are mainly caused by lack of input validation, namely SQL Injection, Directory Path Traversal and Cross Site Scripting. The engine detects also a more general Malformed Input vulnerability that represents any input that is not validated using a standard implementation. The engine can be easily extended to support new kinds of vulnerabilities caused by missing input validation. One needs to add the definition of the new vulnerability to the centralized knowledge base (and, if exist, adding trusted objects that mitigate it), and creating a new class extending an interface, that implements the checks to be done on the result of the static analysis to detect the vulnerability.

V. Assisted Remediation

Performing static analysis is yet integrated in quality processes in several companies. But, the actual identification of vulnerabilities does not mean they are correctly mitigated. Given this problem, we can have several approaches: (i) refactoring the code, (ii) applying a proxy in inbound and outbound connections, and finally the solution we adopted, (iii) to generate protection code linked to the application being analyzed.

Software refactoring involves the developer into understanding the design of its application and the potential threats, to manually rewrite part of the code. The refactoring improves the design, performance and manageability of the code, but is difficult to address. It costs time and is error prone. Up to six distinct activities have been observed in [4] from identification to verification of refactoring. The impacted code is generally scattered over the application, and some part can be left unchecked easily. This can lead to an inconsistent state where the application does not reflect the intended goal. In terms of vulnerability remediation, the software refactoring is one of the most powerful due to the flexibility in terms of code rewriting and architecture evolution.

The proxy solution is equivalent to a gray-box approach, with no in-depth visibility of internal processes. It can be heavy to put in place, especially when the environment is under control of a different entity than the development team. For instance, on cloud platforms, one can deploy its application but has limited management on other capabilities, leading to the impossibility to apply filter on the application. The lack of flexibility and the absence of small adjustments make it complicated to adopt at the development phase.

In this work we provide inline protection with the application. This solution has several advantages, but also brings new limitations due to the technology we use: Aspect-Oriented Programming paradigm (AOP) [5], which is a paradigm to ease programming concerns that crosscut and pervade applications. In the next section, we describe our methodology and provide a comprehensive list of advantages and drawbacks.

A. Methodology

The approach comprises the automatic discovery of vulnerability and weaknesses in the code. In addition, we integrate a protection phase tied to the analysis process which guides developers through the correct and semi-automatic correction of vulnerabilities previously detected. It uses information from the static analysis engine to know what vulnerabilities have to be corrected. Then it requires inputs from the developer to extract knowledge about the context, like in Figure 5. These steps allow gathering places in the code where to inject security correction. The security correction uses AOP. The goal is to bring proper separation of concerns for cross cutting functionalities, such as security. Thus, code related to a concern is maintained separately from the base application. The main advantage using this technology is the ability to intervene in the control flow of a program without interfering with the base program code.

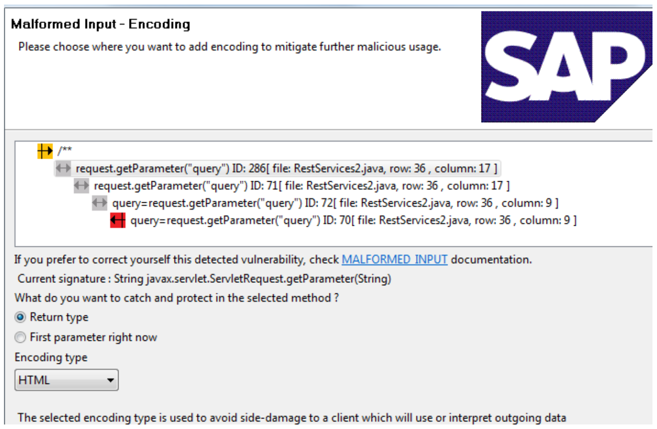
Vulnerability	Origin	Potential Remediation
Cross-Site Scripting	Server does not validate input coming from external source	Validate input and filter or encode properly the output depending on the usage: the encoding differs from HTML content to Javascript content for example
SQL Injection	Server does not validate input and use it directly in a construct of a SQL Query	Use a parameterized query or a safe API. Escape special characters. Validate the input used in the construction of query
Directory Path Traversal	Application server is misconfigured, or the file-system policy contains weaknesses	Enclose the application with strict policies, that restrict access to the filesystem by default. Filter and validate the input prior to direct file access
Other malformed input	Misvalidation	Validate input, determine the origin and possible manipulation from externals

Table I
LIST OF DETECTED VULNERABILITIES WITH POTENTIAL ORIGIN AND POTENTIAL REMEDIATION.

The list of vulnerability we cover principally are in Table I. The Table highlights the potential origin vulnerabilities and some of known remediation techniques. These vulnerabilities are known and subject to high attention. For instance, we can retrieve them in the OWASP Top Ten [6] for several years now, but also in the MITRE Top 25 Most Dangerous Software Errors [7]. Albeit several approaches exist to remediate the vulnerabilities, we are considering mainly escaping and validation to consistently remediate the problems with the aspect-oriented technique.

By adopting this approach, we reduce the time to correct vulnerabilities by applying semi-automatic and pre-defined mechanisms to mitigate them. We use the component to apply protection code which is mostly tangled and scattered over an application.

Correcting a security vulnerability is not trivial. Different refactoring are possible depending on the issue. For instance, the guides for secure programming advises SQL prepared statement to prevent SQL Injection. But, developers might be



Malformed Input - Encoding

Please choose where you want to add encoding to mitigate further malicious usage.

```
/**
  request.getParameter("query") ID: 286[ file: RestServices2.java, row: 36 , column: 17 ]
    request.getParameter("query") ID: 71[ file: RestServices2.java, row: 36 , column: 17 ]
      query=request.getParameter("query") ID: 72[ file: RestServices2.java, row: 36 , column: 9 ]
        query=request.getParameter("query") ID: 70[ file: RestServices2.java, row: 36 , column: 9 ]
```

If you prefer to correct yourself this detected vulnerability, check MALFORMED INPUT documentation.

Current signature : String javax.servlet.ServletRequest.getParameter(String)

What do you want to catch and protect in the selected method ?

[✓ Return type]

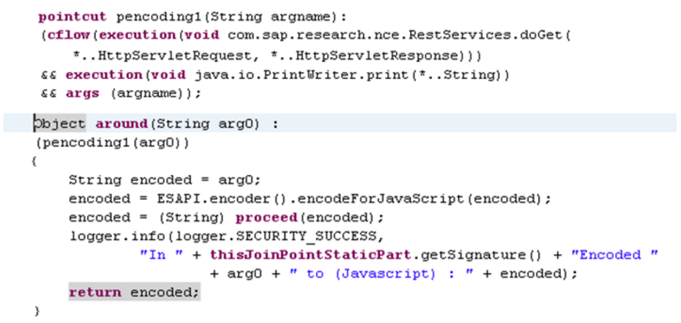
First parameter right now

Encoding type: HTML

The selected encoding type is used to avoid side-damage to a client which will use or interpret outgoing data

Figure 5. Gathering context for vulnerability protection

constrained by their frameworks to forge SQL queries themselves. Therefore, developers would try another approach such as input validation and escaping of special characters.



```
pointcut pencoding1(String argname):
 (cflow(execution(void com.sap.research.nce.RestServices.doGet(
     *..HttpServletRequest, *..HttpServletResponse)))
 && execution(void java.io.PrintWriter.print(*..String))
 && args (argname));

Object around(String arg0) :
(pencoding1(arg0))
{
     String encoded = arg0;
     encoded = ESAPI.encoder().encodeForJavaScript(encoded);
     encoded = (String) proceed(encoded);
     logger.info(logger.SECURITY_SUCCESS,
             "In " + thisJoinPointStaticPart.getSignature() + "Encoded "
                   + arg0 + " to (Javascript) : " + encoded);
     return encoded;
}
```

Figure 6. Example of correction snippet generated for a malformed input

We assist developers by proposing them automated solutions. For the previously mentioned correction, our integrated solution would propose to mitigate the vulnerability with an automatic detection of incoming, unsafe and unchecked variables. The developer does not need to be security expert to correct vulnerabilities as our approach provides interactive steps to generate AOP protection code, like in Figure 6. Although semi-automation simplifies the process to introduce protection code, the technique can introduce several side-effects if the developers are not following closely what is generated. The plug-in gives an overview for the developer of all corrected vulnerabilities, allowing him to visually manage and re-arrange them in case of need. Currently, the prototype does not analyze interaction between the different protection code generated. By adopting this approach, we allow better understanding from a user point of view of the different vulnerabilities affecting the system, and we guide the developer towards more compliance in its application. The protection code can be deployed by security expert teams and change without refactoring.

B. *Constraints from Aspect-Oriented Programming*

The usage of AOP in the remediation of vulnerability bring us more flexibility. One can evolve the techniques used to protect the application, by switching the process to resolve a problem, making the security solution independent from the application. But this approach also brings us some limitations we discuss in this section.

Firstly, the language is designed to modify the application control flow. One of the limitations we have is related to the deep modification we need to perform in order to replace partially a behavior. For example, suppose a SQL query written manually in the application we would like to validate. We are able to weave validation and escaping code, but we can hardly modify the application to construct a parameterized query.

Secondly, the aspects cover the application in the whole. When more than one aspect is involved, the cross-cutting concerns can intersect. Therefore, we need to analyze aspect interaction and prevent an annihilation of the behavior we intended to address.

Thirdly, the evolution of the program leads to a different repartition of vulnerabilities. The vulnerabilities are detected after the static analysis phase. We are not addressing yet this problem of evolution to maintain the relation between the aspects and the application. This differs from the *fragile pointcut* problem inherent of aspect using pointcut languages referring to the syntax of the base language: the evolution affects the application as a whole, by introducing new entry points and exit points that need to be considered, or introducing methods that validate a flow for a given vulnerability.

The fourth constraint is that aspects have no specific certification. The actual protection library is defined globally, but applied locally, with a late binding to the application. The protection code is the same everywhere, but we put strong trust in the protection library by assuming that aspects are behaving properly with the actual modification of the flow to mitigate the vulnerability.

Finally, the fifth constraint is user acceptance. Since the developers rely on cross cutting solution, the code itself does not reflect the exact state of the application. The point where the aspect interferes with the base application is not presented in the code. We address this limitation with the strong interaction with the developer's environment. The Eclipse plugin provides a mean to display remediation code in place at a given time.

VI. Related work

The interest into static analysis field has led to several approaches. They go from simple techniques like pattern matching and string analysis like in [8]–[11] to more complex techniques like data flow analysis in [12]–[14]. Commercial tools, such as Fortify [15] or CodeProfiler [16] propose better integration in developers' environment but lack of decentralized approach and assistance in security management. Several tools are based on the Eclipse's platform and detect vulnerabilities in web applications [17] , flaws [18], bugs [19], and propose testing and audit to verify respect of organizational guidelines [20]. Compared to the aforementioned techniques, we advocate a better integration into the daily development lifecycle with our tool, and propose an integrated correction with good accuracy as we leverage developer's knowledge on development context.

Hermosillo et al. [21] uses AOP to protect against web vulnerabilities: XSS and SQL Injection. They use AspectJ - the mainstream AOP language, to intercept method calls in an application server then perform validation on parameters. Viega et al. [22] presents simple use case on the usage of AOP for software security. Masuhara et al. [23] introduces an aspect primitive for dataflow, allowing to detect vulnerabilities like XSS. Our approach reduces the overhead brought by the detection of vulnerability patterns at runtime and allows wider range of vulnerability detection. Also, the aforementioned approaches do not rely on external tools to gather security context, but rather a manual processing to understand the architecture and decide where to apply aspects. Our approach also brings more awareness to the developer as he obtains a visual indicator of what is applied at which place in its application.

A combination of detection and protection is found in Deeprasertkul et al. [24] approach for detecting faults identified by pre-compiled patterns. Faults are corrected using a correction module. The difference with our approach lies in the detection of faults rather than security vulnerabilities. Also, the correction module fixes the faults statically and prevents further modifications of the introduced code. A recent work conducted by Yang et al. [25] uses static analysis to detect security points to deploy protection code with aspects, on distributed tuple space systems. These two approaches suffer from same limitation as the ones presented in the previous paragraph, which is a lack of visual support from the tool, and a loose of context. It is worth mentioning the work from Hafiz et al. [26], where authors propose several techniques to correct data injection through program transformations. They have list several cases along with steps to describe transformations to realize security policies. Their work can benefit our overall methodology to propose multiple corrections once vulnerability has been identified.

VII. Conclusion and future works

We presented how to overcome several security vulnerabilities using a combination between a static analyzer that assists developers to report security vulnerabilities and a semi-automated correction of these findings with AOP. The usage of an integrated tool to provide support for security bugs detection and mitigation has several advantages. It benefits to several stakeholders at the same time. First, security teams are able to distribute the maintenance of the code to the people writing their code and let them mitigate security bugs whenever they are detected. They can interact closely to decide of the best solutions for a given situation, and apply security across development teams. Developers benefit from this approach, having an operational tool already configured for their development. They can focus on writing their functional code and, time to time, verify the accuracy of their implementation. Security concerns are often cross cutting the application, which tends to have security checks spread around application. Using one central tool to have an overview is more efficient and productive, and gives the possibility to track all applied protection code. The automation allows a broader and consistent application of security across applications. The usage of AOP eases the deployment and change of security protection code, in a single environment and during the development phase. The overall vision we would like to achieve in the future is the specification and maintenance of security concerns in one central place, and usage by developers of these concerns by defining some places in application where they should be active.

We have designed this plug-in for an improved awareness of security concerns from a developer point of view. It is important to notice that correcting vulnerabilities doesn't make the whole system secure. It only means the code tends to be free of security bugs. Other parts of the application, such as authentication flow, authorization checks, etc. are not covered by our analysis. Besides, we encourage developers to look further in vulnerabilities' descriptions, as the automated correction proposed might not be the best choice in all situations. We do not want developers to believe our solution is bullet-proof. It leads to a false sensation of security, which is the opposite of our goal.

Albeit we have listed several benefits for an integrated tool, we know that it suffers from limitations. For instance, when we are developing a tool such as an Eclipse plug-in, we are targeting a platform and a language, thus voluntarily restricting the scope of application. From the tool itself, we have designed a working prototype that we have validated on projects internally at SAP and compared to commercial softwares. In several cases, the agile approach leads to a reduction of false positives and an absence of false negatives. Also, the approach of providing support for correcting the vulnerability is novel and we focus now in improving accuracy of the protection code. Especially, we need to investigate in the cost in term of complexity and maintainability for the different stakeholders interacting with the system.

Acknowledgment

This work has been partially carried out in the CESSA project (project id.: 09-SEGI-002-01).

References

[1] T. Scholte, D. Balzarotti, and E. Kirda, “Quo vadis? a study of the evolution of input validation vulnerabilities in web applications,” in *Proceedings of Financial Cryptography and Data Security 2011*, ser. Lecture Notes in Computer Science, February 2011.

[2] M. Guarnieri, P. El Khoury, and G. Serme, “Security vulnerabilities detection and protection using eclipse,” in *ECLIPSE-IT 2011*, Milano, ITALY, September 2011.

[3] J. Gosling, B. Joy, G. Steele, and G. Bracha, “Java(TM) Language Specification,” http://docs.oracle.com/javase/specs/, January 2005.

[4] T. Mens and T. Tourwe, “A survey of software refactoring,” *Software Engineering, IEEE Transactions on*, vol. 30, no. 2, pp. 126 – 139, February 2004.

[5] G. Kiczales, J. Lamping, and al., “Aspect-oriented programming,” in *ECOOP*, ser. Lecture Notes in Computer Science, M. Aksit and S. Matsuoka, Eds. Springer Berlin / Heidelberg, 1997, vol. 1241, pp. 220–242.

[6] OWASP, “OWASP Top Ten Project,” http://www.owasp.org/index.php/OWASP_Top_Ten_Project, 2010.

[7] MITRE, “CWE/SANS Top 25 Most Dangerous Software Errors,” http://cwe.mitre.org/top25, September 2011.

[8] J. Viega, J. T. Bloch, Y. Kohno, and G. McGraw, “Its4: A static vulnerability scanner for c and c++ code,” in *ACSAC*. IEEE Computer Society, 2000, pp. 257–.

[9] C. Gould, Z. Su, and P. T. Devanbu, “Jdbc checker: A static analysis tool for sql/jdbc applications,” in *ICSE*. IEEE Computer Society, 2004, pp. 697–698.

[10] G. Wassermann and Z. Su, “An analysis framework for security in web applications,” in *Proc. FSE Workshop on Specification and Verification of Component-Based Systems*, ser. SAVCBS’04, 2004, pp. 70–78.

[11] A. S. Christensen, A. Moller, and M. I. Schwartzbach, “Precise analysis of string expressions,” in *Proc. 10th International Static Analysis Symposium*, ser. SAS’03. Springer-Verlag, 2003, pp. 1–18.

[12] M. S. Lam, J. Whaley, V. B. Livshits, and al., “Context-sensitive program analysis as database queries,” in *Symposium on Principles of database systems*, ser. PODS’05. ACM, 2005, pp. 1–12.

[13] Y. Liu and A. Milanova, “Static information flow analysis with handling of implicit flows and a study on effects of implicit flows vs explicit flows,” in *Proceedings of the 2010 14th European Conference on Software Maintenance and Reengineering*, ser. CSMR ’10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 146–155. [Online]. Available: http://dx.doi.org/10.1109/CSMR.2010.26

[14] D. Balzarotti, M. Cova, V. Felmetsger, N. Jovanovic, E. Kirda, C. Kruegel, and G. Vigna, “Saner: Composing static and dynamic analysis to validate sanitization in web applications,” in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2008, pp. 387–401.

[15] HP, “Fortify 360,” https://www.fortify.com/, June 2012.

[16] Virtual Forge, “Codeprofilers,” http://www.codeprofilers.com/, June 2012.

[17] V. B. Livshits and M. S. Lam, “Finding security errors in Java programs with static analysis,” in *Proceedings of the 14th Usenix Security Symposium*, Aug. 2005, pp. 271–286.

[18] J. Dehlinger, Q. Feng, and L. Hu, “Ssvchecker: unifying static security vulnerability detection tools in an eclipse plug-in,” in *Proc. OOPSLA Workshop on eclipse technology eXchange*, ser. Eclipse’06. ACM, 2006, pp. 30–34.

[19] University of Maryland, “Findbugs,” http://findbugs.sourceforge.net, July 2012.

[20] Google, “Codepro analytix,” http://code.google.com/javadevtools/codepro/, June 2012.

[21] G. Hermosillo, R. Gomez, L. Seinturier, and L. Duchien, “Aprosec: an aspect for programming secure web applications,” in *ARES*. IEEE Computer Society, 2007, pp. 1026–1033.

[22] J. Viega, J. T. Bloch, and P. Ch, “Applying aspect-oriented programming to security,” *Cutter IT Journal*, vol. 14, pp. 31–39, 2001.

[23] H. Masuhara and K. Kawauchi, “Dataflow pointcut in aspect-oriented programming,” in *APLAS*, ser. Lecture Notes in Computer Science, A. Ohori, Ed., vol. 2895. Springer, 2003, pp. 105–121.

[24] P. Deeprasertkul, P. Bhattarakosol, and F. O’Brien, “Automatic detection and correction of programming faults for software applications,” *Journal of Systems and Software*, vol. 78, no. 2, pp. 101–110, 2005.

[25] F. Yang, T. Aotani, H. Masuhara, F. Nielson, and H. R. Nielson, “Combining static analysis and runtime checking in security aspects for distributed tuple spaces,” in *COORDINATION*, ser. Lecture Notes in Computer Science, W. D. Meuter and G.-C. Roman, Eds., vol. 6721. Springer, 2011, pp. 202–218.

[26] M. Hafiz, P. Adamczyk, and R. Johnson, “Systematically eradicating data injection attacks using security-oriented program transformations,” in *Proceedings of the 1st International Symposium on Engineering Secure Software and Systems*, ser. ESSoS ’09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 75–90.

Risk Identification for an Information Security Management System Implementation

Nor Aza Ramli, Normaziah Abdul Aziz
Kulliyyah of Information and Communication Technology
International Islamic University Malaysia
Gombak, Malaysia
azaramli@gmail.com, naa@iium.edu.my

***Abstract* — ISO/IEC 27001 is an international standard that provides a set of requirements for an Information Security Management System (ISMS) implementation. A risk assessment exercise for an ISMS implementation requires human expertise with comprehensive understanding and considerable knowledge in information security. A common risk assessment exercise is based on three sub-processes, namely, risk identification, risk analysis and risk evaluation. The lack of tools especially in the automation of risk identification emphasized the need of experienced personnel and this becomes a challenge for organizations seeking compliance with the ISMS standard. This paper proposes a relationship concept in asset and threat identification which is part of the risk identification sub-process. The concept provides a foundation to automate the risk assessment process for an identified scope of an ISMS implementation.**

Keywords – ISMS; information security risk; asset identification; threat; risk assessment

I. INTRODUCTION

A. *Information security*

Information is an asset that has value to an organization. It is, like other important business assets, essential to an organization’s business and consequently needs to be suitably protected, which is especially important in the increasingly interconnected business environment [1]. The International Organization for Standardization and International Electrotechnical Commission (ISO/IEC) published various standards for ISMS. ISO/IEC 27002 defined information security as the preservation of confidentiality, integrity and availability of information; in addition, other properties, such as authenticity, accountability, non-repudiation, and reliability can also be involved [1]. In order to achieve information security, an organization needs to first identify what are the assets that need the protection and perform a risk assessment exercise to determine the level of risks and the suitable set of controls to minimize these risks, eventually securing the assets.

B. *Information security risk*

Organizations that are dependent on information technologies consequently have to face a common issue of managing information security risks which are inherited with the use of the technologies. In 2009, SANS Institute has issued a report entitled “The Top Cyber Security Risks” that discussed on the importance of understanding security threats and their corresponding vulnerabilities prior to identifying security controls to mitigate the associated risks [2]. Global State of Information Security Survey, 2010 has in its findings, organizations have considered taking a risk-based approach as well as adopting a recognized security framework in addressing information security issues [3]. According to Humphreys, if an organization does not know the risks it faces, it will not be able to implement proper and effective protection [4]. In 1995, Kailay and Jarrat have highlighted that one of the gaps then was the limited risk analysis methodologies and corresponding tools for certain domain users [5]. At present, that gap has been addressed through publication of documented guidelines on several risk assessment methodologies such as the ISO/IEC 27005:2011 [9]. However, methodologies alone could not guarantee an effective information security risk assessment. Risk assessment process comprises of three sub-processes, namely, risk identification, risk analysis, and risk evaluation. Automation of the process is common in many risk assessment tools with the exception of risk identification. Hence, automation of risk identification would be useful for organizations especially for those carrying out risk assessment for the first time.

C. *Information Security Management System (ISMS)*

Acknowledging the importance of understanding and managing information security risk, a global effort by information security practitioners has resulted in the development of a standard for an Information Security Management System (ISMS). ISMS standards started in the early 90s with the first draft of an information security management standard published as British Standard (BS), BS 7799. It focused on security related to people, processes, information as well as information technology [4]. In 2005, BS 7799 Part 2 became an international standard known as ISO/IEC 27001:2005 [6]. ISO/IEC 27001 standard is a specification for information security management system developed jointly by the ISO/IEC, and was published in 2005 [7]. This standard adopts a risk-based approach for an effective information security management taking into consideration the information security aspects of various areas within an organization [6]. In an ISMS implementation, organization will have to identify a scope for the ISMS and this scope will be subject to a risk assessment to identify appropriate controls to mitigate the identified risks.

Current tools including documented guidelines in risk management such as the ISO/IEC 27005:2011 could be used by organizations to facilitate the risk assessment process [9]. However, these tools are lacking in automation and its usage requires human expertise with professional judgment and knowledge of information technologies as well as capability to relate information security threats with organizational risk management [5], [8], [10], [12].

The lack of tools especially in the automation of risk identification emphasized the need of experienced personnel and this becomes a challenge for organizations in implementing information security management especially those seeking compliance with the ISMS standard.

This paper discusses some relationship concepts in asset and threats identification. Identifying accurate assets and relevant threats are very important to ensure reliable risk assessment results. This is part of our current work to automate the risk assessment process.

The contribution in this paper is the identification of assets and their relationships to relative threats for an ISMS scope to facilitate automation of the risk assessment process. The relationships developed in this study are limited to the identified ISMS scope which is secure data centre. It aims to address automation in risk assessment for network security threats which can be expanded to other category of threats.

II. Related Work

Previous works on similar efforts to automate risk assessment process are reviewed in this section. In 1995, a prototype expert system for computer security risk analysis and management was developed at the School of Computer Science, University of Birmingham. RaMEX was developed based on RAM (Risk Analysis and Management) methodology and concentrated on the category of intentional threats [5]. As the name suggests, RaMEX facilitates risk assessment step-by-step following a methodology developed specific for it.

Another work sighted has emphasized on the importance of using previously acquired knowledge in risk analysis. A risk analysis system in electronic commerce environment was developed at the Korea Advanced Institute of Science and Technology, Seoul [10]. The system was based on case-based reasoning (CBR), taking advantage of the experience and learning from incidents knowledge into the analysis of risks. According to Liao and Song [11], even though the CBR approach could make use of past solutions, it takes time to collect such cases and in the event that a case is the first one to occur, the results of the assessment could be limited.

Liao and Song have taken a different approach in developing a computer-aided system to facilitate risk assessment process [11]. Their work has focused on transaction-based risk assessment by looking at the value of a transaction to an organization to determine the impact of losses to the business. Instead of depending on past solutions, risk assessment is performed based on transactions that have been defined and known to the system [11].

Similarly, a work by Aime, et al. [8] approached risk analysis based on models that can be built at runtime and during system monitoring of a known target system. The model was used to automate some portion of the risk analysis processes namely the collection of threats data, the identification of applicable threats to the target system and the calculation of risk level.

In 2007, Software Engineering Institute at Carnegie Mellon University has published a technical report on OCTAVE Allegro which focused on information assets in its methodology that is used for identifying and evaluating information security risks. It approached risk assessment by focusing on information asset and its containers such as people, physical and technology aiming to produce a more robust assessment result [12]. In 2009, Chivers et al. has assessed risks to a particular system incrementally using formally defined risk profiles [13].

As a summary, scholars have carried out studies applying different approaches on various scope of assessment to achieve improvements in risk assessment process including targeting its automation.

III. Our Proposal

The core idea of this research is to automate all the three sub-processes in a risk assessment process for an identified scope of ISMS. We are proposing to focus on the risk identification as this is a sub-process where domain knowledge in information security needs to be applied. Domain knowledge on what are the significant assets for an identified ISMS scope, and what are the threats and corresponding vulnerabilities on those assets, will be modelled using an ontology editor to develop relationships of these important risk assessment parameters. With our proposed work, the tools are expected to be easily comprehended by a non-experienced risk analyst as both angles of the risk assessment i.e. the methodology and information security domain knowledge would have been carefully modelled with the use of ontology rules. Involvement of an experience risk analyst could be minimized and their resources could then be utilized effectively, only when needed.

There are various tools for ISMS implementation that addressed the whole process of the management system based on the “Plan-Do-Check-Act” (PDCA) model. The tools have been designed to ensure implementation complies with the standard, i.e., ISO/IEC 27001:2005. As ISMS adopts a risk-based approach, risk assessment is one of the main components of these tools. As far as automation is concerned, current tools have been observed to facilitate the end-to-end risk assessment methodology as well as performed calculation based on selected formulas during risk analysis and risk evaluation sub-processes exercises. Automation of the risk identification sub-process, however, has not been included as part of the tools’ feature. Considerable involvement of a competent risk analyst with information security domain knowledge was still required. For example, to identify assets of an ISMS scope given the possible list of asset types which is taken from guidelines such as the ISO/IEC 27005 is rather confusing. Is an identified asset subject to risk assessment or the asset is in fact a control that has been implemented to mitigate a risk? As an example, is a firewall an asset that needs to be protected or is it a control that has been implemented to protect an asset? Forming the basis of what assets are indeed subject to risk assessment for a specified

ISMS scope have yet seen to be explicitly addressed by existing tools. In a common risk assessment exercise, expert resources have been observed to be utilized ineffectively due to this lack of automation. The output of this study is to build a prototype that enables risk assessment automation for organizations going for ISO/IEC 27001 certification. The proposed relationship concepts in asset identification contributed to the automation of the risk assessment for an identified scope of implementation. Protégé OWL (Web Ontology Language) was used to create classes and corresponding rules to demonstrate the relationships. Protégé is an ontology editor which is based on an open-source platform. It was developed by Stanford Center for Biomedical Informatics Research at the Stanford University School of Medicine. There are two types of the system; Protégé Frames and Protégé Owl. The former supports frame-based ontology while the latter supports Web Ontology Language. Both are actively being used as well as updated from time to time as observed from its website [14].

A. *ISMS Scope*

According to the International Register of ISMS Certificates, there were 7,686 certificates registered by organizations from eighty-five countries [15]. Malaysia held fifty-eight certificates and was ranked at 14th place as of March 2012. About eighty percent of the certificates in Malaysia have identified scope that is related to secure data centre service. Implementation of ISMS to manage a secure data centre is thus indicated to be very relevant in the context of organizations that highly depending on IT as their business enabler.

B. *Asset identification*

Asset identification is the first step in risk identification. Following the ISMS requirements, assets within the ISMS scope shall be identified prior to carrying out a risk assessment process [7]. There are two categories of assets as described in the ISO/IEC 27005; primary asset and supporting asset [9]. Primary assets are core business processes and their corresponding information whilst supporting assets are those required to be in place to support the activities of the primary assets. OCTAVE Allegro on the other hand has a different approach in asset identification. Its asset profiling is focused on information assets and their corresponding containers in which these assets lived [12]. The concepts used by the ISO/IEC 27005 and OCTAVE Allegro are similar i.e. information was identified as the key asset and other relevant assets were identified in relation to the information. For this work, both approaches were adopted and streamlined to assist in the asset identification.

TABLE I: TYPES OF ASSETS

Asset	Descriptions	Remarks
Information	Digital format/ printed on hardcopies i) Application data ii) System configuration files iii)System log files	Here, data is divided into three types to ensure consistent approach & understanding of the type of information that require protection. **Adoption & extension of:** a) Octave Allegro i) information asset ii) physical container b) ISO/IEC 27005 – i) primary asset: information ii) supporting asset: hardware
Data centre system	Applications and systems storing the 'Information' asset.	The system includes software and hardware. **Adoption & extension of:** a) Octave Allegro: i) technical container b) ISO/IEC 27005: i) supporting asset: software, hardware & network
Data centre infra	The physical location; data centre including general telecommunication equipment, utilities such as power, air –conditioning & humidity control.	**Adoption & extension of:** ISO/IEC 27005 i) supporting asset: site
People	The people involved with the information asset: Staff - internal Client & Contractor - external	**Adoption & extension of:** a) Octave Allegro – people container b) ISO/IEC 27005 – supporting asset: personnel

Taking a common and significant scope of ISMS implementation from Section III.A, an example of secure data centre is used to demonstrate the relationship concept derived for asset identification.

Table I listed the generic type of assets in a data centre, mapping them to how assets are being described in ISO/IEC 27005 and Octave Allegro.

At this stage, it is very important to fully understand and able to identify the assets involved and their corresponding types. Inaccurate asset identification with vague description of each asset type may lead to unnecessary efforts in the subsequent steps of the risk assessment. In Table 1, each of the asset type and its description is detailed and these descriptions are supported by the corresponding guidelines provided by ISO/IEC 27005 and Octave Allegro. The four types of assets above are proposed to be the set of assets for ISMS scope of a secure data center. The types of assets are explicitly set for the automation of the risk assessment exercise.

The relationship concept used for the asset identification phase above is designed to further eliminate complexity when it comes to threats and vulnerabilities identification. An experience risk analyst would be able to easily point out a potential repetition of risk exercises due to unstructured identification of assets. For the purpose of this paper, 'People' as an asset will not be included in rest of the discussion as it has a unique relationship which is addressed separately for the automation of the risk assessment. The relationship concepts are further discussed in the following section.

IV. ANALYSIS

In many ISMS implementations, the identification of assets was driven by organizational asset management process. This could pose as a challenge especially when the asset definitions and categorizations did not consider the information infrastructure which the organization had in place. This section discusses the analysis of the proposed relationships for the set of assets identified in Table I above. The discussion is limited to the scope of ISMS as discussed in Section III.A, i.e., secure data centre. The following relationships are demonstrated by ontology graphs which were generated using ontology editor, Protégé OWL.

A. *Relationship 1*

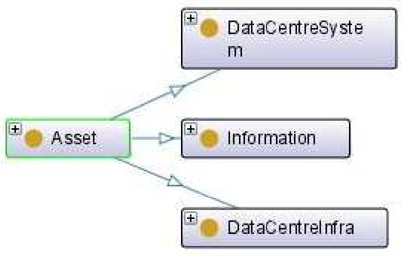


Figure 1. Assets within an ISMS scope

A secure data centre commonly housed key information asset. This asset is in digital format and requires corresponding hardware and software for it to be usable to an organization. These hardware and software components are defined as ‘Data Centre System’.

The ‘Data Centre System’ requires a suitable environment for it to operate at its maximum capacity with minimal disruption. This environment is defined as ‘Data Centre Infra’.

Thus the identified assets for ISMS scope of a secure data centre are Information, Data Centre System and Data Center Infra. Therefore, these three assets are subject to a risk assessment.

B. *Relationship 2*

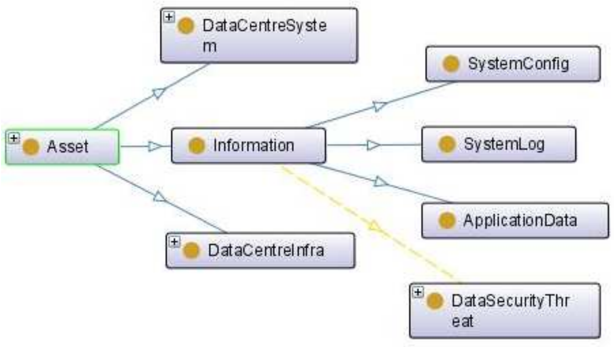


Figure 2. Information asset

The information asset of a secure data centre is further broken down into ‘Application Data’, ‘System Configuration’ and ‘System Log’. These are important components of Information asset which are susceptible to threats related to information asset. The threats are defined as ‘Data Security Threat’.

It is noted that up to this point, the relationships described are very common. It is however very significant to be discussed in this section as the rest of the relationships are based on these foundations.

C. *Relationship 3*

This relationship is for identifying threats for Data Centre System which is an asset of a secure data centre.

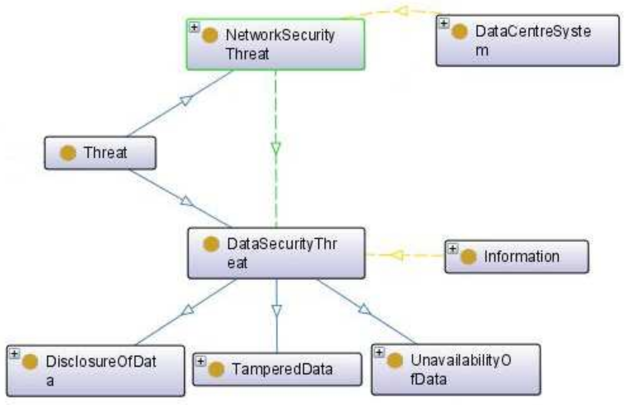


Figure 3. Threats

1) *Descriptions*

a) Two types of threats are shown in Figure 3; Network Security Threat and Data Security Threat

b) Data Security Threats are DisclosureOfData, TamperedData and UnavailabilityOfData

c) Network security threats are reconnaissance attacks, session attacks, unauthorized network access, DoS/DDoS (denial-of-service/ distributed denial-of-service) and malware attacks.

d) Network Security Threats are threats to Data Centre System.

e) Data Security Threats are threats to Information

f) Network Security Threats on Data Centre System resulted into Data Security Threats on Information

2) *Analysis*

a) “Network Security Threats” will eventually lead to threats on Information. This is justified due to the fact that Information resides in the Data Centre System and hence inherited the threats to the Data Centre System.

b) “Data Security Threats” are therefore inferred to be the results of “Network Security Threats”.

c) This is an example of a relationship between data security threats and network security threats. Other relationships involving different types of threats might have the same results and will be used later in this study.

d) In a common risk assessment exercise, asset owners will need to be involved. Threat identification phase for an application owned by a business unit, may have the following scenario:

TABLE 2: THREAT IDENTIFICATION – A SCENARIO

Asset ID	**Asset Description**	**Asset Owner**	**Asset Type**
Asset 1	Business Application-System	Business Unit	Data centre system
Asset 2	Business Application-Application Data	Business Unit	Information
Asset 3	Business Application-System Log	Security Unit	Information

Guided by Relationship 1, Business Unit has identified both system (Asset 1) and data (Asset 2) as their assets. The latter could be unintentionally left out during an assessment as it could be assumed to be bundled in Asset 1 without specifying it explicitly and may result in an incomplete assessment. Next, with Relatioship 2, system log (Asset 3) has been identified by Security Unit which was not the main owner of the application.

Applying Relationship 3, unauthorized network access from a network security threat may be exploited by some vulnerabilities and could cause tampered data for Asset 2 and Asset 3. However, in a common risk assessment exercise, this threat may have only been identified for Asset 1 and the cascading impact on information asset residing in it might not be properly highlighted and analysed. Instead, a different set of assessment could have been carried out on information assets resulting in risk assessment results which were repetitive and lack of clarity.

e) Hence, Relationship 3 indicates that risk assessment could be conducted in a more structured manner whereby repetition of identifying threats for both Information and Data Centre System would be avoided.

V. CONCLUSION AND FUTURE WORK

Three relationship concepts have been discussed. These concepts were used to develop other relationships which have enabled the automation of risk assessment for an identified ISMS scope. An advisory system prototype was developed based on a risk assessment approach taken from the ISO/IEC 27005 to demonstrate the relationship concepts. Four types of assets were identified during the asset identification phase. However the threats identification phase had focused on two types of the assets namely Data Centre System and Information.

Future work will extend the relationships into selection of control measures to mitigate the identified risks.

VI. REFERENCES

[1] ISO/IEC, "ISO/IEC 27002 Code of practice for information security management," ISO/IEC, 2005.

[2] “The Top Cyber Security Risks”, (SANS Institute), [online] September 2009, http://www.sans.org/top-cyber-security-risks (Accessed: 29 March 2012).

[3] “Trial by fire”, (PWC), [online] 2009, http://www.pwc.com/giss2010 (Accessed: 25 April 2010).

[4] E. Humphreys, "Information security management standards: Compliance, governance and risk management," Information Security Technical Report, vol. 13, 2008, pp. 247-255.

[5] M. P. Kailay and P. Jarratt, "RAMeX: a prototype expert system for computer security risk analysis and management," Computers & Security, vol. 14, 1995, pp. 449-463.

[6] E. Humphreys, Implementing the ISO/IEC 27001 Information Security Management System Standard. Boston,London: ARTECH House, 2007, pp 21-25.

[7] ISO/IEC, "ISO/IEC 27001 Information Security Management Systems - Requirements," ISO/IEC, 2005.

[8] M. D. Aime, A. Atzeni, and P. C. Pomi, "AMBRA - Automated Model-Based Risk Analysis," in CCS: Conference on Computer and Communications Security (Proceedings of the 2007 ACM workshop on Quality of protection table of contents, SESSION: Risk analysis), Alexandria, Virginia, USA, 2007, pp. 43-48.

[9] ISO/IEC, "ISO/IEC 27005 Information security risk management," ISO/IEC, 2011.

[10] C. Jung, I. Han, and B. Suh, "Risk Analysis for Electronic Commerce Using Case-Based Reasoning," International Journal of Intelligent Systems in Accounting, Finance & Management, vol. 8, 1999, pp. 61-73.

[11] G.-Y. Liao and C.-H. Song, "Design of a Computer-Aided System for Risk Assessment on Information Systems," in IEEE 37th Annual International Carnahan Conference on Security Technology, 2003, pp. 157-162.

[12] R. A. Caralli, J. F. Stevens, L. R. Young, and W. R. Wilson, "Introducing OCTAVE Allegro: Improving the Information Security Risk Assessment Process," May 2007.

[13] H. Chivers, J. A. Clark, and P.-C. Cheng, "Risk profiles and distributed risk assessment," Computers & Security, vol. 28, 2009, pp. 521-535.

[14] “What is protégé?”, (protégé), [online] 2012, http://protege.stanford.edu (Accessed: 29 March 2012)

[15] “International Register of ISMS Certificates”, [online] 2012, http://iso27001certificates.com/ (Accessed: 29 March 2012)

A QTE-based Solution to Keylogger Attacks

Chien-Wei Hung*, Fu-hau Hsu*, Shih-Jen Chen†, Chang-Kuo Tso*, Yan-Ling Hwang‡, Po-Ching Lin§ and Li-Pin Hsu¶
*Dept. of Comput. Sci. & Inform. Eng., National Central University, Zhongli City, Taiwan (R.O.C.)
Email: winestwinest@gmail.com; hsufh@csie.ncu.edu.tw; cktso@csie.ncu.edu.tw
†Institute for Information Industry, Taipei City, Taiwan (R.O.C.)
Email: jchen@iii.org.tw
‡Dept. of Applied Foreign Languages, Chung Shan Medical University, Taichung City, Taiwan (R.O.C.)
Email: yanling@csmu.edu.tw
§Dept. of Comput. and Inform. Sci., National Chung Cheng University, Chiayi County, Taiwan (R.O.C.)
Email: pclin@cs.ccu.edu.tw
¶Dept. of Applied Inform. Sci., Chung Shan Medical University, Taichung City, Taiwan (R.O.C.)
Email: apple@csmu.edu.tw

***Abstract*—Nowadays keystroke logging is one of the most widespread threats used for password theft. In this paper, rather than detecting existing malware or creating a trusted tunnel in the kernel, we present a different method QTE (Quick Time Events) to protect the password that a user provides for a web page to login to a web service. Installing such solutions in a host only requires limited privileges of related computers. The QTE method utilizes a canvas to cue users when their input will be recorded or ignored by the QTE add-on, which provides a chance for users to obfuscate keyloggers by inserting meaningless letters among the keystrokes of their passwords. QTE can be applied to all websites without any modification of them.**

***Keywords*-Authentication, computer security, keylogger, privacy.**

I. Introduction

Keystroke logging is a behavior of recording keystrokes, and a program with this property is called a keylogger. Typically, keyloggers sniff users' input, such as bank accounts or passwords, behind the scenes and upload the logs to attackers. Since keyloggers are easy to implement and produce high profit, about half of malware includes a keylogger in their code according to a Symantec's report [1]. In 2005, more than eight hackers were accused of planning to steal over 423 million dollars from a Japanese bank by installing keyloggers in her systems [2], which highlights the potential danger of keystroke logging.

In spite of many related works proposed various promising methods to defend keyloggers, they are less feasible in practice owing to the prerequisite of having root privileges or a dependable device. Besides, even though some companies like Google and Facebook provide the one-time password mechanism, it is only available for their services rather than all websites. Furthermore, in order to use their mechanism, one must reveal her/his personal data, such as a phone number, to each service provider and therefore sacrifices her/his privacy right indeed. To make matters worse, in last year, some reports [1][3] found that about 12% of leaked data were revealed by insiders and 96% of leaked information are personal data.

With the popularization of computers and Internet, it is very common for users to log in online banks or e-mail services using public computers. However, it is challenging to design a secure system in a public computer since common users only have limited privileges, and various persons can use the computer without being monitored. Therefore, an appropriate solution for keyloggers in this scenario must be able to be able to be actively executed by users without special privileges and applied to all websites. To meet the prerequisites, we propose the QTE method for users to defend their systems against keyloggers.

The QTE method provides a secure environment for users to input passwords in web browsers. The QTE method records a keystroke as a letter of a password only when the system is at a valid time interval. Valid time intervals are interleaved with invalid time intervals. Letters input at different time intervals are combined according to their time interval order. QTE utilizes animation to visualize valid time intervals, such as a special object moves over a specific screen area. QTE allows a user to obfuscate keyloggers by tapping keyboards haphazardly when the system is not at a valid time interval.

In this paper, we focus on practicable solutions when using public computers, and our work makes the following contributions:

1) This paper provides a feasible solution to obfuscate or bypass most kernel, hypervisor, hardware, and second-channel keyloggers even with a privilege-limited account in the user-space.
2) Our solution can be applied to all websites rather than a specific web service.
3) We do not rely on a password set beforehand or disclose users' information when using our approach.
4) Our solution will not slow down the speed of surfing the Internet.

5) Compared with most related work, our solution is easier to use but provide more powerful defense to keyloggers.

The rest of this paper is organized as follows. Section II discusses previous work about keyloggers with the evolution of attack and defense techniques. Section III depicts the principle and the system design of our solutions. Section IV gives the implementation details of our methods. Section V evaluates our solutions with seven different user-space and kernel-space keyloggers, and estimates the time an attacker has to consume in order to get a password under the protection of our mechanism. Experimental results show that our approach is immune to all tested keyloggers. Section VI concludes this paper and discusses the feasibility of extending our work to more applications.

II. Related Work

Since the first keylogger written by Perry S. Kivolowitz [4] appeared in 1983, keyloggers have become an essential element for most malware. While user-space keyloggers like Home Keylogger and Family Keylogger can be easily detected [5] and obfuscated by sending fake keystroke messages [6][7], most malware nowadays uses rootkit to hide itself in kernel or hypervisor. As a result, a lot of previous work was proposed and can be categorized into five types.

A. Signature-Based Solutions

A common way to detect kernel keyloggers is to use anti-keylogger software, such as SpyReveal. Anti-keylogger software detects keyloggers based on signatures [8]. For hardware keyloggers, Mihailowitsch suggested differentiating it according to the clock cycles of the keyboard [9]. However, both of them require known samples or devices for profiling; therefore, are resistless to new variants.

B. Encryption and Decryption

Another viewpoint adopted by both attackers and administrators utilizes cryptography to encrypt keystrokes in the device drivers and decrypts them in the applications [10]–[12]. To enhance the security, McCune used a mobile phone to communicate with the kernel module installed on the host with RSA encryption [13][14], and later the kernel module transmitted encrypted keystrokes to the registered application for decryption. Besides, similar principle is taken by Hirano who used a hypervisor-based isolation of trusted domain to exchange data over TLS/SSL [15].

These solutions provide stronger defense, but need root privileges of the related computers, which is inapplicable for public computer users, not to mention performance degradation caused by encryption and decryption. Moreover, with the evolution of hacking techniques, some keyloggers can even "upgrade" firmware for the purpose of concealment. A notable event is the exploitation of Mac OS X discovered by K. Chen [16]. More importantly, second-channel keyloggers are the focus of public attention nowadays. This kind of keyloggers can eavesdrop keystrokes via acoustics [17]–[19], electromagnetism [20], registers [21], power leakage, or even from optical sampling of mechanical energy [22]. In fact, solutions of this type are resistless to these keyloggers.

C. Graphical Password and On-Screen Keyboard

As keyloggers become more and more sophisticated, researchers continue developing new techniques to protect a computer against keyloggers. Graphical passwords [23]–[26] and on-screen keyboards [27][28], which make users type their passwords by mice, are new mechanisms extensively used by online banks to hedge against accounts theft.

Notwithstanding the new safeguards mentioned above, keyloggers also developed various approaches, such as taking screen snapshots and tracking cursor events, to bypass the protection or to facilitate information collection. The capability of screen capturing may be the most serious threat since everything displayed to users is also displayed to the attacker; the attacker can therefore speculate what users really type or click. In brief, an attacker can get passwords simply by a screen-shot and mouse event logs.

D. One-Time Password

One defensive way chosen by Facebook is one-time passwords [29]. Because each one-time password is only valid for one login session and is sent to a user through SMS by Fackbook, keystrokes logs become useless. Another variant of one-time passwords is the two-step verification [30] of Google that requires a random certificate as the second verification when logging in to a Google account.

One-time passwords are one of the strongest measures against keyloggers. However, users may need to render their real information, such as their cell phone numbers, to service providers, which may infringe users' privacy surreptitiously. Furthermore, a critical defect is that a one-time password could only be recognized by its supporting website instead of all websites. In other words, if the website that a user tries to log in does not provide the one-time password mechanism, the user cannot be protected by the mechanism. Moreover, for each account on each website that provides one-time passwords, in order to enable one-time passwords, a user needs to make related setup. When a user has multiple accounts with one-time passwords, the management of the setup becomes a non-trivial work. However, our approach not only can be applied to all websites without their support, but also keep users' privacy since they do not have to provide their private information to our mechanism.

E. Proxy Server

For the purpose of generality, Wu, Gieseke, and Pashalidis [31]–[34] recommend a proxy server as the intermedium between public computers and login servers. They keep a copy of passwords on the proxy, and authenticate users with shared secrets defined beforehand such as personal questions or answer lists. These approaches provide a convenient and securer way to login to an account, but a compromised server or an unethical webmaster may seriously damage all related users. For this reason, Dinei [35] advices an adapted version that keeps nothing but password decryption algorithms used by users. However, this solution suffers non-trivial performance overhead.

III. System Design

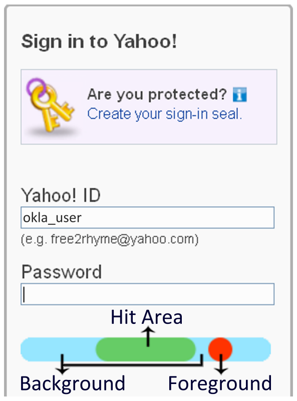
Occasionally, a user has to type her/his password on a public computer. However, a public computer is usually a hostile environment since it may contain any software installed by any users, including skilled attackers. Besides, a normal user usually has only a limited privilege on a public computer. Thus, to develop a secure anti-keylogger solution on a public computer, the above limitations should be taken into account; otherwise, the proposed solution may not be a feasible one. To satisfy the above requirements, our solution assumes users have only limited privileges on public computers. Furthermore, our solution does not ask users to provide their private information or account information beforehand. Our solution is implemented as a browser extension that everyone can install using her/his account.

In our design, we assume there is no Man-In-the-Middle attack between a public computer with our extension and a website to which a user of the public computer tries to log in. But, if the website supports encryption, Man-In-the-Middle attacks can be solved by encrypting all messages. This section depicts the structures, components, advantages and disadvantages of the QTE method first. Implementation details will be expatiated in the next section.

A. QTE Method

QTE, which stands for Quick Time Event, is a video game term coined by Yu Suzuki [36]. Literally, QTE requires players to immediately perform some actions on control devices for on-screen prompts; otherwise, they will get some penalties or different outcomes. In other words, people may get nothing from the event if they do not react to the prompt in time. This idea is therefore be adopted by us to obfuscate most sorts of keyloggers.

1) Analysis: Most keyloggers have a full privilege of a compromised public computer while a user has a limited privilege in a public computer. Hence, forging any keystroke or hiding something by software may not be able to fool keyloggers. Specifically, the only thing that can interfere with most kernel, hypervisor, hardware, and second-channel keyloggers is what a user really types. As a result, we convert our problem to *"When could users type keys safely?"* or *"How could users know whether they could hit a keyboard haphazardly without affecting the original result?"*


Sign in to Yahoo!

Are you protected?
Create your sign-in seal.

Yahoo! ID
okla_user
(e.g. free2rhyme@yahoo.com)

Password

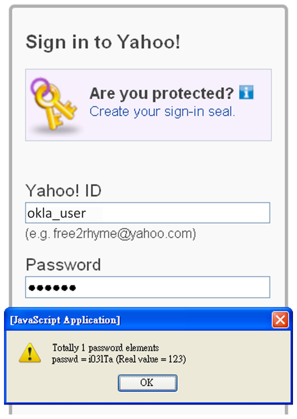
Figure 1. A view of canvas components.

2) Preliminary QTE Utilization: Remind that what users do will be ignored if they do not respond QTE in time. For this property, the QTE method adds a canvas to visualize both valid and invalid input areas on the screen. Furthermore, it divides the canvas into three parts: background, hit area, and moving foreground as shown in Figure 1. The hit area and background do not overlap with each other.

The most crucial parts of the QTE method are the hit area and the foreground. The foreground can be one or more moving objects that slide over the background. A user's input will be memorized by the QTE add-on extension as the characters of a password only if one moving object is over the hit area when the user presses the keyboard. The design therefore gives users a way to obfuscate most keyloggers by hitting their keyboards haphazardly when no foreground object is over the hit area. Besides, when the cursor is in a password filed, no matter what key a user presses on the keyboard, we replace it with an alphanumeric character and append it to the string in the password field to make keyloggers harder to retrieve the real password. The alphanumeric character is randomly generated. An example of this way is given in Figure 2, where we typed "`ABC`" when the moving foreground was over the background and "`123`" when it was over the hit area. Then the password field was stuffed with meaningless characters, "`i031Ta`", while the real password was stored by the QTE add-on somewhere in the heap segment. Afterwards, the real password was copied to the password field just before the form was submitted; therefore, users could still log in as though nothing was changed.

Despite the fact that the preliminary way seems sturdy enough to resist most keyloggers since users really press on keyboards; it is still vulnerable to those who can capture the screen whenever a user presses a key. To make the QTE method more secure, we revise the preliminary way slightly and advise an enhanced version.

3) Enhanced QTE Utilization: To begin with, we discuss functions about screen snapshots. So far as we know,


Sign in to Yahoo!

Are you protected?
Create your sign-in seal.

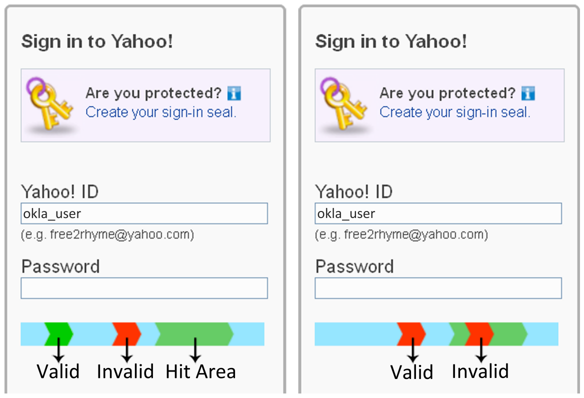
Yahoo! ID
okla_user
(e.g. free2rhyme@yahoo.com)

Password
●●●●●●

[JavaScript Application]

Totally 1 password elements
passwd = i03lTa (Real value = 123)

Figure 2. The result of typing ABC in background and 123 in hit area.


Sign in to Yahoo!

Are you protected?
Create your sign-in seal.

Yahoo! ID
okla_user
(e.g. free2rhyme@yahoo.com)

Password

Figure 3. (a) The real entity was colored differently at the first second. (b) The real entity masqueraded itself since the next second.

commercial spyware takes a screen-shot in three situations according to periodical, event-triggered, and manual conditions; hence, it might be insufficient to take our first countermeasurement merely, if keyloggers are able to capture the screen. Nevertheless, screen capturing, which chose by the majority of spyware, has a great discrepancy with regard to screen recording: while the former simply gets a frame of a time point, the latter keeps track of every instant a user acts. Therefore screen capturing, in all probability, may miss to take a screen-shot at some critical moments, if they do not happen at the time specified a keylogger. Thus, if the events used to hint a user when she/he could type a valid character should be inferred from the continuous time, screen snapshots will no longer work.

Our prototype was inspired by the DDR machine, a musical game which requires users to tread on corresponding buttons when arrows on the screen become valid in a specific area. As the instances given in Figure 3, we randomly generate valid and invalid objects moving from left to right over the background and hit area. Keystrokes will be memorized as parts of a password only if a valid object is over the hit area. Both valid and invalid objects have the same shape. All valid objects have the same initial color, so do invalid objects. But the initial color of valid objects is different from the color of invalid objects. When approaching a hidden boundary, valid objects will disguise themselves as invalid objects by changing their color to the color of invalid objects. In this way, even keyloggers have the ability to log the screen whenever a user presses a key, they still cannot obtain a password.

By virtue of QTEs, we bring up two simple ways that are immune to most kernel, hypervisor, hardware, and second-channel keyloggers merely with normal privileges. While the first measure can not withstand those who can make snapshots, we reinforce our work and make an enhanced version that is resistant to them. Nonetheless, there might be few scenarios which demand the most secure and confidential procedure while screen recording is possible. With a view to apply for such circumstances, we address another innovative solution which works both safely and efficiently.

IV. Implementation

We implemented our solutions as a Firefox extension for the reason that every user can install the extension alone without special privileges of the related computer.

In order to visualize QTEs, we use the canvas element newly introduced in HTML5 standard [37]. Although the canvas element makes it easier to do animations in JavaScript, it costs more resources to maintain and repaint the same thing in different positions since the canvas element does not remember what it had drawn. As a result, we create only one canvas element per page to save performance overhead.

In practice, we firstly find all password fields when a webpage is loaded, and add event listeners for each password control. On this ground, we can easily insert or remove the canvas element when a control is focused or blurred. Secondly, we replace the default keyboard events of password fields with our method, which checks whether there exists at least one foreground object sliding over the hit area when detecting a keystroke. If the condition is true, we will store the key somewhere in the heap segment. However, when the cursor is in a password control, no matter what key a user presses on the keyboard, we replace it with a random alphanumeric character and append it to the string in the password control to simulate the original behavior. Eventually, when the form is being submitted, we restore the password control with the password string we have memorized.

V. Evaluation

Our evaluation is composed of two parts. Firstly, we examine the effectiveness of our solutions with seven different user-space or kernel-space keyloggers.

Table I
THE LIST OF KEYLOGGERS WE HAVE TESTED.

Keylogger	Level	Clip-board	Screen-shots	Packets Logging	Solve
Home	User	✓	✗	✗	✓
Family	User	✓	✗	Only URL	✓
Revealer	Kernel	✗	✗	✗	✓
Refog	Kernel	✓	✓	Only URL	✓
Ardamax	Kernel	✓	✓	Only URL	✓
All-In-One	Kernel	✓	✓	Only URL	✓
SpyAgent	Kernel	✓	✓	✓	✓

A. *Experiments*

To meet the real circumstances, we created a privilege-limited account on Windows XP SP3 in our machine. Besides, our system was installed with the Firefox web browser and seven different keyloggers including free and commercial trials. After that, we changed to the limited account and installed our extension without any special privileges. The list of keylogger features and experimental results are given in Table I.

At first, we tested the QTE method by typing the real password when a foreground was sliding over the hit area; otherwise, we just tapped on the keyboard arbitrarily. In the end, we found that the noise we produced were recorded by all keyloggers and made the real password indistinguishable. The experiment proved that keys we hit outside the hit area can successfully obfuscated attackers since the real password characters were interleaved with irregular strings. As a matter of fact, the outcome can be easily predicted since we really pressed the keys.

To sum up, our experiments showed the feasibility that one can bypass the monitor of keyloggers in web browsers even though the user has lower privileges than those of keyloggers'.

B. *Attack Analysis*

In this subsection, we analyze possible strategies that attackers can adopt to bypass QTE's protection.

If an attacker can obtain multiple keystroke samples which are used to input the same password under a QTE system, through common substrings analysis, the attacker can greatly increase his chance to get the real passwords. However, because a user usually will not use the same public computer frequently, an attacker may not be able to collect enough samples through a public computer. Besides, by adopting appropriate approaches to input a password, a user can increase the difficulty to leak the real password. What follows is an example. Assume `egi` is a password. A user can type a sequence of '`a`' first, then a sequence of '`b`', and then a sequence of '`c`', and so on, until he types a sequence of '`z`'. But only when '`e`', '`g`', and '`i`' are considered, the user will type them when the moving foreground is over the hit area. No other character will be typed when the moving foreground is over the hit area. As a result, no matter how many times the password is typed, a keylogger can only obtain a sequence of '`a`' followed by a sequence of '`b`' followed by a sequence of '`c`', and so on. The above information is not useful for the attacker. In the future, we plan to develop a new QTE component to execute the above operations automatically; thus, a user can input his password more quickly.

VI. CONCLUSIONS

While most solutions either require root privileges or leak users' information to service providers, we propose the QTE method to bypass most kernel, hypervisor, hardware, and second-channel keyloggers in the user-space even with a privilege-limited account. In general, the QTE method gives users a chance to obfuscate keyloggers by tapping keyboards haphazardly without affecting the original results. Moreover, we have tested seven different user-space and kernel-space keyloggers, and none of them could get our real passwords.

A. *Contributions*

Our work demonstrates the feasibility of bypassing almost all keyloggers, even users only have lower privileges, and it causes no latency after logging into the related services. Furthermore, the QTE method is compatible with all web-sites without the need of their support. To conclude, one can protect himself from keystroke logging with the QTE method. The QTE method can conquer nearly all keyloggers presumed the attacker will not record the screen and analyze them manually.

B. *Future Work*

In the future, one thing that we plan to finish is applying our solutions to all programs more than web browsers. For the moment, we can only catch edit controls of applications which use Win32 API to create user interface, and our approaches can be applied to them in the same way except that, instead of automatically inserting canvas or information tips next to password fields, users have to select the edit control on the screen manually. For other programs written in Java or some languages that draw edit controls in their ways, we can only make a screen-shot to ask users the position of edit controls, and simulate the mouse events to focus on it and fill passwords by user-space keystroke events. In most cases, since keyloggers do not want to be obfuscated by user-space events mentioned previously [6], [7], it may be a safer way to enter passwords in this manner at present.

Acknowledgment

Our work is funded by National Science Committee of Taiwan (ROC), and the numbers of the projects are NSC 99-2220-E-008-001 and NSC 99-2219-E-008-001.

References

[1] Symantec, Symantec Internet Security Threat Report: Trends for 2010, vol. 16, Symantec, 2011.

[2] B. News, "UK police foil massive bank theft," BBC News, BBC News, 2005.

[3] InfoWatch, Global Data Leakage Report 2010, InfoWatch, 2011.

[4] P. S. Kivolowitz. "A Program To Allow ANYONE To Crack Unix (4.1 and 2)," http://securitydigest.org/unix/archive/006 06.11.2012.

[5] S. Ortolani, C. Giuffrida, and B. Crispo, "Bait your hook: a novel detection technique for keyloggers," Proc. the 13th International Conference on Recent Advances in Intrusion Detection, Ottawa, Ontario, Canada, 2010, pp. 198-217.

[6] Wuul, "AntiKeylogger," 2007.

[7] Wuul, "Log This!," 2008.

[8] SpyReveal, "SpyReveal," 2009.

[9] F. Mihailowitsch, Detecting Hardware Keylogger, 2010.

[10] F. J. Cini, Keystroke Encryption System, US, E. P. Kristina M. Grasso;Kristina M. Grasso, 2010.

[11] M. Kassner. "KeyScrambler: How keystroke encryption works to thwart keylogging threats," http://www.techrepublic.com/blog/security/keyscrambler-how-keystroke-encryption-works-to-thwart-keylogging-threats/4648 06.11.2012.

[12] A. Young and Moti Yung , "Deniable password snatching: on the possibility of evasive electronic espionage," Proc. IEEE Symposium on Security and Privacy, 1997 , pp. 224-235

[13] J. M. McCune, A. Perrig, and M. K. Reiter, "Bump in the ether: a framework for securing sensitive user input," Proc. the annual conference on USENIX '06 Annual Technical Conference. pp. 17-17.

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

[15] M. Hirano, T. Umeda, T. Okuda, E. Kawai, and S. Yamaguchi, "T-PIM: Trusted Password Input Method against Data Stealing Malware," 2009 Sixth International Conference on Information Technology: New Generations, 2009, pp. 429-434.

[16] K. Chen, "Reversing and Exploiting an Apple®Firmware Update," in Black Hat, Las Vegas, Nevada, 2009.

[17] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," Proc. the 12th ACM conference on Computer and communications security, Alexandria, VA, USA, 2005, pp. 373-382.

[18] Y. Berger, A. Wool, and A. Yeredor, "Dictionary attacks using keyboard acoustic emanations," Proc. the 13th ACM conference on Computer and communications security, Alexandria, Virginia, USA, 2006, pp. 245-254.

[19] A. Kelly, "Cracking Passwords using Keyboard Acoustics and Language Modeling," School of Informatics, The University of Edinburgh, Edinburgh, 2010.

[20] M. Vuagnoux, and S. Pasini, "Compromising electromagnetic emanations of wired and wireless keyboards," Proc. the 18th conference on USENIX security symposium, Montreal, Canada, 2009, pp. 1-16.

[21] K. Zhang, and X. Wang, "Peeping tom in the neighborhood: keystroke eavesdropping on multi-user systems," Proc. the 18th conference on USENIX security symposium, Montreal, Canada, 2009, pp. 17-32.

[22] A. Barisani, and D. Bianco, "Sniff Keystrokes With Lasers/Voltmeters - Side Channel Attacks Using Optical Sampling Of Mechanical Energy And Power Line Leakage," in DEFCON 17, Riviera Hotel and Casino, 2009.

[23] M. N. Doja, and N. Kumar, "Image Authentication Schemes against Key-Logger Spyware," in ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing. pp. 574-579.

[24] H. Wei, W. Xiaoping, and W. Guoheng, "The Security Analysis of Graphical Passwords," in International Conference on Communications and Intelligence Information Security. pp. 200-203.

[25] E. Stobert, A. Forget, S. Chiasson, P. C. v. Oorschot, and R. Biddle, "Exploring usability effects of increasing security in click-based graphical passwords," Proc. the 26th Annual Computer Security Applications Conference, Austin, Texas, 2010, pp. 79-88.

[26] A. P. Waterland, Secure password entry, NY, USA,to International Business Machines Corporation, 2009.

[27] K. Lab. "Kaspersky Internet Security 2009 includes a virtual keyboard that enables users to enter logins and passwords safely," http://www.kaspersky.com/news?id=207575675 06.11.2012.

[28] D. Hoover, Method and apparatus for secure entry of access codes in a computer environment, CA US Patent No. 6,209,102,to Arcot Systems, Inc., 2001.

[29] J. Brill, "More Ways to Stay Secure," More Ways to Stay Secure | Facebook, October 12, 2010, Facebook, 2010.

[30] N. Shah, "Advanced sign-in security for your Google account," Official Google Blog: Advanced sign-in security for your Google account, February 10, 2011, 2011.

[31] M. Wu, S. Garfinkel, and R. Miller, "Secure Web Authentication with Mobile Phones," in DIMACS Workshop on Usable Privacy and Security Software, 2004.

[32] A. Pashalidis, and C. J. Mitchell, "Impostor: a single sign-on system for use from untrusted devices," in IEEE GLOBECOM. pp. 2191-2195.

[33] E. Gieseke, and J. McLaughlin, "Secure Web Authentication with Mobile Phones Using Keyed Hash Authentication," Computer Science, Harvard University Extension, 2005.

[34] A. Pashalidis, "Accessing Password-Protected Resources without the Password," in WRI World Congress on Computer Science and Information Engineering. pp. 66-70.

[35] F. Dinei, and H. Cormac, "KLASSP: Entering Passwords on a Spyware Infected Machine Using a Shared-Secret Proxy," in Computer Security Applications Conference, 2006. ACSAC '06. 22nd Annual. pp. 67-76.

[36] T. Rogers, "Full Reactive Eyes Entertainment," Game Developer, December, 2010, 2010.

[37] A. C. Inc., M. Foundation, and O. S. ASA., "HTML Living Standard," Ian Hickson, 2011, pp. 288-315.

AWESOME - Automated Web Emulation for Secure Operation of a Malware-Analysis Environment

Martin Brunner, Christian Martin Fuchs and Sascha Todt
Fraunhofer Research Institution for Applied and Integrated Security (AISEC)
Munich/Garching, Germany
{*martin.brunner,christian.fuchs,sascha.todt*}*@aisec.fraunhofer.de*

***Abstract*—We present AWESOME, a novel approach for integrated honeypot-based malware collection and analysis which extends the functionalities of existing approaches. In contrast to purely network-based approaches, the goal of our collection and analysis system is runtime retrieval of internal malware logic information. This approach allows us to provide analyzed malware with all requested resources in real time, despite the fact that it is executed within an isolated environment. Our assumption is that being able to track the entire malware execution life-cycle will enable a better understanding of current and emerging malware. This paper introduces our design, outlining its contributions and design considerations. An in-depth description and evaluation of each component will be discussed in separate work. While still under development, we expect our approach to make a significant contribution to enhanced analysis of current malware.**

Keywords-malware collection; malware analysis and defense.

I. Introduction and Related Work

Some of today's most disruptive cyber threats can be attributed to malware, usually organized within a botnet in large-scale scenarios. This results in a fundamental need to track the rapid evolution of malware, which, in turn, depends on collection and examination of current real-world attack data, commonly acquired through meticulous analysis of the most recent samples. The evolution of malware over time has led to the development of intensive obfuscation and anti-debugging mechanisms, as well as a complex and multi-staged malware execution life-cycle [16]. Usually, this life-cycle can be partitioned into three phases: (i) *propagation and exploitation*, which covers the spread of malicious payloads and unpacking and deobfuscation of the corresponding shellcode; (ii) *infection and installation* covers the deployment of a binary (e.g., dropper) on the victim host followed by possible preparatory activities and the retrieval of the actual malware body; and, (iii) the *operation and maintenance* phase, in which the malware's core components harvest valuable information and attempt to establish a command-and-control (C&C) channel awaiting further instructions. Each phase may include numerous measures aimed at maximizing installation success and reliability. In order to comprehensively analyze the full life-cycle, the malware under analysis must have unhindered access to all requested resources during runtime. While this could easily be achieved by allowing full interaction with the Internet, this is not a viable approach in setups which are forced to consider liability issues. Advanced large-scale malware collection and analysis infrastructures, such as [1][5][8], can satisfy the requirements for automated tracking of malware, but suffer from several limitations:

(i) Despite being executed within an isolated environment, samples must be supplied with requested network services during analysis. Otherwise, achieving high-quality results is impeded, potentially causing different malware behavior or even a refusal of execution. While certain services can be offered using sinkholing techniques, existing approaches are purely network-based, and reactions to malware-initiated connection attempts remain static during runtime. Predefined commonly-used services, which are usually queried, are offered by these infrastructures to the malware; however, other requests can not be handled accordingly.

(ii) High-interaction (HI) honeypots pose, aside from their complexity and maintenance issues, high operational risks. These are often inadequately addressed. While there are many methods of mitigation, the remaining risk is still higher than with low-interaction (LI) honeypots, resulting in ethical questions and possibly even legal and liability issues for the operating organization.

(iii) Existing approaches separate collection and analysis, thus forfeiting the system context (i.e., file handles, requests, sockets) of the victim host. While such separation is not necessarily a limitation (it may not be mandatory to gain qualitative analysis results), we argue that this loss of information hinders analysis and may degrade analysis results or prevent analysis of certain malware altogether.

II. Approach

A. Goals

Our overall goal is to capture and dynamically analyze novel malware on a large scale. To identify trends of current and emerging malware, we aim to cover the entire life-cycle. That is, we want to track malware communicating via unknown (e.g., C&C) protocols in an automated way within a controlled environment. In order to minimize harm to third parties, malware should by default have no Internet access during analysis. The whole procedure intends to trick

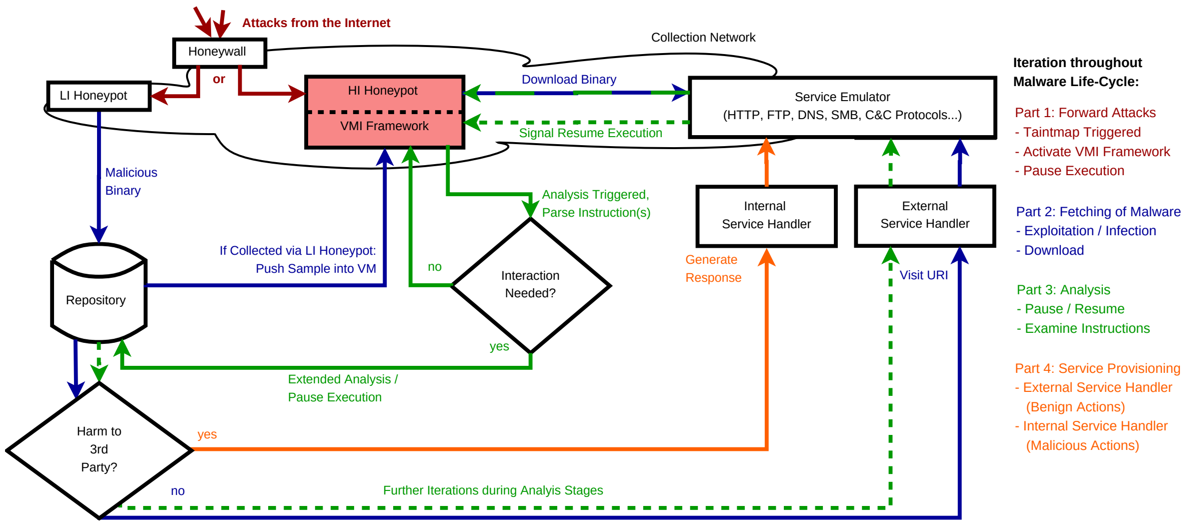


Figure 1. General Design of the Presented Approach

a sample into believing it is running on a real victim host with full Internet access.

B. Basic Concept

To identify the services and protocols required in the next step of the life-cycle, we intend to harvest information on internal malware logic during execution. In contrast to purely network-based approaches, our method also operates at the binary level, directly interacting with the malware's host system. It therefore aims to integrate network-based analysis and binary analysis, as in [21]. As depicted in Figure 1, the presented approach is based on a HI honeypot and a virtual machine introspection (VMI) framework. We enhance our architecture with a transparent pause/resume functionality, which is instrumented to determine and, if needed, interrupt the program flow. Hence, we enable the extraction and alteration of program logic and data within the victim environment during runtime. This is specifically valuable for extracting protocol information and cryptographic material embedded within malware in order to determine the protocol type and intercept encrypted communication. After checking, extracted information is forwarded to a service handler (SH) and sinkholing service in order to maintain full control over all interactions between the malware and the outside world. For handling unknown traffic as well, finite state machines (FSM) are automatically derived from the observed traffic and used for service emulation. An important goal of automating the whole collection and analysis process is to handle large amounts of malware while allowing scalability.

C. Added Value

The system context of the malware collection facility persists and is also used in the subsequent analysis. The capabilities resulting from the merge of collection and analysis is similar to the approach used in HI honeypots. Thus, it is more closely aligned to real-world scenarios than LI honeypots. In addition, we achieve increased transparency during analysis due to the use of VMI. We consider this to be a benefit, since we argue that VMI based analysis is more likely to remain undetected by malware. Compared to other techniques, VMI requires no trusted support components which could be compromised [7] inside the sample's context of execution. Hence, the approach is more likely to observe the entire malware execution life-cycle. Furthermore, we are able to extract and inject data as well as instructions from or into the memory of the infected virtual machine (VM) during runtime (for example, in order to tap and manipulate encrypted C&C traffic). Since our approach does not depend on analysis components within the VM, we believe it to be more secure while also expecting better overall performance. Moreover, we are able to control any interaction between malware and third party systems. Thus, our architecture can fulfill legal and liability constraints. Since our approach is applied directly at the instruction level, we are aware of the actions initiated by the malware, thus allowing us to provide matching services and even to service novel communication patterns. Subsequently, the risk resulting from HI honeypot operation is minimized.

III. Design and Implementation

A. Components

Our approach utilizes the following components:

For *malware collection*, a modified HI honeypot [18] is used. *Malware analysis* is conducted based upon Nitro [17], a KVM-based framework for tracing system calls via VMI. In particular, we determine whether a given action initiated by the currently-analyzed malware requires Internet access and thus apply a complex rule-set to the tracing component.

Our *service provisioning* component manages all malware-initiated attempts to request Internet resources. Malicious attempts are handled via an appropriate sinkholing service spawned by Honeyd [19], and unknown traffic patterns may be handled utilizing ScriptGen [13].

B. Setup

While most popular LI honeypots have proven to be efficient for malware collection, their knowledge-based approach has also drawbacks regarding the quantity and diversity of the collected malware [23]. With respect to our primary goal (to handle unknown malware), we chose to apply the taint-map-based approach of ARGOS, since it allows the detection of both known and unknown (0-day) attacks. In addition, it is independent of special collection mechanisms. Moreover, it can cooperate with the KVM based VMI framework Nitro. Hence, several components were modified: (i) the victim VM's RTC is detached from the host's clock, since ARGOS is more time-consuming than traditional approaches and thus detectable by an abnormal latency and timing-behavior; (ii) once the taint-map reports tainted memory being executed, we activate the analysis functionality provided by the VMI framework; and, (iii) simple interpretation and filtering of system calls and their parameters is conducted directly within hypervisor space, while more complex analysis is performed via the VMM in the host environment [10]. The entire process consists of three parts (collection, analysis, and service provisioning) and is structured as described below. The steps are repeated iteratively throughout the entire life-cycle of the malware.

C. Malware Collection

To overcome the poor performance of ARGOS, we build a two-staged malware collection network. That is, we deploy a hybrid honeypot system similar to existing approaches, such as [2][11][22]. We then take advantage of our preexisting honeyfarm infrastructure [3], which utilizes a large-scale network telescope employing various different LI honeypots. This infrastructure is used to filter noise and known (and thus uninteresting) attacks. Only novel incidents are forwarded to ARGOS, thus reducing the overall load on it.

D. Malware Analysis

Dynamic malware analysis utilizing virtualization can be recognized and thus evaded by environment-sensitive malware [9][10][14][20]. Hence, our goal is to achieve a reasonably transparent dynamic malware analysis; however, we also consider VMI as the most promising available approach to evade malware's anti-debugging measures. Thus, in order to provide the best chance at evading detection while still gaining the benefits of VMI, we have chosen Nitro since it offers several advantages regarding performance and functionality when compared to other publicly available tools such as Ether (see [17]). As Nitro is based on KVM, we have, in addition to guest portability, full virtualization capability, thanks to the host CPU's virtualization extensions; thus, we can expect reasonable performance. During the analysis process, we expect a malicious binary to be shellcode or a dropper rather than the actual malware binary. This initially retrieved binary is then decoded and usually contains a URL pointing at the resource used for deploying the next stage of the malware. In the second iteration, execution of this binary continues after it has been downloaded and the VM has been resumed. The resulting system call trace is then examined for routines related to connection handling (e.g., NTConnectPort). If present, we transparently pause execution of the VM and forward related traffic to the service provisioning component.

E. Service Provisioning

Malware-driven outbound requests are evaluated to prevent harm to third party systems. For these checks, we rely upon existing measures, such as IDSs or a web application firewall. We are aware that such measures will not be sufficient to tell benign and malicious flows apart in every case; thus, we may build on existing approaches [12]. We assume that a purely passive request (e.g., a download) does not cause harm to a third party. It is thus considered to be benign and handed over to the *external service handler* (see Figure 1). Since the external SH has Internet access, it resides in a dedicated network segment separated from the analysis environment. If a given request can not be determined to be benign, it is redirected to the *internal service handler*. The sole task of these SHs is to fetch, prepare and provide information for the *service emulator* (SE). The SE launches the requested service in order to deliver the appropriate payload supplied by the SH. Afterwards, execution is transparently resumed. Since these services can be extremely heterogeneous, the SE is based on Honeyd. It is a very flexible and scalable tool which is able to emulate or spawn arbitrary services, given that a protocol template exists.

The creation of templates for novel protocols is a much more challenging task. Therefore we plan to instrument a tool, which derives FSMs from observed traffic, such as *ScriptGen* or a similar approach [4][6][15]. Each FSM represents the behavior of a given protocol at an abstract level while not depending on prior knowledge or protocol semantics. Based on the generated FSMs, service emulation scripts for the SE can be derived. By integrating such a tool into our approach, we aim toward adding 'self-learning capabilities' to the service provisioning element. Obviously this requires (at least) one-time observation of a given communication between the honeypot and the external system. Hence we need a (supervised) back-channel for learning about novel protocols. Once a corresponding communication has been recorded and the appropriate FSM has been generated, we are able to handle the new protocol as well. While the need for a back-channel is a clear limitation, we consider it to be a reasonable trade-off.

IV. Summary and Future Work

In this paper, we have presented a novel approach for integrated honeypot-based malware collection and analysis which extends existing functionalities. Specifically, it addresses the separation of collection and analysis, the limitations of service emulation, and the operational risk of HI honeypots. The key contribution of the approach is the design of the framework as well as the integration and extension of the stated tools. While this is an ongoing research activity and thus still under development, several modifications to ARGOS and Nitro have already been implemented and successfully tested, indicating the feasibility of our approach. Future work will include the completion and evaluation of the service emulator and the measures to prevent harm to third party systems.

References

[1] M. Apel, J. Biskup, U. Flegel, and M. Meier. Early warning system on a national level - project amsel. In *Proc. of the European Workshop on Internet Early Warning and Network Intelligence (EWNI 2010)*, January 2010.

[2] M. Bailey, E. Cooke, D. Watson, F. Jahanian, and N. Provos. A hybrid honeypot architecture for scalable network monitoring. In *Technical Report CSE-TR-499-04*, 2006.

[3] M. Brunner, M. Epah, H. Hofinger, C. Roblee, P. Schoo, and S. Todt. The fraunhofer aisec malware analysis laboratory - establishing a secured, honeynet-based cyber threat analysis and research environment. Technical report, Fraunhofer AISEC, September 2010.

[4] J. Caballero, P. Poosankam, C. Kreibich, and D. Song. Dispatcher: enabling active botnet infiltration using automatic protocol reverse-engineering. In *Proc. of the 16th ACM Conference on Computer and Communications Security*, CCS '09, pages 621–634, New York, NY, USA, 2009. ACM.

[5] D. Cavalca and E. Goldoni. Hive: an open infrastructure for malware collection and analysis. In *proc. of the 1st Workshop on Open Source Software for Computer and Network Forensics*, 2008.

[6] W. Cui, V. Paxson, Nicholas C. Weaver, and Y H. Katz. Protocol-independent adaptive replay of application dialog. In *13th Annual Network and Distributed System Security Symposium (NDSS)*, 2006.

[7] M. Dornseif, T. Holz, and C.N. Klein. Nosebreak - attacking honeynets. In *Information Assurance Workshop, 2004. Proc. from the Fifth Annual IEEE SMC*, June 2004.

[8] M. Engelberth, F. Freiling, J. Göbel, C. Gorecki, T. Holz, R. Hund, P. Trinius, and C. Willems. The inmas approach. In *1st European Workshop on Internet Early Warning and Network Intelligence (EWNI)*, 2010.

[9] P. Ferrie. Attacks on virtual machine emulators. In *AVAR Conference, Auckland*. Symantec Advanced Threat Research, December 2006.

[10] C. M. Fuchs. Deployment of binary level protocol identification for malware analysis and collection environments. Bacherlor's thesis, Upper Austria University of Applied Sciences Hagenberg, May 2011.

[11] X. Jiang and D. Xu. Collapsar: a vm-based architecture for network attack detention center. In *Proc. of the 13th conference on USENIX Security Symposium - Volume 13*, SSYM'04, Berkeley, CA, USA, 2004. USENIX Association.

[12] C. Kreibich, N. Weaver, C. Kanich, W. Cui, and V. Paxson. Gq: practical containment for measuring modern malware systems. In *Proc. of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 397–412, New York, NY, USA, 2011. ACM.

[13] C. Leita, K. Mermoud, and M. Dacier. Scriptgen: an automated script generation tool for honeyd. In *Proc. of the 21st Annual Computer Security Applications Conference (ACSAC)*, Washington, DC, USA, 2005. IEEE.

[14] M. Lindorfer, C. Kolbitsch, and P. Milani Comparetti. Detecting environment-sensitive malware. In *Recent Advances in Intrusion Detection (RAID) Symposium*, 2011.

[15] P. Milani Comparetti, G. Wondracek, C. Kruegel, and E. Kirda. Prospex: Protocol specification extraction. In *Proc. of the 30th IEEE Symposium on Security and Privacy*, pages 110–125, Washington, DC, USA, 2009.

[16] G. Ollmann. Behind today's crimeware installation lifecycle: How advanced malware morphs to remain stealthy and persistent. Whitepaper, Damballa, 2011.

[17] J. Pfoh, C. Schneider, and C. Eckert. Nitro: Hardware-based system call tracing for virtual machines. In *Advances in Information and Computer Security*, volume 7038 of *Lecture Notes in Computer Science*. Springer, November 2011.

[18] G. Portokalidis, A. Slowinska, and H. Bos. Argos: an emulator for fingerprinting zero-day attacks. In *Proc. ACM SIGOPS EUROSYS'2006*, Leuven, Belgium, April 2006.

[19] Niels Provos. A virtual honeypot framework. In *Proc. of the 13th USENIX Security Symposium*, 2004.

[20] J. Rutkowska. Red pill... or how to detect vmm using (almost) one cpu instruction, 2004. http://invisiblethings.org [retrieved: July, 2011].

[21] D. Song, D. Brumley, H. Yin, J. Caballero, I. Jager, M. Gyung Kang, Z. Liang, J. Newsome, P. Poosankam, and P. Saxena. Bitblaze: A new approach to computer security via binary analysis. In *Proc. of the 4th International Conference on Information Systems Security.*, 2008.

[22] M. Vrable, J. Ma, J. Chen, D. Moore, E. Vandekieft, A. C. Snoeren, G. M. Voelker, and S. Savage. Scalability, fidelity, and containment in the potemkin virtual honeyfarm. In *Proc. of the 20th ACM Symposium on Operating Systems Principles*, SOSP '05, New York, NY, USA, 2005. ACM.

[23] J. Zhuge, T. Holz, X. Han, C. Song, and W. Zou. Collecting autonomous spreading malware using high-interaction honeypots. In *Proc. of the 9th International Conference on Information and Communications Security*, ICICS'07, Berlin, Heidelberg, 2007. Springer-Verlag.

Preserving Continuity of Services Exposed to Security Incidents

Dariusz Caban, Tomasz Walkowiak
Institute of Computer Engineering Control and Robotics
Wroclaw University of Technology
Wroclaw, Poland
dariusz.caban@pwr.wroc.pl

***Abstract*—System reconfiguration is often the only means of preserving the continuity of business-critical services after a successful penetration attack. When a fragment of the system has to be isolated (to prevent escalation of damages caused by the incident), then the other system components need to take over the functions normally performed by the affected servers. The paper addresses a specific aspect of this reconfiguration – the need to predict its impact on the availability of the services. It proposes an efficient approach, based on network simulation, for assessing the response times of services after reconfiguration. This takes into consideration the identified resource consumption interactions between the co-located services. All the presented simulation results are verified on testbed installations. The accuracy of the simulation method allows prediction of the risk of system overloading as an effect of reconfiguration.**

Keywords-system dependability, simulation, reconfiguration

I. INTRODUCTION

It is always a strategic issue on how to react when one detects an attempted security breach or a concerted denial-of -service attack. There are a lot of publications addressing this problem – usually from the point of view of apprehending the culprit and identifying the potential damage.

In this paper, we consider a different aspect of the problem: it is often critically important from the business and reputation point of view to preserve the continuity of service, disregarding the potential risk of damage escalation. In such situations, the administrator is faced with two options: to continue all the services as is (usually a totally unacceptable approach) or to isolate the affected part of the system and redeploy the service components to the unaffected servers. This forced reconfiguration is considered hereafter, especially the prediction of adverse service interactions that may impact the services availability (and thus may also be used by the aggressor to achieve denial-of-service).

Network simulation is normally used to analyze service protocols and interactions. It is generally not used to analyze performance, as it is very difficult to obtain service timing of adequate precision. Thus, when a system is being deployed it is a standard procedure to precede it with testbed experiments. In case of a forced reconfiguration, there is no time for such preparations. On the other hand, the data collected prior to reconfiguration (on the live system) can be used to fine-tune the simulation models. Thus, it is postulated that, in this case, the simulation should be used. It ensures fast access to the predictions of adequate precision.

The paper first presents the specification of the system that is considered for reconfiguration (Section III). Specifically, it introduces the concept of configuration, as it is used throughout the paper. Then, in Section IV, the idea of redeployment of services (reconfiguration) aimed at preserving the continuity of service is discussed. Sections V and VI present the results of experiments with predicting the service availability and response times of reconfigured system using simulation. The results are validated on a virtualized testbed installation.

II. RELATED WORK

The paper addresses some issues connected with relocating services as a reaction to security incidents. This is a well-documented concept, proposed in the NIST Computer security incident handling guide [17] – to “isolate the affected system or network” and to “relocate the target”. The guide does not propose any specific methods to achieve this, though. These are addressed in other works, a review of which is given in [9]. A policy-driven approach to this reconfiguration was proposed by the European Union VI Framework Project “Dependability and Security by Enhanced Reconfiguration DESEREC” [1, 5]. This approach is extended in the paper.

It is a non-trivial problem to predict the effects of service relocation on the system quality of service, especially its availability and response time [7, 18]. Mainly, the approach is based on the queuing networks [8] that describe the services in the underutilization or normal utilization ranges [19]. Usually, they don’t predict well the transition to the over-utilization range, which is crucial for dependability analysis.

Moreover, there is also a clear gap between the client models used in stress testing and in network modeling and simulation. This is addressed in detail in Section VI.

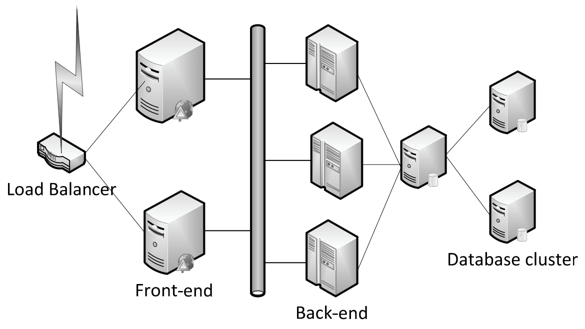


Figure 1. A simple system infrastructure based on multiple levels of load balancing

III. SYSTEM MODEL

The paper considers a very wide class of web based information systems. Business services are accessed by the clients using web interactions. The service responses are dynamically computed by the service components, which also interact with each other using the client-server protocols.

The model is fully compliant with the SOA architecture [4], but it can just as well be applied to the classical solutions based on a front-end web server, communicating with it worker applications and back-end database servers.

The system is described at 3 levels. On the high level, it is represented by the interacting service components. At the physical layer, it is described by the hosts, on which the services are deployed, and by the network communication environment. The intermediate level is the mapping between the service components and the hosts/networks.

A. Physical model

The system consists of network interconnected hosts. The network may either be represented by the physical links between the computers or by the guaranteed levels of throughput via public networks. In either case it is characterized by the throughput which determines the time needed to transmit the data load between hosts (and services deployed on them). The hosts are abstracted to represent the computing resources provided to the service components (the abstraction encompasses hardware, operating systems and server software). Fig. 1 presents a simple network used to provide the services. The configuration encompasses a load balancer, used to distribute requests between the front-end hosts. At the back-end the load is also distributed between the workers.

B. Service model

At this level the system is represented by a set of service components. Interaction between the components is based on the client-server paradigm, i.e., one component requests a service from some other components and uses their responses to produce its own results. In turn, its response is sent either to the end-user or to yet another component.

A service component is a piece of software that is entirely deployed on a single host. All of its communication is done by exchange of messages with end-users or other components.

The overall description of the interaction between the service components is determined by its choreography. In complex systems this choreography is described using either a dedicated language (e.g., BPEL, WS-CDL [13]) or the UML sequence diagrams. In case of simple web-based systems, the usage scenarios are specified informally.

A very simple choreography description is given in Fig.2 for illustration purposes. It represents a common dynamic web service architecture based on an Apache server with Tomcat servlet containers and a back-end SQL database. The system serves static HTML pages and dynamic web-page that requires some computation and database access.

The service components generate demand on the networking resources and on the computational power of the hosts running the components. This demand determines the timing characteristics of the system.

C. System configuration

System configuration is determined by the deployment of service components onto the hosts. This is characterized by the subsets of services deployed at each location. The deployment clearly affects the system performance, as it changes the communication and computational requirements imposed on the infrastructure.

Reconfiguration takes place when service deployment is changed. It can be achieved by redistribution of tasks to be performed by each worker host. This is fairly easily realized by modifying the algorithm used by the front-end servers, responsible for workload distribution among the worker nodes.

IV. RECONFIGURATION POLICY

There may be multiple situations when a change of system configuration is desirable. In most cases the reconfiguration is foreseen well in advance and can be properly planned. In the presented considerations, reconfiguration is used as a technique for ensuring continuity of service. This means that it is performed in reaction to an unforeseeable security incident. And it has to be completed in strict time constraints – the short period of time that the service is allowed to be inaccessible (as prescribed in the requirements for service continuity).

The reconfiguration is done in a hurry, to bring up the system service as quickly as possible. Consequently, it is likely that some side-effects of reconfiguration may be overseen, especially if these are connected with the system performance.

A. Security issues and fault isolation

In these considerations, system reconfiguration is initiated only as a reaction to an *observed* breach of security. This means that we are considering only a limited class of security events that are detected by the deployed monitoring and detection mechanisms: malware detected in the system (virus infections, network worms, etc.), detected unauthorized activity in the system, unexpected modifications of the software, denial-of-service and soft

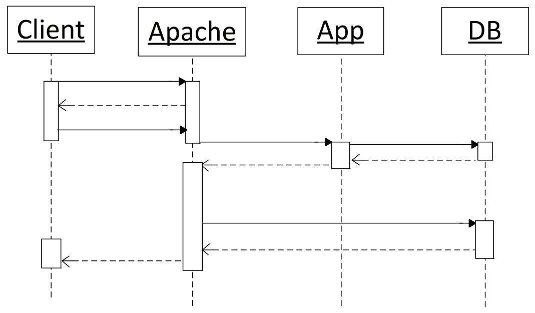


Figure 2. A simple service choreography (UML)

breakdowns, proliferation of bogus service requests and distributed denial of service (DDOS) attacks.

The operation of services located on hosts affected by these types of security breaches becomes unpredictable and potentially dangerous to services located on the other nodes that communicate with them. Thus, the fault may propagate to other connected hosts and services. To prevent this, it is a common practice to isolate the affected hosts to prevent problem escalation [14, 17].

The reconfiguration policy [1,5] starts with the identification of hosts to be isolated. Once this is done, all the services initially located on these hosts have to be moved elsewhere. This is the second step in the reconfiguration policy.

While considering the service components to be redeployed, it is important to identify those, which are essential for the fulfillment of the business requirements. When moving service components there is always a risk of propagating the security issue to the yet unaffected hosts. This can occur if the administrator uses backup data from a restart point that was already affected by the then undetected security breach; or if the redeployed service component has a vulnerability which has already been exploited (unknowingly to the administrator). An important aspect of DDOS attacks is that it may be either IP site locked or service locked. Reconfiguration is effective only in the first case: moving the affected services to other network addresses can prevent further damage. On the other hand, if a service is moved in case of a service locked attack, then the fault will also be propagated to the new location.

When the components to be redeployed are identified, it is necessary to decide where each should be moved to. Obviously, this step of reconfiguration affects not only the availability of services being redeployed, but also of all the services already running on the unaffected hosts. An acceptable reconfiguration policy should ensure that all the services satisfy the minimal requirements regarding their availability. This choice is usually made by the administrator on the basis of his intuition, which is often fallacious.

B. *Continuity of service*

Dependability is defined as the capability of systems to deliver service that can justifiably be trusted [3]. It is an integrative concept that encompasses: availability (readiness for correct service), reliability (continuity of correct service), safety (absence of catastrophic consequences), confidentiality (absence of unauthorized disclosure of information), integrity (absence of improper system state alterations), maintainability (ability to undergo repairs and modifications).

Reconfiguration is used to improve one aspect of dependability: ensure the continuity of service when an incident occurs, which would normally cause it to become unavailable. Let us consider how this continuity can be defined and assessed. To this end, service availability has to be considered.

Availability A is defined as the probability that a system can provide service (i.e., can correctly respond to requests) at a specific moment in time. In stationary conditions, this probability does not depend on the time instance. The asymptotic property of availability can thus be used to predict its value as the ratio of the sum total uptime τ_{up} to the total time τ :

$$A = \tau_{up} / \tau. \tag{1}$$

A very useful property of availability (from the point of view of web services) is derived by transforming the above equation, assuming a uniform rate of service requests [4]. This yields a common understanding of availability as the number of properly handled requests n_{ok} expressed as a percentage of all the requests n :

$$A = n_{ok} / n. \tag{2}$$

The continuity of service is defined as the probability that the service will be available for at least the specified period of time T, i.e., that its availability will not drop below a specified level A_{min}. In fact, short periods of unavailability are practically undetectable (indistinguishable from normal fluctuations in the number of properly handled requests) and do not impact the business notion of continuity. We are only interested in predicting the probability that in the time horizon T the availability never drops below A_{min} for longer than some small time needed to react to the incidents.

Actually, it is not necessary to evaluate the measure of continuity of service. It is obvious that reconfiguration will improve it, if and only if it can be completed in adequately short time and the availability after reconfiguration is greater than A_{min}. In other words, a breach of security will not cause a discontinuity of the service if there is an alternative configuration that is not affected by the breach, that ensures the required availability of service, and the reconfiguration can be completed in short time.

Availability and continuity of service do not reflect the comfort of using the service by the end-users. This has to be analyzed using the response time, i.e., the time elapsed from the moment of sending a request until the response is completely delivered to the client [21]. The average is calculated only on the basis of correctly handled response times. The error response times are excluded from the assessment (or assessed as a separate average).

V. Network simulation

Network simulators are plentiful, both open-source (SSFNet [11], ns3 [16], Omnet+ [20]) and commercial. Most of them simulate the transport algorithms and package queues [8]. These simulators can fairly well predict the network traffic, even in case of load balancing [15]. What they lack is a comprehensive understanding of the computational demands placed on the hosts, and how it impacts the system performance. They are useful to predict the network traffic, not the level of service availability or the response times.

The simulators need to be extended, by writing special purpose models to accommodate the resource consumption model [21, 22]. Availability and response time simulation is based on the models of choreography, end-user clients, service components, processing hosts (servers), network resources.

The network simulator has a number of parameters that have to be set to get realistic results. These parameters are attributed to the various models, mentioned above. In the proposed approach we assume that it is possible to formulate such (fairly simple) models describing the clients and service components, which will not be unduly affected by reconfiguration. Then, we can identify the values of the parameters on the production system. Simulating the target configuration with these parameters should provide reliable predictions of the effects of reconfiguration.

VI. Client – server interaction

The basis of operation of all the web oriented systems is the interaction between a client and a server. In the considered architecture, this interaction can occur between the end-user and the web component that he communicates with. It can also occur between a system component and another server that it queries while processing its requests. The rate of requests processing and the individual response times depend on a number of factors: the processing to be done at the server site, response time of other services that need to be queried to determine the response, etc.

A proper model of client-server interactions is the basis for accurate simulation of the considered system. For this reason, a number of testbed experiments have been conducted to capture the realistic timing characteristics that can be abstracted into a simple model. For this purpose, we have set up a simple testbed, consisting of a virtual machine running an Apache server. The server hosts a PHP script application, on which we can accurately regulate the processing time needed to produce a result. This application is exposed to a stream of requests, generated by a choice of client applications (a Python script written by the authors, open source traffic generators such as Funkload [6] and Apache JMeter [2]). The available processor resources are monitored via the virtualization hypervisor to ensure that the traffic generation programs do not compete for the resources with the system software (which would lead to unrealistic results).

A. Service component response time

The model of a client-server interaction depends a lot on how the traffic is generated by the client. The simplest approach is adopted by the software used for server/service benchmarking, i.e., to determine the performance of computers used to run some web applications. In this case, the server is bombarded with a stream of requests, reflecting the statistics of the software usage. The important factor in this approach is the lack of any feedback between the rate of requests and the server response times. In other words, the client does not wait for the server response, but proceeds to send further requests even if a response is delayed or missing.

Fig. 3 shows the results of stress experiments performed on the testbed application. It should be noted that the results reflect the normal server behaviour in such tests, but the processing thresholds are much lower than expected in modern web servers. This should be expected, since the virtual server has comparatively limited processing power. The response time depends on the rate of incoming requests. It should be noted that the system is characterized by two distinct thresholds in the requests rate. Up to approximately 6 requests per second, the response time very slowly increases with the rate of requests. This is the range, where the server processing is not fully utilized: the processor is mainly idle and handles requests immediately on arrival. There is a gradual increase in the response time due to the increased probability of requests overlapping.

When the requests rate is higher than the underutilization threshold, the processing power is fully used up, the requests are queued and processed concurrently. The increase in the response time is caused by time sharing/queuing: it is proportional to the number of handled requests and the time needed to process a single one. This holds true, until the server reaches the second threshold – overutilization.

In fact, there are at least two different types of thresholds coming into play, depending on the type of server application one of them is dominant. Fig. 3 illustrates the behaviour observed in Apache servers: above the fixed threshold new requests are not processed, but are queued until timeout. In fact, a fixed maximum number of requests are handled correctly and all the rest result in error responses. The response time remains almost constant since the number of processed requests does not increase. On the other hand, the percentage of requests handled incorrectly increases proportionately to the request rate. Similar behaviour is observed in other server applications (MySQL, simpleHTTPD).

A different threshold is observed in some cases, e.g., the IIS server. In this case, the overutilization threshold forces immediate rejects of incoming excessive requests. This does not differ significantly in the observed response time and availability characteristics (in case of the stress benchmarking). But it has other implications when building a simulation model. In fact, all the server applications have this second type of reject threshold (built in the software), in Apache it is simply observed in the range of request rates that should never be allowed to occur.

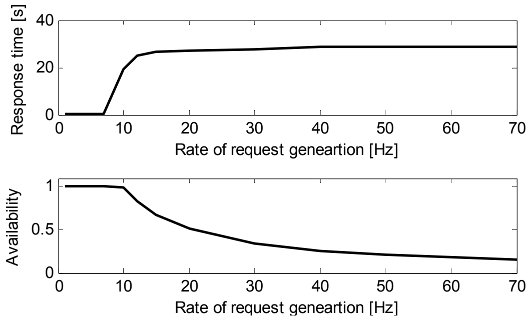


Figure 3. The testbed system performance under varying rates of incoming client requests

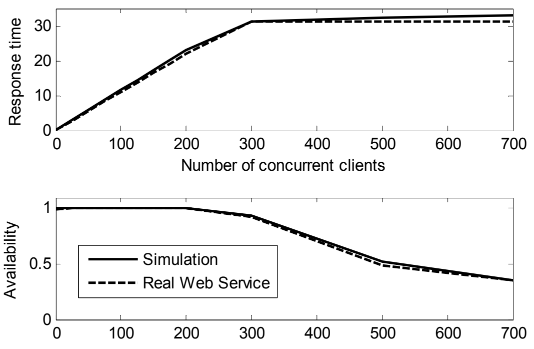


Figure 4. The performance a real web service (dashed line) and simulated one (solid line)

The presented results led to the formulation of a simple model for client-server interaction (used in the simulator:

- The server response time is described by 3 ranges, delimited by the underutilization and overutilization thresholds; a constant response time below the underutilization threshold; a linearly increasing response time in the normal operation range; a constant limit response time when the server is over-utilized.
- The model responds with error messages to some requests when the server is over utilized. Two types of overutilization thresholds are defined, lower determines the error response times in the overutilization range. In case of rejects, the error response time is almost 0 (with a small random factor), in case of timeouts the error response time is random with a fixed average.
- The model is inaccurate in the underutilization range. It does not consider the "worm-up" time observed in this range (probably a side-effect of compiling scripts on the fly and server side caching). Anyway, the model is to be used for determining the load limits, where the underutilization threshold does not affect the results.

B. Realistic client behaviour

The server response time is strongly related to the client behaviour, as determined by the request-response interaction. Such factors as connection persistence, session tracking, client concurrency or client patience/think times have a documented impact. For example, it has been shown in [12] that if user does not receive a service response in less than 10 seconds he or she will probably resign from active interaction with the service and will be distracted to other ones.

The real behaviour of clients differs significantly from the model discussed so far. In fact, the client sends a burst of related requests to the server, then waits for the server to respond and, after some "think" time for disseminating the response, sends a new request. This implies that the request rate depends on the response time. This is implemented in a number of traffic generators (such as the mentioned Apache JMeter and Funkload). The workload is characterized by the number of concurrent clients, sending requests to the server. The actual requests rate depends on the response time and the think time.

Fig. 4 shows how the response time depends on the number of concurrent clients, as observed in the testbed system (with the "think" time set to 0, which corresponds to a new request being generated by the client immediately on receiving the response to a previous one). Even though the number of concurrent clients is much greater that the maximum request rate handled in the previous experiments, the server operates practically only in the normal utilization range (between the underutilization and overutilization thresholds). This holds true until the system reaches the maximum number of clients that it can handle correctly (roughly 300 clients in the considered testbed). Thereafter, increasing the number of clients (concurrent requests) leads to a commensurate increase in the number of error responses.

The proposed model can be used to simulate all the interactions between the service components, using the extended SSFNet simulation tool, developed to predict the results of reconfiguration. This also applies in case of the modified client – server interactions under consideration.

The interaction model is based on the thresholds identified in Fig. 3. The client behaviour is modeled on the values that were actually used in the traffic generators. So, how do the simulation results bear out the response times observed in reality, as presented in Fig. 4?

This is shown in Fig. 5. The results are very accurate considering that we are approximating the complex behaviour of a software component with just a few

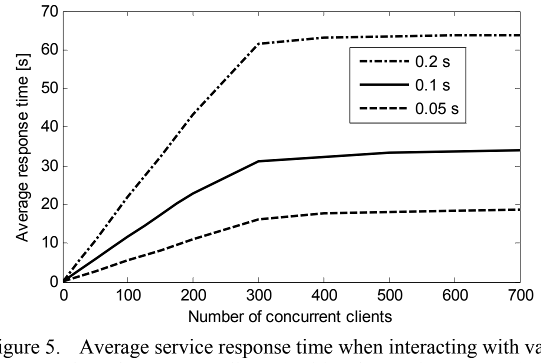


Figure 5. Average service response time when interacting with varied number of concurrent clients (waiting for service response)

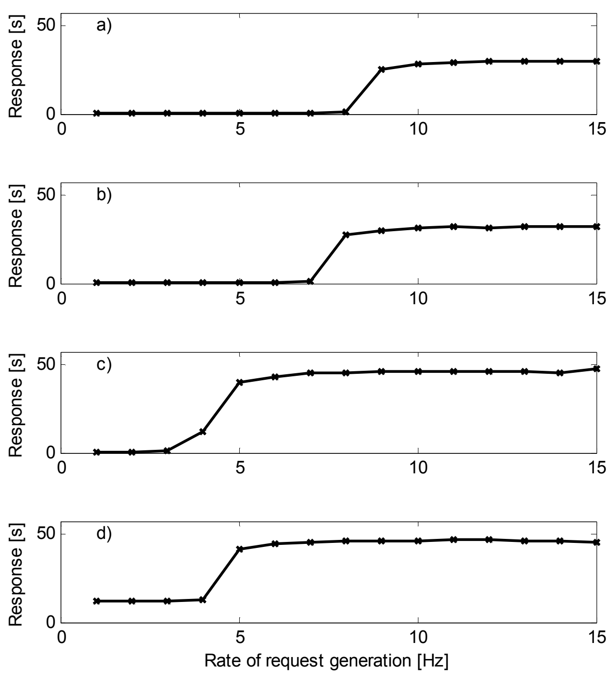


Figure 6. The system performance under varying rates of incoming client requests, with a co-located service: a) not accessed, b) accessed with 5 requests/sec., c) 10 req./sec., d) 15 req./sec

parameters. More to the point, the observed accuracy is fully satisfactory for the purpose of reconfiguration analysis. Of course, the accuracy is important only in the range of normal server operation. If the simulation indicates that the server is working much over its normal utilization range, then the availability will not meet the A_{min} requirement and should be flagged as unsatisfactory. Accuracy of response time prediction is then completely immaterial. For this reason, it is not really important that the error response time is characterized in the model with just a single parameter.

C. *Co-located service components*

In case of reconfiguration, service components are moved from one host (server) to another. At each location they share the computing resources with other co-located service components. Of course, as the components are relocated, their resource consumption models are also changed.

Fig. 6 presents the results of testbed experiments illustrating the changes in the model, caused by sharing the resources between co-located services. The model discussed so far, i.e., response times observed when the component does not compete for the processor with any other services, is presented for reference in Fig. 6a. The proportional changes in the model thresholds, caused by resource sharing, are illustrated in Fig. 6b and c (request rate from the second service was equal to 5 and 10 per second respectively). When the background service is overutilized, further increase in its loading does not increase the demand for processor. Thus, the model in Fig. 6d (request rate 15 per second) is not affected by it.

This proportional sharing is the basis of the implemented simulation model. In fact, the simulator is capable of taking into account prioritization between the components, but this has not been necessary in the conducted experiments.

D. *Choreography description*

The client-server interaction model has to consider the various tasks initiated by the client. In a typical web application, these tasks can exercise the server resources in a wildly varied manner: some will require serving of static web pages, some will require server-side computation, yet others will initiate database transactions or access to remote web applications. A common approach to traffic generation is based on determining the proportion of the various tasks in a typical server workload and then mixing the client models representing these tasks in the same proportion [10].

This approach assumes that the proportions of tasks in a workload do not change significantly due to response delays or reconfiguration. Traffic analysis can distinguish requests on the basis of client addresses, response times, size of requests and responses, connections and session identifiers. It does not yield any information on the semantics of client-server interactions, which should be the basis for determining the client models. It can be improved using the service choreography description, mentioned in Section II.

The web service is described by its choreography, using one of the formal languages. This description determines all the sequences of requests and responses performed in the various tasks, described in the choreography. Traffic analysis can then be employed to classify the observed request-response sequences to these business tasks. This procedure determines the typical proportion of the various business tasks in the workload that is much less affected by the service response times or reconfiguration. This workload is then used in the simulator to model the client behaviour.

VII. CONCLUSION AND FUTURE WORK

The model proposed in Section VI can be used to simulate all the interactions between the service components, to predict the results of reconfiguration. The performance of this simulator is currently under study and the results are very promising. All the presented results, besides using simulation, were validated against testbed system performance. It is still too early to conclude, though, whether these models are sufficiently accurate in general, considering the multiple server technologies that are on the market.

It should be noted that the proposed approach assumes that we have access to the running system. There is no clear method to predict the simulation model parameters prior to system implementation.

The most accurate results require knowledge of the stress test thresholds, obtained by active benchmarking of the deployed service components.

Adequate results are also obtained by observing the response times over various loads in passive monitoring (if the system requests rate is sufficiently varied). In this case the thresholds are derived using the best fit method.

ACKNOWLEDMENTS

The presented work was supported by the Polish National Science Centre under grant no. N N516 475940.

REFERENCES

[1] M. D. Aime, P. C. Pomi, and M. Vallini, “Policy-driven system configuration for dependability,” Proc. 1st International Workshop on Dependability and Security in Complex and Critical Information Systems DEPEND 2008, Cap Esterel, pp. 420-425, doi: 10.1109/SECURWARE.2008.54

[2] The Apache Software Foundation, “Apache JMeter,” vers. 2.7, http://jmeter.apache.org.

[3] A. Avizienis, J. C. Laprie, and B. Randell, “Fundamental Concepts of Dependability,” Technical Report vol. 1145, LAAS-CNRS, 2001.

[4] D. Caban, “Enhanced service reconfiguration to improve SOA systems depedability,” in Problems of dependability and modelling, Wrocław : Oficyna Wydawnicza Politechniki Wrocławskiej, 2011, pp. 27-39.

[5] D. Caban and W. Zamojski, “Dependability analysis of information systems with hierarchical reconfiguration of services,” Proc. 1st International Workshop on Dependability and Security in Complex and Critical Information Systems DEPEND 2008, Cap Esterel, pp. 350-355, doi: 10.1109/S.ECURWARE.2008.32.

[6] B. Delbosc, “Funkload documentation,” March 2011, http://funkload.nuxeo.org.

[7] R. Jain, The Art of Computer Systems Performance Analysis, Wiley and Sons, Inc., 1991.

[8] S. S. Lavenberg, “A perspective on queueing models of computer performance,” Performance Evaluation, vol. 10, Oct. 1989, pp. 53-76, doi: 10.1016/0166-5316(89)90005-9.

[9] R. Lent, “Evaluating a migration-based response to DoS attacks in a system of distributed auctions,” Computers & Security, vol. 31, is. 3, May 2012, pp. 327–343, doi: 10.1016/j.cose.2012.01.001.

[10] Ch. Lutteroth and G. Weber, “Modeling a Realistic Workload for Performance Testing,” Proc. 12th International IEEE Enterprise Distributed Object Computing Conference, 2008, pp. 149-158, doi: 10.1109/EDOC.2008.40.

[11] D. Nicol, J. Liu, M. Liljenstam, and Y. Guanhua, “*Simulation of large scale networks using SSF,”* Proc 2003 Winter Simulation Conference, vol. 1, 2003, pp. 650−657, doi: 10.1109/WSC.2003.1261480.

[12] J. Nielsen, Usability Engineering, Morgan Kaufmann: San Francisco 1994.

[13] J. Pasley, “How BPEL and SOA are changing Web services development”, IEEE Internet Computing Magazine, vol. 9, May-June 2005, pp. 60-67, doi: 10.1109/MIC.2005.56.

[14] P. Pérez and B. Bruyère, “DESEREC: Dependability and Security by Enhanced Reconfigurability,” European CIIP Newsletter, vol. 3, no. 1, 2007.

[15] H. Rahmawan and Y. S. Gondokaryono, “The simulation of static load balancing algorithms,” International Conference on Electrical Engineering and Informatics, 2009, pp. 640-645, doi: 10.1109/ICEEI.2009.5254739.

[16] G. F. Riley and T. R. Henderson, “The ns-3 network simulator,” Modeling and tools for network simulation (eds. K. Wehrle, M. Güneş and J. Gross), part 2, Springer, 2010, pp. 15-34, doi:10.1007/978-3-642-12331-3_2.

[17] K. Scarfone, T. Grance and K. Masone, Computer Security Incident Handling Guide, rev. 1, NIST Special Publication 800-61, 2008.

[18] S. Tozer, T. Brecht, A. Aboulnaga, “Q-Cop: Avoiding bad query mixes to minimize client timeouts under heavy loads,” Proc. 26 International Conf. on Data Engineering ICDE’10, 2010, pp. 397-408.

[19] B. Urgaonkar, G. Pacificiy, P. Shenoy, M. Spreitzery, and A. Tantawi, “An analytical model for multitier internet services and its applications,” Proc. of the ACM SIGMETRICS'05 Conference on measurement and modeling of computer systems, 2005, pp. 291-302, doi: 10.1145/1064212.1064252.

[20] A. Varga and R. Hornig, “An overview of the OMNeT++ simulation environment,” Proc. 1st International Conference on Simulation tools Simutools’08, ICST, Brussels, 2008, pp. 1-10, doi: 10.1145/1416222.1416290.

[21] T. Walkowiak, “Information systems performance analysis using task-level simulator,” Proc. 4th DepCoS – RELCOMEX, IEEE Press, 2009, pp. 218−225, doi: 10.1109/DepCoS-RELCOMEX.2009.49.

[22] T. Walkowiak and K. Michalska, “Functional based reliability analysis of Web based information systems,” Dependable computer systems, vol. 97, Springer: Berlin-Heidelberg, 2011, pp. 257–269, doi: 10.1007/978-3-642-21393-9_20.

Accurate Retargetable Decompilation Using Additional Debugging Information

Jakub Křoustek, Peter Matula, Jaromír Končický, Dušan Kolář
Faculty of Information Technology, IT4Innovations Centre of Excellence,
Brno University of Technology, Božetěchova 1/2, 612 66 Brno, Czech Republic
Email: {ikroustek, kolar}@fit.vutbr.cz, {xmatul01, xkonci01}@stud.fit.vutbr.cz

***Abstract*—In this paper, we present an extension of an existing automatically generated retargetable decompiler that is capable to parse, process, and utilize compiler-generated debugging information. This tool can be used for dealing with several security-related issues (e.g., forensics, malware analysis, vulnerability detection). Additional debugging information is used for an accurate reconstruction of platform-dependent binary applications into a well-readable high-level-language representation. The proposed solution is platform and debugging-format independent. In present, two major debugging formats—DWARF and Microsoft PDB—are supported; the extracted information is used for a recovery of several high-level constructions (e.g., variables, functions and their arguments). The proposed concept was validated by experimental results.**

***Keywords-decompilation*; *debugging information*; *PDB*; *DWARF*; *Lissom*.**

I. Introduction

Reverse engineering is an old method used for analysis and reconstruction of a given object. In information technology, a lot of reverse-engineering tools have been created for understanding and reconstruction of binary software, e.g., disassemblers, dumpers, and sniffers. However, the result of mentioned tools is usually a very low-level representation of the input object, e.g., assembler code, raw dump of memory, or data stream. Therefore, it is hard to understand its meaning. The reconstruction of the original object is not automatic and it requires manual re-implementation based on the gathered information.

A machine-code decompiler (i.e., reverse compiler) is a more advanced reverse-engineering tool. It can theoretically produce the most readable and re-compilable code; it transforms the input binary executable application into a particular high-level-language (HLL) representation. In software maintenance, this process can be used for source code recovery, translation of code written in an obsolete language into a newer language (see [1] for such an application), migration of binary code to a new platform, injection of additional features into an existing application, etc.

However, decompilation is typically used in the field of computer security and forensics. Within this field, decompilation is used for compiler verification (i.e., verification of code and debugging information generated by compiler in software-critical systems, since the compiler cannot be trusted in these systems [2]), vulnerabilities detection, and for malicious code analysis and its understanding.

In comparison, disassemblers have been used for malware analysis in the past decades, but decompilers are used more and more often last years. Their primary advantage over disassemblers is a more readable output (i.e., HLL code) that is independent on a particular processor architecture. Moreover, it is possible to precisely analyse code without a deep knowledge of underlying architecture (e.g., processor resources, instruction set, micro-architecture). This is very useful because of a massive expansion of malware to the new platforms (e.g., smartphones, tablets, or even cars [3]). Therefore, for a malware analyst, it is not necessary to learn all the details about these platforms.

However, the machine-code decompilation is very complicated task because the input applications are often obfuscated by packers or heavily optimized by compilers. Therefore, it is necessary to take advantage of every available information. The decompiled code must meet two essential criteria—it has to be functionally equivalent to the input binary code, and it has to be highly readable. It is a great advantage if the decompiled code is re-compilable too.

In this paper, we present a concept of a debugging-information exploitation within a decompilation process. The proposed solution allows extracting additional information from two major debugging formats—DWARF and Microsoft PDB. It is possible to find a lot of valuable information within these two formats, such as lists of original source files, line numbers, functions, or variable names and their types.

Such information is used for the recovery of several high-level constructions (e.g., functions together with their arguments and local variables) in an existing automatically generated retargetable decompiler developed within the Lissom project [4]. This decompiler is independent on a particular target architecture, operating system, or origin of the decompiled application (i.e., the used programming language and its compiler). This tool can be used in every previously mentioned area.

The paper is organized as follows. Section II discusses existing debugging information formats. Then, we briefly describe the retargetable decompiler developed within the Lissom project in Section III. The exploitation of debugging information within the decompiler is then presented in Section IV. Section V describes the state of the art of debugging information exploitation in decompilation. Experimental results are given in Section VI. Section VII closes the paper by discussing future research.

II. Debugging Information Formats

Despite all programmers efforts, any meaningful program contains bugs or defects that need to be found and fixed. This process is called debugging and it can be done in several different ways (e.g., code insertion, stepping through instructions, displaying registers or memory). The goal of today's debuggers is to provide source-level approach, which matches lines of source code with machine-code instructions. For this purpose, debugged executables or libraries must contain additional information to identify the corresponding positions, variables, functions and other useful details in the form of debugging data format. This section introduces two of today's the most common formats.

A. DWARF (Debugging With Attributed Record Formats)

DWARF [5] is a debugging data format originally developed in the mid-1980s at Bell Labs for Unix System V Release 4. Even though it is mainly associated with ELF [6], it is independent on the used object file format, programing language, operating system, or target architecture (e.g., MIPS, ARM, Intel x86). The latest, fourth version was published in 2010, but its second version (DWARFv2) is still the most commonly used one among compilers and debuggers.

DWARF information is stored in special sections in the same object file as machine code. Each of these sections contains different kind of debugging data, such as basic file information, line numbers, look-up tables, call frames, and macros. The whole program is represented as a tree, whose nodes can have children or siblings. Each node is a *Debugging Information Entry* (DIE) structure that has a tag and a list of attributes. A tag specifies the type of the DIE (e.g., function, data type, variable) and attributes contain all the description details. There are two general types of DIEs:

- Those describing data and types such as base types, variables, arrays, structures, classes, etc. Locations of data are defined by location expressions, consisting of a sequence of operations and values, evaluated by a simple stack machine.
- Those describing executable code. Functions (they have a return value) and subprograms are treated as two flavours of the same construction. These DIEs typically own many other DIEs that describe them (e.g., parameters, local variables, lexical blocks).

There is also a special type of DIE called compilation unit which is the parent for all DIEs created by a separate compilation of a single source file. It holds information about compilation (e.g., name of the source, language, producer) and locations of other DWARF data not described by DIEs (e.g., line number table, macro information).

DWARF is generated by the most major compilers (e.g., `gcc`, LLVM `llc`, Intel `icc`), supported by nearly all debuggers, and there are several libraries and utilities that allow its examination or manipulation (e.g., `libdwarf`, `dwarfdump`, `objdump`).

B. PDB (Program Database)

PDB is a format developed by Microsoft Corporation. It is generated only by Microsoft Visual Studio compilers and used by Microsoft debuggers. PDB is mainly used on the ARM and Intel x86 architectures and it is based on the older CodeView format. PDB debugging information for a particular application is stored in a separate file with the `pdb` extension. Each executable binary file in a PE format [7] has its counterpart in a PDB file. Both files are paired by a unique GUID code.

PDB is a proprietary format, and its specification has never been publicly published. Therefore, the analysis of this format can be only done via reverse engineering. A PDB file structure is similar to a file system. A PDB file consists of many streams (i.e., sub-files), each of them contains a different kind of information. There is a stream with type information (e.g., common types, enumeration, structures), compilation units (i.e., modules), symbol tables and PE sections. Furthermore, each module has its own stream with information about functions, local and global variables, and line number information within the module.

The official DIA SDK is provided for processing debugging information [8]. However, this toolkit is platform dependent because it can be used only under the MS Windows operating system.

C. Other Formats

Except the two aforementioned formats, there are some other formats that can be used to store debugging information. The best known are *Symbol Table Strings* (STABS), where data are stored as special entries in symbol table, and *Common Object File Format* (COFF) [9], that could contain debugging information in special sections much like DWARF. Both were strongly tied to specific object file formats and became obsolete along with them.

III. Lissom Project's Retargetable Decompiler

In this section, a brief description of an existing automatically generated machine-code retargetable decompiler is presented. This tool is developed within the Lissom project at Brno University of Technology [4]. It is supposed to be independent on a particular target architecture, operating system, or executable file format. See [10] for its detailed

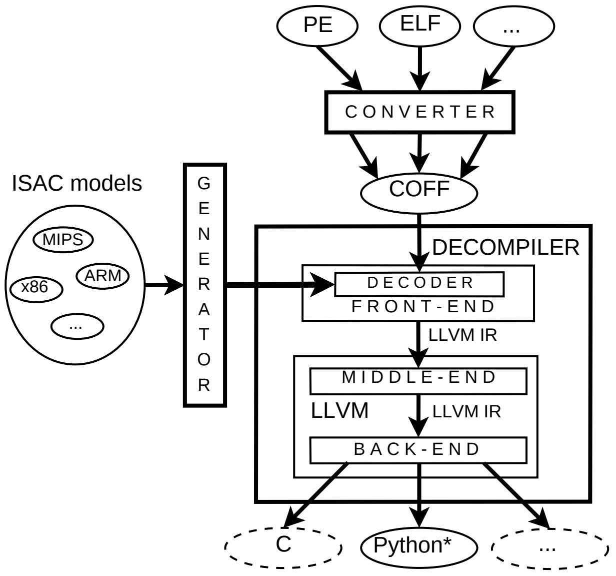


Figure 1. The concept of the Lissom project's retargetable decompiler.

description. The decompilation process consists of four phases, see Figure 1.

(1) At first, the input binary executable file is transformed using a *binary file converter* from a platform-specific executable format (e.g., Windows PE, Unix ELF) into an internal, uniform COFF-based file format, see [11] for details.

(2) The subsequent part of the decompiler is a *front-end*, which is its only platform-specific part because of the instruction decoder that is automatically generated based on the target architecture model (e.g., MIPS, ARM, Intel x86). The architecture description language ISAC [12], developed also within the Lissom project, is used for this purpose. The front-end analyzes and transforms the input application (i.e., its machine code, data and other information) into the LLVM assembly language code, LLVM IR [13], which is used as an internal code representation of decompiled applications in the remaining decompilation phases.

(3) Afterwards, this program representation is optimized in a *middle-end* using many optimization passes.

(4) Finally, the program intermediate representation is emitted as the target HLL in a *back-end*. Currently, a Python-like language is used for this purpose, while a support of other target languages is under development (e.g., the C language). Both middle-end and back-end are built on the top of the LLVM Compiler System [14]. This language is very similar to Python, except a few differences (e.g., we use several C-like constructs, address and dereference operators).

IV. Debugging Information Exploitation Within The Retargetable Decompiler

In this section, the concept of the DWARF and PDB information parsing, processing, and utilization within the Lissom project's retargetable decompiler is discussed.

The debugging information is parsed by two different parsers, the first for DWARF and the second for PDB. Afterwards, the parsed information is processed and stored in a debugging-format-independent way. This information is utilized for an enhancement of the LLVM IR code representation within the front-end part of the decompiler; see Figure 2 for details. Finally, this code representation is passed to the subsequent decompiler parts, see Section III for details.

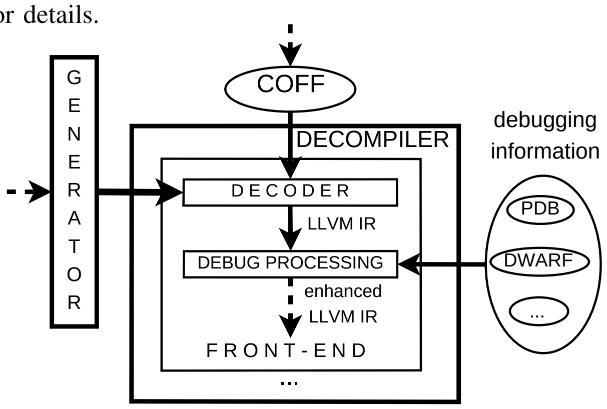


Figure 2. Debugging information processing within the front-end.

A. DWARF Parsing

Parsing of debugging information from an object file to structures defined in the DWARF specification is done by the `libdwarf` [15] library. Original `libdwarf` is using `libelf` to access sections of interest in ELF binaries. However, the input of our decompiler are uniform COFF-based objects. In order to parse information from such files, we exploited `libdwarf` object access capabilities and provided a new interface using our internal object manipulation library.

`libdwarf` creates a low-level representation of debugging data, whose usage in a decompiler or debugger would be very complicated and unintuitive. That is why we decided to create a new, mid-layer library called `dwarfAPI`, that builds high-level, object-oriented data structures and provides convenient access methods. `dwarfAPI` represents all debugging information about a binary file as one entity consisting of several vectors of objects such as compilation units, lines, types, functions and global variables. Each of these objects contains complete information about itself and about all the other objects they own (e.g., local variables in functions). This way, the information that was in a low-level representation scattered among multiple DIEs is grouped together, what allows very natural and easy way of processing complex DWARF structures. It should be mentioned that `dwarfAPI` is not limited to a particular DWARF version or target architecture.

B. PDB Parsing

The only available official toolkit for PDB parsing—DIA SDK—is unusable for our aims because of the limitations discussed in Section II. Therefore, our own PDB parser has been created. At first, it was necessary to reverse-engineer the PDB format and analyze its internal structures. For this purpose, we reused an existing unofficial PDB analyzer [16].

The PDB file parsing consists of two steps. (1) At first, the streams have to be extracted and separated. They are divided into constant-size data blocks. At the end of the PDB file, there is a root directory, which stores each stream's size and the indexes of used data blocks. (2) The main stream processing is done in a consequent step. Most of the streams are organized into *symbols*, which are data structures with a type, size, and data.

While processing all the symbols, the parser fills the internal data structures. For example, we can extract and process an address, length, return type, arguments, local variables and a line number information (i.e., mapping between machine-code and HLL code location) for each function described in the PDB file.

Meaning of many data entries depends on a context (i.e., the previous data entries). The information stored in a particular stream is often interconnected with another stream. For example, each data type has an index to its definition—base types (e.g., `int`, `float`) have a predefined index, but the user types are defined in a type information stream.

In present, we are able to extract and utilize most of the PDB streams, but a few remaining streams are still unknown for us. As well as the DWARF parser, the PDB parser is not limited to a particular target architecture (e.g., ARM, Intel x86).

C. Decompiler

After the debugging information is extracted by a parser, it is ready to be used in the decompiler's front-end, which updates its intermediate code representation (LLVM IR) based on this extracted information. In the following paragraphs, we depict the useful information and its usage within the front-end.

Module information: Based on the type of the original HLL, the decompiled binary application is created from one or more *modules* (e.g., source files). The debugging information is usually divided into smaller pieces, where each piece corresponds to one particular module. This is useful for the decompiler because it is possible to divide the resulting code into similar modules, e.g., the decompiler can divide the decompiled code into several files with the original names.

Function information: Probably the most important debugging information is related to functions. A proper function detection and recovery is a crucial task of each decompiler. Otherwise, the unstructured and hardly readable ("spaghetti") code will be generated (e.g., code containing `goto` statements). The Lissom project's retargetable decompiler utilizes its own function detection heuristics based on the principles described in [17], [18], [19]. However, the function's debugging information is used instead whenever it is available because it is essentially more precise. For example, it is possible to obtain the function location within the machine code, its name, the number of its arguments, their names and types, and many others.

Variables: Our retargetable decompiler is able to detect both global and local variables, and guess their types. In the resulting code, it generates variable names from a fixed list of well-readable names, see [10] for details. However, the decompiler emits the original name whenever it is stored in the debugging data. This is just a detail, but it gives the feeling of listing the original code. Furthermore, both DWARF and PDB support storage of variable types, too. This is handy because the recovery of composite data types is a non-trivial task, see [20].

Line Information: Mapping of source-code locations to the machine-code is done via so-called *line information* and it is an essential part of each debugger. However, its usage within decompilation is not so important. It can be only used for re-formatting the decompiled code to better fit its original format (e.g., splitting the complex expressions into more statements).

In present, the decompiler uses the function and variable debugging information, while exploitation of other information is under development.

V. Related Work

We can find two existing decompilers that exploit debugging information. Both of them support more target architectures; however, none of them is truly retargetable because the support of such architectures is hand-coded by their authors.

The Hex-Rays Decompiler [21] supports decompilation of the ARM and Intel x86 target architectures. It allows using PDB debugging information for enhancing the generated C code. However, this proprietary software probably uses the Microsoft DIA SDK [8] or Microsoft Debugging Tools for Windows (see their limitations in Section II) for PDB parsing and processing. Therefore, this feature is only available in the MS Windows version of this software. The Unix version of Hex-Rays Decompiler needs a remote MS Windows server with this library for gathering PDB information. This decompiler can exploit the PDB debugging information for reconstruction of functions, their arguments, global variables, etc. However, recovery of local variables is not always correct (details about this problem are described in Section VI).

The REC decompiler [22] has a more extensive list of supported target architectures (e.g., MIPS, PowerPC). According to the official homepage [22], the DWARFv2 support is in an early stage of development. However, it is supported only for a subset of the supported architectures. Therefore, it is plausible that their DWARF parser is also not retargetable and it needs to be manually re-configured for each given architecture. Finally, the PDB support is planned as well.

As can be seen, the debugging information is a valuable resource of additional information and it is already exploited by existing decompilers. However, their current implementation is limited for several reasons.

VI. Experimental Results

To demonstrate several selected abilities and advantages of our solution, this section presents a decompilation of a simple program calculating factorial function for the ARMv4 architecture. The C source code for this program is given in Figure 3. It was compiled using Microsoft Visual Studio 2005 compiler with enabled PDB debugging-information (`/Zi`) and disabled optimizations (`/Od`).

```
#include <stdio.h>

volatile int value = 6;

int fact(int n)
{
   if (n==0)
      return(1);
   else
      return n*fact(n-1);
}

int main(int argc, char *argv[])
{
    int a = fact(value);
    printf("%d\n", a);
    return a;
}
```

Figure 3. Source code in C.

The size of the generated PDB file is 1.07MB. Its processing by our parser is almost instant (approximately one second on Intel Core i5 with 3.3 GHz, 8GB RAM running Windows 7 64-bit operating system). The resulting HLL code generated by our decompiler can be seen in Figure 4. Observe the following key aspects of our solution.

(1) With the use of debugging information, we are able to completely recover functions, i.e., their names, arguments, and their mapping to a machine code (e.g., `fact`, `main`). Our other algorithms of functions, arguments, and return values detection that do not use debugging information (briefly mentioned in Section IV) achieve 80-90% accuracy of detection. However, the detection method based on debugging information is absolutely precise as long as a compiler generates the correct debugging information.

(2) Furthermore, the recovery of global and local variable names is also possible in most cases (e.g., `value`, `a`, `n`). The only limitation is for local variables because LLVM optimizations, used in the decompiler's middle-end, sometimes remove them in the resulting code. On the other hand, the optimizations can produce variables that were not in the original code; therefore, they are missing in the debugging information. In such situations, the decompiler assigns more appropriate names to such variables than just their addresses.

```
# -------- Global Variables --------
value = 6

# -------- User Functions ----------
def fact(n):
    if n == 0:
        return 1
    return n * fact(n - 1)

# -------- Main Function -----------
def main(argc, argv):
    a = fact(value)
    printf("%d\n", a)
    return a
```

Figure 4. Decompilation result in the Python-like language.

In the real world, the debugging information is not always available for the decompiled binaries. Its presence strongly depends on a type of decompiled binary application and purpose of its decompilation. In a case of source code recovery, binary code migration, or vulnerability detection, there is a very good chance that the debugging information is present because applications are usually not stripped, obfuscated or protected. Presence of debugging information within the decompiled application is guaranteed in the case of compiler verification.

On the other hand, malware rarely contains such additional information. The main reason is that this harmful code is stripped after creation (i.e., the unnecessary information is removed). Furthermore, the application is often obfuscated, ciphered, and packed by some of the existing packing or protecting software. Based on our internal malware database, only a few percents of malware contain debugging or symbolic information. However, as has been said in the introduction, the decompilation is so much complicated process that every possible additional information must be exploited.

Finally, it should be noted that decompilation of compiler-optimized application can produce a more readable code than decompilation of a non-optimized application in some cases. For example, it is easier to track values in registers (i.e., optimizations enabled) than in memory (i.e., optimizations disabled) for the decompiler. However, this is possible only for several compilers because debugging-information generation sometimes disables the most aggressive optimizations.

VII. Conclusion and Future Work

This paper has proposed an extension of the existing retargetable decompiler which produces a more accurate decompiled code. The presented solution is based on the exploitation of compiler-generated debugging information whenever it is available. The available debugging information is utilized for a precise recovery of several HLL constructions and enhancement of code readability. In present, two common debugging formats are supported—DWARF and Microsoft PDB.

The fundamentals of debugging information parsing, processing, and utilization were briefly discussed and we have presented results of the current state of the implementation. As can be seen, we are already able to use debugging information for the recovery of several HLL constructions (e.g., functions, their arguments, variables). The resulting code is a highly readable code in a Python-like language that can be used in many areas related to software maintenance and security.

However, there is still a lot of space for improvements. In the first place, it is necessary to finish the reverse-engineering analysis of the remaining PDB streams. Moreover, the decompiler does not use several extracted information (e.g., source-code line information, information about modules) at the moment. After that, we will be able to produce a more accurate LLVM IR code, which will improve the resulting HLL code.

Acknowledgment

This work was supported by the project TA ČR TA01010667 System for Support of Platform Independent Malware Analysis in Executable Files, BUT FIT grant FIT-S-11-2, by the project CEZ MSM0021630528 Security-Oriented Research in Information Technology, and by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

References

[1] L. Ďurfina, J. Křoustek, and P. Zemek, "Generic source code migration using decompilation," in *10th Annual Industrial Simulation Conference (ISC'2012)*. EUROSIS, 2012, pp. 38–42.

[2] C. Cifuentes, "Reverse compilation techniques," Ph.D. dissertation, School of Computing Science, Queensland University of Technology, Brisbane, AU-QLD, 1994.

[3] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, and T. Kohno, "Comprehensive experimental analyses of automotive attack surfaces," in *20th USENIX conference on Security (SEC'11)*. USENIX Association, 2011.

[4] Lissom, http://www.fit.vutbr.cz/research/groups/lissom/, [retrieved: June, 2012].

[5] *DWARF Debugging Information Format*, 4th ed., DWARF Debugging Information Committee, 2010.

[6] TIS Committee, "Tool Interface Standard (TIS) Executable and Linking Format (ELF) Specification," 1995.

[7] Microsoft Corporation, "Microsoft portable executable and common object file format specification," http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx, [retrieved: June, 2012], version 8.2.

[8] ——, "Debug interface access SDK," http://msdn.microsoft.com/en-us/library/ee8x173s.aspx, [retrieved: June, 2012].

[9] G. R. Gircys, *Understanding and Using COFF*. Sebastopol, CA, USA: O'Reilly & Associates, Inc., 1988.

[10] L. Ďurfina, J. Křoustek, P. Zemek, D. Kolář, K. Masařík, T. Hruška, and A. Meduna, "Design of a retargetable decompiler for a static platform-independent malware analysis," *International Journal of Security and Its Applications*, vol. 5, no. 4, pp. 91–106, 2011.

[11] J. Křoustek, P. Matula, and L. Ďurfina, "Generic plugin-based convertor of executable file formats and its usage in retargetable decompilation," in *6th International Scientific and Technical Conference (CSIT'2011)*. Lviv Polytechnic National University, 2011, pp. 127–130.

[12] K. Masařík, *System for Hardware-Software Co-Design*, 1st ed. Brno, CZ: Faculty of Information Technology BUT, 2008.

[13] LLVM Language Reference Manual, http://llvm.org/docs/LangRef.html, [retrieved: June, 2012].

[14] The LLVM Compiler System, http://llvm.org/, [retrieved: June, 2012].

[15] D. Anderson, "Libdwarf and dwarfdump," http://reality.sgiweb.org/davea/dwarf.html, [retrieved: June, 2012].

[16] "Parsing library for PDB file format," http://code.google.com/p/pdbparser, [retrieved: June, 2012].

[17] D. Kästner and S. Wilhelm, "Generic control flow reconstruction from assembly code," *ACM SIGPLAN Notices*, vol. 37, no. 7, 2002.

[18] J. Kinder, F. Zuleger, and H. Veith, "An abstract interpretation-based framework for control flow reconstruction from binaries," in *10th International Conference on Verification, Model Checking, and Abstract Interpretation (VMCAI'09)*. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 214–228.

[19] N. Bermudo, A. Krall, and N. Horspool, "Control flow graph reconstruction for assembly language programs with delayed instructions," in *5th IEEE International Workshop on Source Code Analysis and Manipulation SCAM'05*, 2005, pp. 107–116.

[20] K. Troshina, Y. Derevenets, and A. Chernov, "Reconstruction of composite types for decompilation," in *10th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM'10)*. IEEE Computer Society, 2010, pp. 179–188.

[21] Hex-Rays Decompiler, www.hex-rays.com/products/decompiler/, [retrieved: June, 2012].

[22] Reverse Engineering Compiler (REC), http://www.backerstreet.com/rec/rec.htm, [retrieved: June, 2012].

An Architecture Based on Agent-manager Model for Automated Data Collection of Security Metrics

Liniquer Kavrokov Vieira, Rodrigo Sanches Miani, Bruno Bogaz Zarpelão, Gean Breda, Leonardo Mendes
School of Electrical and Computer Engineering
University of Campinas (UNICAMP)
Campinas – SP - Brazil
{liniquer, rsmiani, bzarpe, gean, lmendes} @decom.fee.unicamp.br

Abstract— **The requirement of organizations on computer network makes information security a key element to the evolution and continuity of services in our society. Security metrics are developed in order to offer a quantitative and objective basis for security assurance. This study proposes an architecture based on the agent-manager management model to allow the automated data collection from several components in a computer network, aiming to expand the security metrics application. A tool for measurement and automated data collection of metrics based on the architecture proposed were developed and applied in a real computer environment. Tests were performed showing that the architecture proposed is able to integrate information control and support the security monitoring process.**

Keywords-computer network; security metrics; security management; automated data collection.

I. INTRODUCTION

Current changes in computer networks have led to a paradigm shift. The network transmission of voice, data and video has been converging into a single multi-service network platform based on IP (Internet Protocol) [1]. This multi-service network can provide access to applications and content such VoIP (Voice Over Internet Protocol), video over demand and IPTV (Internet Protocol Television). Therefore, triple play services can be provided on a single connection, making customers dependent on the reliability and availability of these networks.

Despite the benefits that an integrated computer network can provide, several incidents and issues related to information security may exist. Some examples are the absence of security protocols in access points, unauthorized access to an information system, computer security vulnerabilities and exploits or misconfiguration of firewalls in computer networks [2].

These incidents may allow possible attacks and also compromise the whole infrastructure of an organization. In large-scale computing networks, identifying failures become a complex task for network administrators. A complex scenario composed by equipments and software provided by different vendors, requires the reduction of human intervention to prevent errors for the process of large amount of data.

Considering that preventive measures may also fail, it is important to adopt tools that might be used to avoid security incidents and prevent intrusions for a rapid response and supporting post-event activities [3].

Important organizations in security field such as SANS (SysAdmin, Audit, Network, Security) and NIST (National Institute of Standards and Technology) recommend the use of metrics in enterprises. Security metrics provide a quantitative approach to measure and analyze security controls and effectiveness at the security management [4]. Due to the lack of consensus on the term definition, most authors relate the concept of security metrics to quantifying and analyzing specific security data [5].

Nevertheless, using metrics to measure security levels usually requires a high cost-time for data collection. The, development of tools to automate the metrics collection process might be an efficient way to measure security in complex and heterogeneous computer network. Moreover, it contributes to decrease the data time collection and also to support the active security monitoring process [3].

This paper proposes an architecture based on a manager-agent management model for data collection from different components in IP-based computer networks, aiming to expand security metrics application. The proposed architecture is also able to integrate information control and support the security monitoring process.

The main motivations of this work are:

- The risk associated to security incidents and the importance of improving the security management;
- The lack of integrated tools to support the collection and analysis of security data in heterogeneous networks;
- High cost to quantify information security data, considering the time that network administrators could spend to manually collect and analyze information security data;

This paper is organized as follows. Section 2 presents some related work in the field. Section 3 presents security metrics and automated data collection. We introduce in Section 4 the criteria to select security metrics. Session 5 describes the architecture proposed in this paper. Next, Section 6 presents a case study conducted in a real computer network and discusses some experimental results. Finally, Section 7 offers the conclusion and poses future research questions.

II. Related Work

Security metrics appear as an option to measure and assess information security issues and also to provide a feedback from organizations network [4].

Security metrics development and its application are discussed in [6] and [7]. Jaquith [4] states that any quantification problem that results in a numeric value can be seen as a metric. Its application can provide many benefits to information security management such as the evaluation of security risks, alerting to impending troubles, understanding the security infrastructure weaknesses and encouraging the use of technology.

Miani [5] proposes a methodology to facilitate the process of data analysis obtained with metrics application. The objective is to measure structural problems and vulnerabilities of network services. In Miani's work, metrics are collected in a real communication computer network. One of the difficulties found was the absence of a specific tool to help the data collection. Two following requirements were cited in this paper: recording security data in a specific database and presenting statistical data.

Ertürk [3] proposes a framework for security monitoring based on security metrics, which aim to examine the information security levels continuously over time. This requires collecting, processing cumulative data and reporting results according to organization requests. The author uses several monitoring tools to collect metrics, such as PRTG Network Monitor, Nessus Vulnerability Scanner, and others, which makes control and data integration more difficult.

Network management systems that enable the monitoring of network communication systems are found in [1] and [8]. Lee [1] presents a viable platform for network management based on architecture model TMN (Telecommunications Management Network) which uses JAVA technology and enables the control of functions such as, sending and receiving requests from network components, alarms and data performance. In this case, it is possible to collect automated data, but this is not the purpose of author.

This paper seeks to solve these difficulties proposing an integrated solution for automated data collection of metrics. It aims at supporting the monitoring of several components and improving the integration of data collected from different sources in IP-based computer networks.

III. Security Metrics and Automated Data Collection

Security metrics are developed aiming to help the assessment of security processes and controls developed by organizations. It also might be used to measure cost, effectiveness and improvements controls of organizations [4].

Security data in heterogeneous networks can be collected using basically two methods: manually or through automated processes. When performed manually, it can require a higher time and effort for collection. Whereas in automated processes, it is necessary to use complex routines of tools like protocol analyzers and in-depth analysis of audit logs [5].

Procedures for data collection on security metrics can be automated using a management platform capable of extracting data from different locations in a computer network. According to Lee [1], a management platform can enable the integration of several resources, such as the creation of graphical user interface, distributed process, networking facilities and object orientated paradigms. It can be implemented, for example, by using object oriented concepts and management API's supported by JAVA language programming [1].

However, when using several management platforms to collect metrics, it makes the control and data integration become more difficult. This paper proposes a solution for automated data collection from different components in a computer network using an integrated platform in order to helps the network administrator to have comprehensive and data integrated control from security information.

Therefore, the development of an integrated platform for security data collection can provide several benefits, such as [4]: increased accuracy in data collection, repeatability, high frequency of measurement, reliability, transparency and auditability.

IV. Metrics Selection

CIS (Center for Internet Security) [6] is a nonprofit organization whose goal is to improve information security of private and public sectors in general. One of its works has been to develop a guide which helps organizations selecting some feasible security metrics. This guide primarily states that in order to implement a metrics program, one must select a set of metrics that satisfies the interests of organizations and supports security managers in decisions making [6].

Inappropriate metrics application or poorly planned metrics can provide a false sense of security and loss of credibility. However, there is no consensus about attributes which makes good metric. Jaquith [4] proposed the following set of desirable features for security metrics: i) Defined consistently, without subjective criteria; ii) Easy to collect, preferably in an automated way; iii) Expressed as a percentage or cardinal numbers, not to qualitatively labeled as "high", "medium" and "low"; iv) Expressed by using at least one measure unit, such as "defects", "hours" or "dollars"; v) Significant enough so that decisions can be taken based on metrics results;

Besides having a set of metrics with the features mentioned above, it is necessary to have a plan and a preliminary evaluation of which data should be gathered. Some organizations collect more data than necessary, whether assuming that it is always better to have extra data, or because it is easier to collect a large amount of data, and then determine which will be used for developing a security metric implementation plan.

In order to facilitate the process of metrics development, security frameworks such as NIST [2], COBIT [10] and ISO/IEC [11] have been proposed taxonomies to classify metrics. Savola [12] considerates the following levels when classifying metrics: security metrics for cost-benefit analysis; trust metrics for risk analysis in terms of business; security metrics for managing information and security metrics for products, systems and services (SDT - Security, Dependability and Trust).

Another point to be considered is whether the metric can be automated or not. Jaquith [4] classifies as technical metrics that allow the identification and diagnosis of security problems across the infrastructure (physical and logical) of an organization and usually do not require human intervention to collect data. A comprehensive taxonomy is also suggested in [2] based on management, technical and organizational classification. In this work we focused on this kind of metrics.

One of the contributions of this work is the creation of criteria to select metrics that can be automated. Automated metrics can be seen as metrics whose data can be collected on servers, desktops and network equipments using protocols grouped by layers of network.

The criteria are classified according to the protocols from OSI model, which might be used to read or collect data from metrics, for example, the protocols from: i) application layer, such as, SNMP, NetFlow, IPFIX and HTTP; ii) network layer, for example, ICMP; iii) transport layer, such as TCP and UDP. These criteria are based on TCP/IP related polling techniques, because TPC/IP is the point of convergence of different services in next generation multi-service networks. It is also important to mention that new criteria for metrics selection can be added as needed, take into consideration the network layer protocol used to collect the data.

The main goal of the criteria is to assist the metrics selection process prioritizing the metrics that could be gathered in an automated way Thus, many benefits can be provided, for example, helping decision making, shortening the time of metrics selection and ensuring greater confidence during the selection process and automation.

Metrics that do not meet all criteria presented previously or require manual intervention are also important and can be implemented by the organization. However, they are out of scope of this work.

V. DATA COLLECTION ARCHITECTURE

After selecting a specific set of metrics based on the automation criteria, it is necessary to develop an architecture that helps the implementation of a security metrics program. This paper proposes an architecture based on an agent-manager management model that is well-known in the network management field through a computer configured as a manager and other components playing the role of agents.

Agent-Manager management model is basically composed by three components [9]: a network management station, that monitors managed resources and adjust them according to the network administrators interests; a set of managed objects that correspond to an agent (switches, servers, routers) and to their associated data; and a management communication protocol used by the station manager and by agents to exchange data.

Figure 1 illustrates an agent-manager structure in a computer network. In this example, a computer is used as a management station that sends data requests to other components via SNMP (Simple Network Management Protocol). In the SNMP architecture, there is a database of managed objects called MIB (Management Information Base) that contains data about the managed device. This data is sent to the manager. In doing so, the management station might be able to execute metrics calculation routines using such collected data, providing useful information to network administrators.

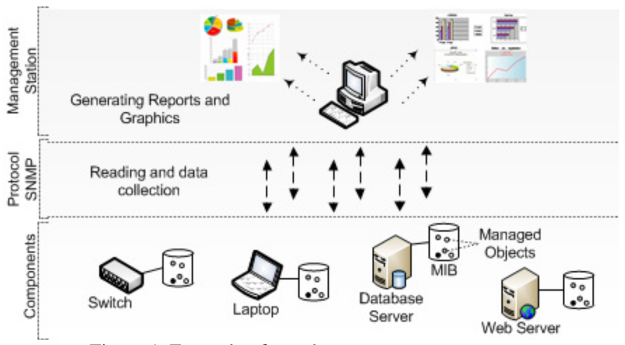


Figure 1. Example of a real management structure

It is possible to note in Figure 1 that only the SNMP protocol is used to exchange data between the management station and the components. Furthermore, several monitoring tools can be used to collect metrics and generate reports, which make control and data integration more difficult.

Whatever, it is important to collect data from several sources using different protocols and only one tool. Then, the architecture proposed in our work is also composed by three elements: i) monitored computer network, where the data to be collected is located; ii) communication protocols for data collection; iii) management station which collect data, transform data into metrics and generate reports and graphics.

Figure 2 shows an overview of the proposed architecture. In this example, different protocols can be used to exchange data between the management station and components. Moreover, a tool for measurement and automated data collection is used to integrate information control and support the security monitoring process.

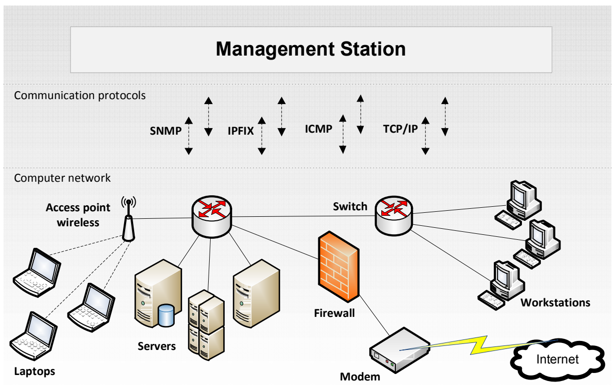


Figure 2. Overview of proposed architecture.

The following subsections present a detailed description of each component, their interactions and how they can help with the improvement of security metrics application and collaborating with data integration.

A. *Computer Network*

The architecture defined in this paper aims to support the monitoring of several components establishing a computer network. Some examples of such components are: workstations, switches, wireless routers, servers, and laptops.

This set of heterogeneous components has a vast amount of data that might be important to organization's security management. Most of this components support management protocols and have a management database which is accessible and manageable.

Therefore, this data gathering can be automated through an integrated solution contributing to the network management processes and supporting the evaluation of network security level. Next subsection introduces some protocols that may be used to collect security data.

B. *Communication Protocols*

The TCP/IP (Transport Control Protocol/Internet Protocol) is a collection of protocols that can be used to perform the communication in a network. Socket [13] is an interface that provides a communication mechanism to computer applications using TCP or UDP protocol. Therefore, applications can be implemented through socket to read network data such as which servers has an Internet connection established, the number of ports opened or closed on a server, and others.

Another important protocol is the SNMP [13]. This is a standard protocol for TCP/IP network management defined by IETF (Internet Engineering Task Force). SNMP requires few resources to be implemented using UDP transport protocol to effectively exchange data between agents and managers. There is a database of managed objects called MIB (Management Information Base), which contains data about the managed device, such as the uptime of a server. The agent provides the interface between the manager and physical device being managed, providing a monitoring of different network components such as switches, servers and wireless routers.

A different way to collect data from network devices is through NetFlow protocol developed by Cisco Systems [13], which enables the network administrator to export and collect data flow information from routers that support this technology.

The IETF has been working to standardize NetFlow considering Cisco NetFlow version 9 [13] as a starting point. This effort is called IPFIX (Internet Protocol Flow Information Export) [13]. IPFIX optimizes the export of data through a template, supporting extensibility and adapting to different scenarios. In addition, a file format is specified for storing data that has been received. It facilitates the interoperability and reusability among a wide variety of flow storage, processing and analysis tools.

C. *Management Station*

The management station needs a tool to perform data collection from different devices in computer network. Its architecture is based on a MVC model (model-view-controller). This model divides the features of a system in layers, facilitating the maintenance and creation of new interfaces. The three layers and their functions are [14]:

- Model: Represents the logical layer. The whole business rule and data persistence are located in this layer. In this work it also aims to make requests to read and collect data from the computer network.
- View: User interaction layer or application interface. In this layer the data collected are pre-processed and viewed by network administrators through reports and graphs.
- Controller: This layer is responsible for determining the application flow, processing user actions and transmitting data to model layer.

Figure 3 illustrates the layers of this management station and its interaction between the network administrator and the computer network.

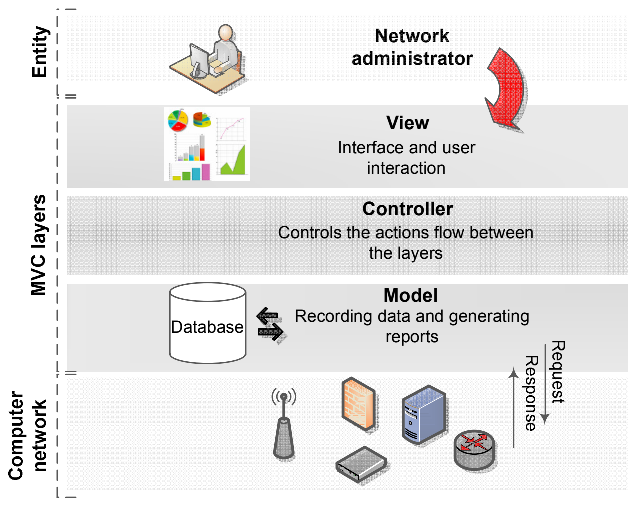

Entity
MVC layers
Computer network
Network administrator
View
Interface and user interaction
Controller
Controls the actions flow between the layers
Model
Recording data and generating reports
Database
Request
Response

Figure 3. MVC Layers and its interactions.

In this example, the network administrator performs an interaction with view layer that allows managing computer network. Then, the controller layer processes the actions to the model layer, which performs requisitions to agents via communications protocols. The protocol returns the collected data to model layer and stores it in a database. With data collected and stored, network administrator can execute queries and analyze information using, for example, graphs and reports.

VI. A Practical Application

In order to demonstrate and validate the proposed solution, tests were performed in a real computer network environment, the Communication Networks Laboratory (LaRCom) at the University of Campinas (Unicamp).

Currently, LaRCom has a computer network that offers a 1Gb Ethernet network, whose external internet connection is provided by Unicamp. It is also composed by several equipments such as application servers, database servers, printers, switches, VoIP terminals and laptops, all working on IP protocol. This infrastructure supports technology innovation projects that use this network to develop new solutions and tools such as the development of e-government to local governance and technologies for education.

Due to the intense use of services provided by LaRCom, some issues might occur, such as: slowness in certain servers, connection problems related to the Internet link, instability on firewall, deficiency in electrical power supply and others. In order to minimize these problems and help the security manager's duties, some security metrics were applied in this network. Then, it was developed an Integrated Platform for Security Metrics Analysis (IPSMA) tool based on the architecture proposed in this paper. IPSMA was implemented using Java technology and Oracle database management system to store data.

A. *Security Metrics Application*

First of all, some components of LaRCom were selected and monitored. The selection criteria here were choosing the essential and frequently used components. Figure 4 presents an overview of LaRCom and which components were selected to be monitored.

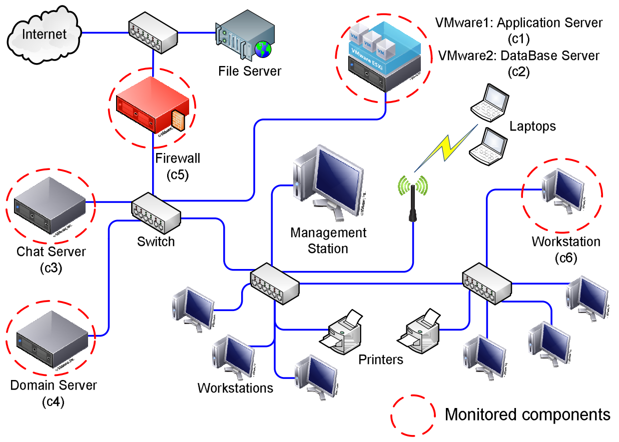


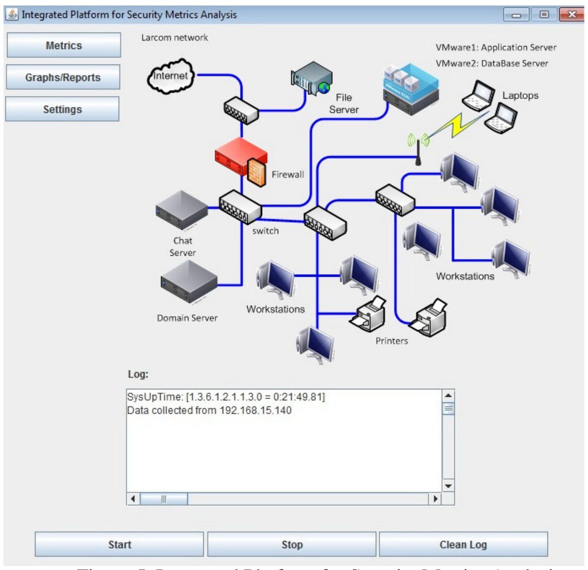
Figure 4. LaRCom network and monitored components.

Based on [2], [4] and [6], we selected metrics which satisfies the interests of organizations and effective the security management. The criteria presented in Section 4 also were used assisting metrics selection process and choosing which metrics would be automated and applied in the components of LaRCom. The following metrics were implemented: i) Uptime; ii) Number of open ports;

Uptime [4] has as a main goal to calculate the time that a computer, application or server is running. We used SNMP protocol to collect such data. Another metric used in this work, is the number of open ports [3]. Socket is used to perform communication with servers and identify the open ports. SNMP4J and Socket API were used for data collection of metrics, respectively. The metrics automation process was implemented using object oriented concepts supported by JAVA language programming.

Furthermore, it was developed a thread to collect each metric in order to initiate and control simultaneously the frequency of data collection. Data collection was performed continuously and automatically without intervention, during a period of one month without interruptions.

An illustration from IPSMA can be seen in Figure 5. This tool performs the automated data collection from metrics, generating reports and graphics. Moreover, it can help the network administrator to manage security metrics application and controls from organization.



```
SysUpTime: [1.3.6.1.2.1.1.3.0 = 0:21:49.81]
Data collected from 192.168.15.140
```

Figure 5. Integrated Platform for Security Metrics Analysis

Finally, Figure 6 shows the real network following the proposed architecture presented in Figure 2.

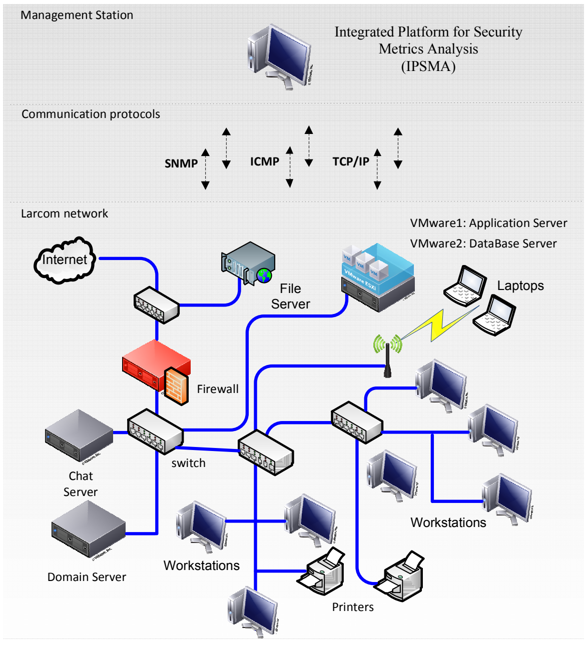


Figure 6. Real network following the proposed architecture

In this example, IPSMA was installed on the management station which uses different protocols to read and collect data from components. It contributes to integrate information control and support the security monitoring process.

B. *Discussion and Results*

Uptime [4] is a classic metric used to measure operations such as "uptime" and "downtime". Downtime is the total amount of time that resources were out of service planned or unplanned. Uptime is the total time for a given period minus any downtime.

Important data can be collected by applying this metric, such as: i) the total amount of time that resources were out of service due to regular maintenance; ii) the total elapsed time related to unexpected service outages; ii) the total time for a given period that a computer was up;

Pham [15] also presents a system to assess the overall security assurance. In this paper is used a metric to evaluate the time that two stations and a server are running. The results indicate that the station does not response to network requests at some points.

In our architecture also was applied this metric using only one platform to implement and collect data from metrics instead of several tools.

Figure 7 presents the total amount of time that components of LaRCom were up during a period of one month.

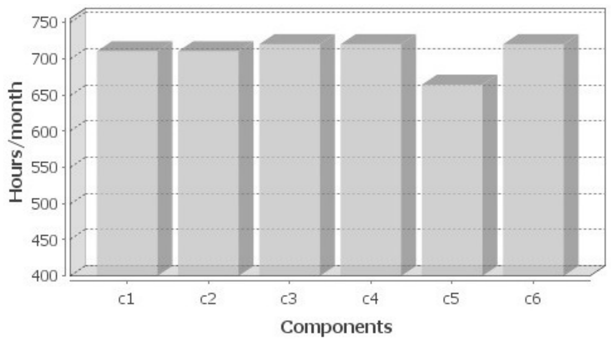


Figure 7. Total amount of "uptime" from components

As we can observe, the firewall (c5) stayed up an amount of hours smaller than other components. This data visualization can be detailed, for example, by the investigation of the number of shutdowns of each component per week, as seen in Table 1.

TABLE I. TOTAL NUMBER OF SHUTDOWNS OF EACH COMPONENT

	C1	C2	C3	C4	C5	C6
W1	-	-	-	-	1	-
W2	2	2	-	-	2	-
W3	-	-	-	-	4	-
W4	1	1	-	-	4	-

When analyzing Table 1 and Figure 8, it is possible to note that the application server (c1) and database server (c2) had three "shutdowns". According to the network administrator, these shutdowns were required and planned. The firewall (c5) had eleven unplanned "shutdowns". So, this fact alerted the network administrator of a potential risk or instability in equipment C5. Soon, it was found that the computer had some hardware troubles and it was sent to maintenance.

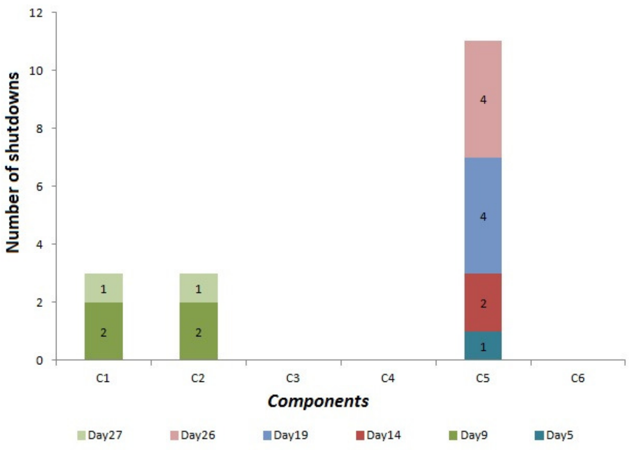


Figure 8. Days when the components have "shutdown"

Another metric was applied aiming to verify which server ports were opened, and also the frequency at which a certain door remains open. Thus, important data can be obtained to support security management such as: number of computers that allow remote access via terminal service, vulnerabilities which can be exploited through the open ports without the consent of the network administrator and others [16].

Table 2 presents the top five servers open ports of LaRCom and the number of components with open ports.

TABLE II. TOP FIVE OPENED PORTS OF LARCOM

Port	Service	Number
80	HTTP	3
3389	Terminal services	3
1521	Database connection	3
53	DNS (Domain name Server)	2
22	SSH	1

In addition, SANS [17] also reported the top ten ports that can be exploited by attacks or source of critical. When comparing Table 2 with [17], we found that there are ports such as 80, 3389 and 53 which were found in both lists. These ports are considered dangerous by [17] and the security policy in relation to these ports should be reconsidered. Ports 80 and 22 which are operating system standard also are considered easy targets of automated attacks and should be changed.

Ertürk [3] proposes a framework for security measurement, collection and reporting data. Further, the implementation was applied in a public organization.

The author also uses a metric to verify the number of open ports on servers. In this case, Nessus Vulnerability Scanner tool is used to collect data from several components. Nevertheless, other metrics are presented by the author and several monitoring tools, such as PRTG Network Monitor, are used to collect data. Due to this fact our architecture showed better to integrate information control and support the active security monitoring process using only one tool.

VII. CONCLUSION AND FUTURE WORK

In this paper, we proposed an architecture for automated collection of security metrics. Based on this architecture, also developed an Integrated Platform for Security Metrics Analysis performing a case study in a real computer network. A practical application was important to test and validate the behavior of architecture.

This application has brought some benefits to information security management, such as: an integrated solution able to support the monitoring of several components in a computer network; reduction in data collection time and greater reliability through automation; development of a historical database, which could be used to discover unexpected relationships and to reveal other predictive interactions that might exist.

The case study revealed some issues on the firewall and also breaches in the laboratory security police, related to several open ports in the investigated servers. This information is used to security analysis and can be improving by adding new security metrics.

Future work includes: i) development of new metrics using other network protocols, such as IPFIX; ii) improvement of the architecture in order to allow distributed management, solving a possible scalability issue caused by the large number of managed components; iii) deployment of the proposed architecture in other network scenarios.

ACKNOWLEDGMENT

The authors would like to thank the State of São Paulo Research Foundation (FAPESP) and the Coordination for the Improvement of Higher Education Staff (CAPES) that supports this work.

REFERENCES

[1] J. Lee, "Enabling network management using Java technologies," IEEE Communications Magazine, Vol. 38, January 2000, pp. 116–123.

[2] M. Swanson, N. Bartol, J. Sabato, J. Hash, and L. Graffo, "Security Metrics Guide for Information Technology Systems," National Institute of Standards and Technology, Special Publication 800–55, 2003. http://csrc.nist.gov/publications/nistpubs/800-55-Rev1/SP800-55-rev1.pdf retrieved: April, 2012.

[3] V. Ertürk, "A Framework Based on Continuous Security Monitoring", A thesis submitted to the Graduate School of Informatics of The Middle East Technical University, 2008. http://etd.lib.metu.edu.tr/upload/12610139/index.pdf retrieved: April, 2012.

[4] A. Jaquith, Security Metrics – Replacing Fear, Uncertainty and Doubt, Addison-Wesley, 2007.

[5] R. Miani, B. Zarpelao, and L. Mendes, "Application of Security Metrics in a Metropolitan Network: A Case Study," The 7th International Telecommunications Symposium (ITS 2010), 2010.

[6] The Center for Internet Security, "Quick Start Guide," Vol. 1.0.0, November 2010. http://www.cisecurity.org retrieved: April, 2012.

[7] P. Black, K. Scarfone, and M. Souppaya, "Cyber Security Metrics and Measures," National Institute of Standards and Tachnology, 2009. http://xlinux.nist.gov/~black/Papers/cyberSecurityMetrics2007proof.pdf retrieved: April, 2012.

[8] J.Park, N. Joo, and T. Kim, "Java-Based Network Management Environment," IEEE Internacional Conference on Communications, Vol. 2, June 1998, pp 1124–1128.

[9] H. Hegering, S. Abeck, and B. Neumair, Integrated Management of Networked Systems – Concepts, Architectures and Their Operational Application, Morgan Faufmann Publishers, 1998.

[10] http://www.isaca.org/COBIT/Pages/default.aspx retrieved: April, 2012.

[11] http://www.iso.org retrieved: April, 2012.

[12] R. Savola, "Towards a Security Metrics Taxonomy for the Information and Communication Ttechnology Industry," Proc. Internacional Conference on Software Engineering Advances (ICSEA 07), 2007, pp. 60.

[13] http://tools.ietf.org retrieved: April, 2012.

[14] An Oracle White Paper, Oracle Application Development Framework Overview, 2011. http://www.oracle.com/technetwork/developer-tools/adf/adf-11-overview-1-129504.pdf retrieved: April, 2012.

[15] N. Pham, L. Baud, P. Bellot, and M. Riguidel, "A Near Real-time System for Security Assurance Assessment," Proc. The Third Internacional Conference on Internet Monitoring and Protection (ICMP 08), 2008, pp. 1252–1260.

[16] http://www.vulnerabilityassessment.co.uk retrieved: April, 2012.

[17] http://isc.sans.edu/top10.html retrieved: April, 2012.

Experimental Evaluations of the Impact of High availability on the Quality of Protection of Hybrid Clouds

Syed Naqvi, Michael Van de Borne, Arnaud Michot

Centre d'Excellence en Technologies de l'Information et de la Communication (CETIC)
29 Rue des Frères Wright, 6041 Charleroi, Belgium
{syed.naqvi, michael.vandeborne, arnaud.michot}@cetic.be

***Abstract*—A common perception of security services is an overhead on distributed system's performance that exponentially increases with scale and heterogeneity of the system's components. While this perception is not untrue, there is no precedence of experimenting the exact impact on its overall performance in terms of quality of protection. This paper presents a formal way of testing the impact of scalability and heterogeneity on the federated Cloud security services. The work presented in this paper aims to develop a mean of quantifying the impact on security functions under various operating conditions and parameters of federated Cloud deployments. The results of this experimental study will help businesses to identify the best security architecture that will fit their Cloud architectures and performance requirements.**

Keywords-Cloud computing; security architecture; performance parameters.

I. INTRODUCTION

Monitoring of security performance is an established requirement of highly available services and infrastructures [1]. Moreover, insight into the composition of security mechanisms is one of the research areas identified by the United States National Institute of Standards and Technology (NIST) in its report on directions in security metrics research [2]. However, as of today, most of the proposed approaches only deal with the monitoring of application level security measures such as monitoring of antivirus updates, installation of patches, IDS (intrusion detection system) buffer monitoring, etc. The range of available commercial products to perform security monitoring generally examine various high level parameters. For example, Cisco's MARS (Monitoring, Analysis and Response System) [3] is designed to monitor logs and to monitor threats; Hewlett Packard's IT Performance Suite [4] is designed to monitor security and risk management processes (such as number of systems under security control, risk indicators, etc.). However, none of these tools provide any information about the monitoring of the impact on security at the infrastructure level.

It is generally understood that security takes its toll on system performance; and providing consistent security to scalable distributed systems requires careful consideration of performance overheads [5]. However, no tangible efforts are made to quantify the performance degradation by relating the quality of protection with the system performance. The knowledge of the impact of security at infrastructure level is particularly important to cope with the emerging challenges of hybrid Clouds such as scalability, heterogeneity, criticality of their applications, etc. It is therefore important to determine an empirically validated function for security services: on different application loads; for a number of virtual machines; on different cloud technologies; with different types of hypervisors.

In this paper, we present our security monitoring experiment that aims to examine the implications of security on the back-end system of emerging realm of IT services - i.e. hybrid Cloud infrastructures. This experiment - ExSec: Experimenting Scalability of Continuous Security Monitoring - is one of the experiments of European Future Internet experimental facility and experimentally-driven research project BonFIRE [6]. Main objectives of the ExSec experiment is to study and quantify the impact on the quality of protection of Future Internet based applications that will be highly scalable in nature and use heterogeneous underlying technologies. These experimental evaluations will be useful to determine the *stretching limit* of Cloud security functions; and eventually, workout some remedial solutions especially to explore the possibility of making use of abundance of Cloud resources to compensate the performance degradation.

The test scenarios of the ExSec experiment are designed to reflect real-life situations where in a routine business context, organizations forming a virtual organization (VO) are most likely to run heterogeneous cloud managers; and hence the situation often arises where hypervisors of different types using different virtual execution environment managers are required to collaborate to form a VO. The bottom line of this study is to develop a mean of quantifying the impact on security functions under various operating conditions and parameters of Cloud deployments.

The rest of this paper is organized as: Section II outlines the context of our work. A set of security policy rules and impact factors are outlined in Section III. The experimental set up of our study is detailed in Section IV. Section V provides a pragmatic discussion of the scope and perspectives of our work. Finally, some conclusions are drawn in Section VI together with a brief account of our future directions.

II. Context and Motivations

A. *Background of Experimental Study*

We implemented a security framework for Grid computing environments in one of our previous European projects - GridTrust [7]. This work included a prototype implementation of the *Usage Control (UCON)* concept in the context of virtual organizations (VO). This work consisted of a vertical approach for Grid security from requirements level right down to application and middleware levels. The GridTrust framework provides policy-driven autonomic access control solutions that provide a continuous monitoring of the usage of resources (usage control) by users. This groundwork on the deployment of UCON model in the security framework of a highly distributed environment gave us an insight in the problematic of security monitoring; and the awareness of the impact of scale on security performance. Unlike access control, usage control perpetually monitors security parameters and is directly affected by the scale of the system being controlled. Therefore, its impact on the performance may become significant enough to be overlooked. However, to the best of our knowledge, there is no formal way of quantifying this relationship or to extrapolate it for complex scenarios.

We explored the monitoring of Cloud security services in the European Cloud computing flagship project RESERVOIR [8] with particular focus on the audit logging for data location compliance issues. The underlying Cloud technology used in the RESERVOIR project was OpenNebula [9] that was uniformly deployed across various sites of the RESERVOIR Cloud. We are therefore using a FIRE (Future Internet Research Experimentation) facility - more precisely the BonFIRE project - now to have access to a large-scale heterogeneous Cloud environment that provides opportunity to perform tests on real infrastructures of scale. Moreover, our study is facilitated by a range of technical solutions provided by the BonFIRE infrastructure such as monitoring tool (ZABBIX [10]), client library for RESTful APIs (RESTfully [11]), JSON [12] interface for data-interchange, etc.

B. *Related Work*

Most of the ongoing Cloud computing endeavors are still pointed to the challenges of their large scale deployments. Security is undoubtedly the cornerstone of these deployments; however, the progress in this area is still in its earlier stage. Therefore, not too many initiatives are taken in this direction so far. Some of these approaches are evaluated in this section.

CloudSec [13] provides active, transparent and real-time security monitoring for multiple concurrent VMs hosted on a Cloud platform in an IaaS setting. CloudSec employs VMI (virtual machine introspection) [14] for monitoring VMs at the hypervisor level. CloudSec aims to protect kernel data structures. However, it does not address the impact of these multiple concurrent VMs on Cloud platform security.

Another important security area where monitoring technology has an important role to play is digital investigations. Security monitoring can facilitate conception of forensic friendly IT infrastructures such as electronic communications [15]. Reliability of monitoring data is of prime importance in this domain as the validity of data as an *acceptable proof* in court of law is crucial in digital investigations. This aspect of security monitoring is useful to keep track of security information. However, it does not offer any solution to the problem of estimating the accuracy of security monitoring for a particular scale and its dependence on specific underlying technology.

Our work presented in this paper is a pioneer initiative in the direction of quantitative analysis of the impact of scalability and heterogeneity on security functions. This work will enable businesses to deploy some Cloud solutions with optimal security architecture that will fit to their Cloud architectures and performance requirements.

III. Security Policy for the Evaluations

This section outlines the security policy for the ExSec experiment's evaluations. The overall objective of this rigorous security policy is to ascertain a real life situation that will use the federated cloud infrastructure for its routine operations. The nature and scope of such paradigm will require simultaneous fulfillment of several policy rules. This situation will strain the overall security policy enforcement mechanism in general; and its policy decision point (PDP) in particular.

A. *Scenario description*

The security policy scenario depicts a Future Internet based social application where access to digital contents (such as music files) requires a number of conditions to be satisfied such as:

1. The foremost condition is to ensure that the subject has paid for the contents he/she intends to access.
2. The type of contents (e.g. latest songs or the songs released a couple of years before the access request)
3. The type of access (e.g. premium for priority download, or ordinary for slower download speed)
4. The access limit (e.g. unlimited access or some limits are applied such as maximum number of songs that can be downloaded in a given time; specific download time – night only, weekend only, etc.
5. The geographical location of the user to protect the rights of the digital distributions.

Moreover, a number of events may raise suspicions leading to some corrective measures such as:

1. If a user with individual subscription simultaneously attempts to access the contents from different locations.
2. If a user makes more than a specific number of futile attempts to access the contents.

Figure 1 depicts a non-exhaustive list of major decision parameters for the Policy Decision Point (PDP) for the abovementioned Future Internet based digital contents application. The impact of the range of these parameters on the performance of PDP is exacerbated when a big number of (scalable) requests are concurrently made to the application gateway. We aim to quantify this impact and establish relationship between the load on the PDP and its impact on the performance. System calls are intercepted to have a lower level control over the incoming requests. Major security challenges of this scenario are to ensure firm access control despite higher scale of requests; to enforce usage control policies; and to cope with the heterogeneity of the underlying technologies.

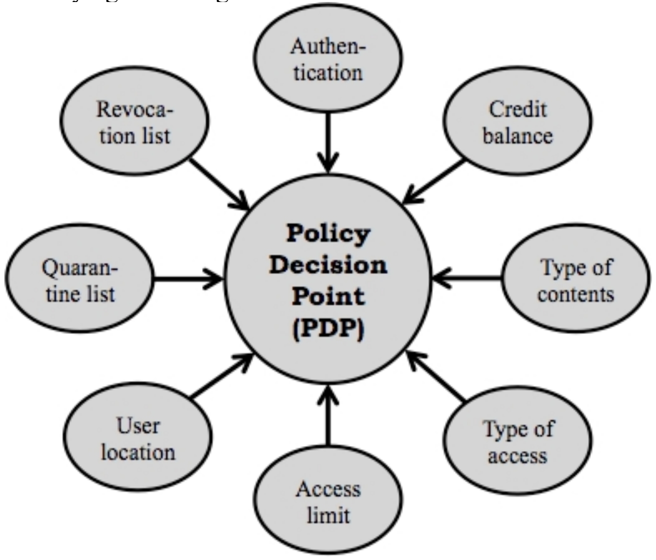


Figure 1. Overview of the security policy scenario

B. *Policy rules*

This section summaries some example security policy rules and the impact parameters to be studied in the course of ExSec experiment.

1) *Access control policy rules*

- **AC1:** Access to digital contents files is restricted to users in good standing
- **AC2:** Have enough credit for requested digital contents
- **AC3:** Maximum number of digital contents files is not reached
- **AC4:** Access digital contents from the eligible location

2) *Usage control policy rules*

- **UC1:** Maximum number of downloads granted
- **UC2:** Maximum number of downloads in a specific time (e.g. per hour)
- **UC3:** Maximum number of downloads of specific type of files (e.g. classical music)

3) *Monitoring policy testing scenarios*

- Security verification for an end-user web application running on the same underlying VM Host technology - for example KVM.
- Security verification for an end-user web application running on the different underlying VM Host technologies - for example KVM and Xen.
- Security verification for an end-user web application running on the same underlying Cloud engine - for example a sub-experiment on OpenNebula or a proprietary technology such as the one offered by the HP for BonFIRE.
- Security verification for an end-user web application running on the heterogeneous Cloud engines - for example, a cloud management environment composed of OpenNebula and HP technology.
- Security verification for a completely heterogeneous environment with diverse VM Hosts and Cloud engines.

4) *Impact parameters*

- Gradually increase number of clients (download requests) to get access to digital contents file (cf. network throughput, CPU utilization, RAM, etc.)
- Add timeslot constraint for the download requests (maximum number of permissible downloads per hour)
- Add file type constraint for the download requests (type of digital contents that can be downloaded – e.g. files of type A, B, and C)
- Add location constraint for the download requests (such as number of download requests per server)
- Placement of PEP/PDP
- Within a VM
- One at each BonFIRE site
- One for the entire infrastructure

IV. Experimental Setup of Our Study

A security policy enforcement architecture using XACML [16] is deployed as a first step of this study so as to secure access to the BonFIRE resources. This security policy enforcement architecture is shown in Figure 2. Access to the BonFIRE infrastructure is based on the authentication mechanism where we have installed a filter to intercept all HTTP requests sent to the ExSec experiment's WebApp. This filter constructs an XACML request with the client information (e.g. the requesting IP address). The Policy Enforcement Point (PEP) invokes the *AccessControl* service followed by the evaluation of access request by the Policy Decision Point (PDP) that returns TRUE if the access is permitted, or FALSE otherwise. The interactions of PEP and PDP together with their port numbers are provided in Figure

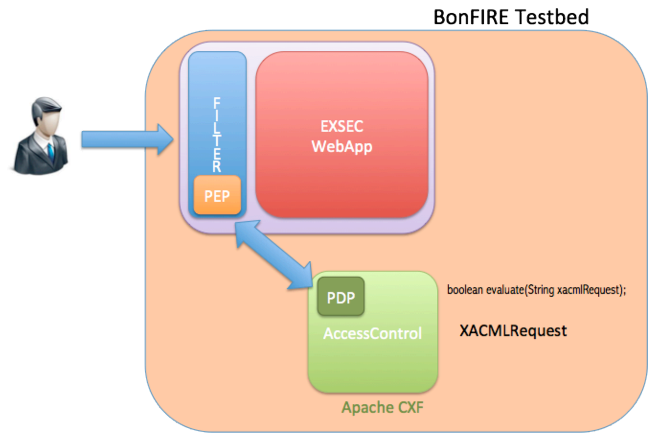


Figure 2. ExSec credentials management system

3. Additional filters will be used for monitoring of security parameters by using UCON model.

We now present fine-grained architectural information of our proposed architectural set-up. We first describe different schemes of placing security policy enforcement points (PEP) in the BonFIRE computer resources followed by the number of testing scenarios for the study.

A. *Case-1: When PEP module is integrated inside server node*

Jikes Java Virtual Machine (JVM) launches a RESTlet HTTP server. Every system call from this REST server to gain physical access to a file (e.g. to open a file) is intercepted by the Jikes JVM, which queries the PEP module. The latter applies the Usage Control security policy (i.e. one download at a time), and returns this answer to Jikes, which grants or denies the physical access request for a file. In this case, there is a PEP module shipped with each compute node. This scheme is presented in the Figure 3.

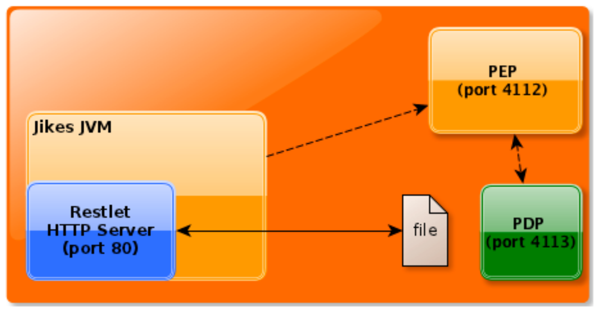


Figure 3. BonFIRE Compute Resource with integrated PEP module

B. *Case-2: When PEP module lies outside server node*

In this set-up, the server nodes are similar to the case-1, but the PEP module is hosted in an external VM. Requests to the external PEP module are triggered over the network. This scheme might result in some performance degradations. ExSec experiment aims to measure the impact of using external PEP on the functioning of security policy enforcement. The architectural set-up is shown in the Figure 4.

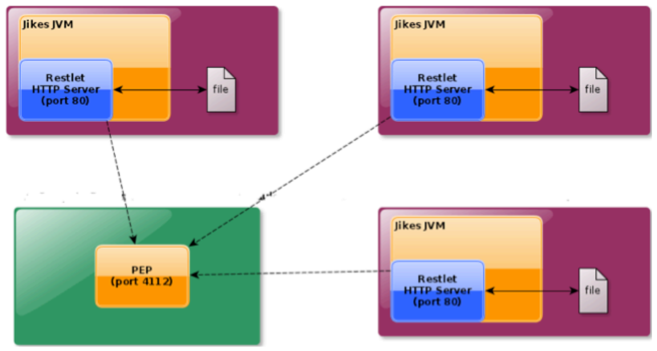


Figure 4. BonFIRE Compute Resource with external PEP module

C. *Testing scenarios*

This section outlines the set of scenarios of ExSec experiment to study the impact of security.

1) *Basic test scenarios*

First we deployed a basic test scenario where only two VMs are used. One contains a server node configuration with an integrated PEP module; and the other performs download request to the first one. We use *HAProxy* load-testing tool to generate large numbers of HTTP client requests to simulate multiple clients. The behavior of the server node is analyzed according to the number of client requests it serves. This scenario is shown in the Figure 5. This test is designed to study the impact of a sizable number of clients' requests on the infrastructure performance.

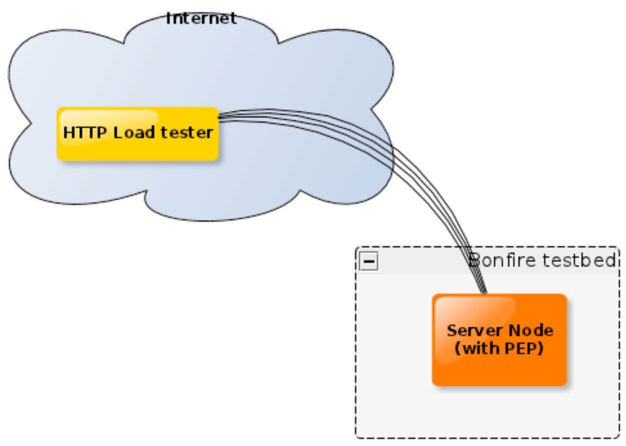


Figure 5. Basic load testing scenario for BonFIRE testbed

2) *Load testing scenario (integrated PEP modules in servers)*

This scenario, as shown in the Figure 6, is meant to analyze the system behavior when multiple client requests are distributed across several server nodes. These could be spread across various BonFIRE testbeds. In this scenario, the PEP modules are integrated into the server nodes. Synchronization of different PEPs can incur some performance overheads. We workout the impact on performance due to increasing load and management overheads.

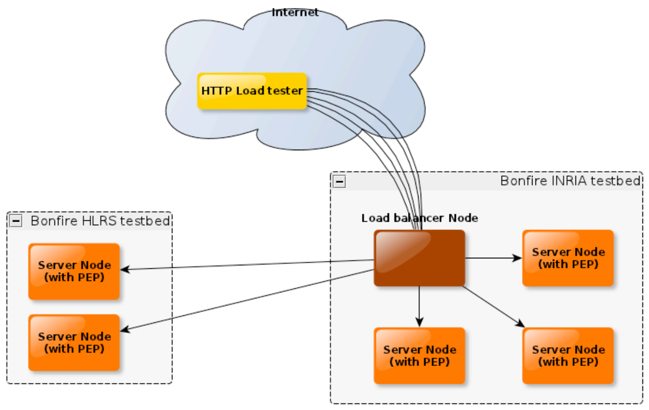


Figure 6. Load testing scenario with integrated PEP module

3) Load testing scenario (one PEP module per BonFIRE site)

This scenario is similar to the previous one, but server nodes are no longer hosting the PEP modules. They are externalized on separate computing resources. However, this scenario provides one PEP module per BonFIRE testbed. This scheme is shown in Figure 7.

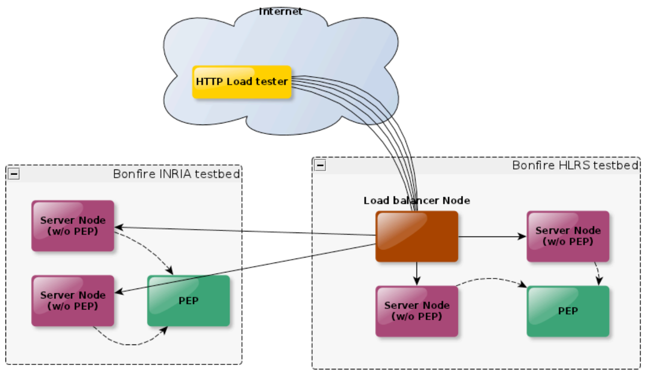


Figure 7. Load testing scenario with one PEP module per BonFIRE site

4) Load testing scenario (one PEP module for the entire infrastructure)

This is a similar scenario as the previous one. However there is only one PEP module, so that server nodes will have to perform Usage Control policy requests across BonFIRE sites. This arrangement is shown in the Figure 8. It would be more measurable to have the PEP on an independent node without server node on it.

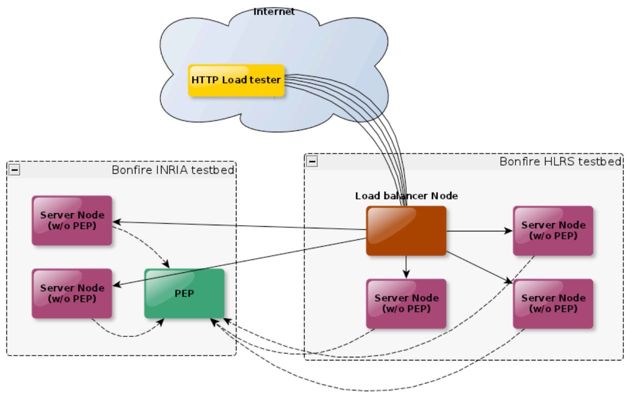


Figure 8. Load testing scenario with one PEP module

V. DISCUSSIONS

Impact on security functions of the wireless devices and platforms is widely explored [17][18][19], mainly due to their significance on the power consumption. The principal objective of these studies is pointed towards improving the battery life of wireless devices instead of investigating the impact on the global functioning of the resources. Whereas, ExSec is aiming to quantify the impact on security performance by studying the impact of its different parameters.

Impact of security services in a client/server exchange of information is evaluated in [20]. The composition of these environments is generally fixed and investigation of the impact of security services on the performance requires less parameters compared to virtual, dynamic and decentralized environments such as federated Clouds. Heterogeneity besides scalability is a non-trivial challenge that we are facing in ExSec experiment. The dynamic nature of the distributed systems is giving rise to *adaptive* security monitoring systems [21]. The decision making process for their adaptiveness also inflicts performance overheads. However, this performance factor is not considered in our work.

Hardware-based security solutions such as *Trusted Computing* [22] are often seen as rigorous in quality of protection. This inspiration has led to the development of hardware-based process security monitoring system such as VMInsight [23] that can provide load-time and run-time monitoring for processes. It can be an interesting follow-up direction for the ExSec experiment; however, the current BonFIRE infrastructure has no Trusted Platform Module (TPM) support. We can nevertheless envision extrapolating ExSec results on some Cloud infrastructure with TPM such as *CertiCloud* [24].

Our work mainly deals with infrastructural side of the performance impact on security. The results can be used to advise Cloud customers and users on the security and performance tradeoffs. However, infrastructure providers (such as Amazon EC2 [25]) do not take responsibility of ensuring protection of their customers' contents. For example, clause 4.2 of Amazon's Customer Agreement explicitly ask their customers to be responsible for taking necessary security measures. It clearly states: *You are responsible for ... taking your own steps to maintain appropriate security and protection, which may include the use of encryption technology to protect Your Content from unauthorized access and routine archiving Your Content.* Likewise, a recent study of *Security of Cloud Computing Providers* [26] reported that around three quarters (73% of US and 75% of European service providers) responded that their cloud services do not substantially protect and secure their customers' confidential or sensitive information. Moreover, nearly two-thirds of the responded (62% of US and 63% of European providers) were not confident that their Cloud applications and resources were secure. It is therefore necessary for the Cloud customers to enforce their security policy and to ensure that its impact on performance remains within acceptable range. Our work can be useful for this kind of public. They can use it together with some trust establishing mechanism for choosing the most appropriate Cloud provider. Example of trust establishing mechanism includes *privacy penetration testing* [27].

VI. CONCLUSIONS AND FUTURE DIRECTIONS

This paper presented our experimental investigations that aim to relate two peculiar characteristics of highly available systems - scalability and heterogeneity - with the performance of security functions. We are working to develop a formal way of quantifying the impact on security services under various operating conditions and parameters of federated Cloud deployments. We as a technology transfer centre perform this experimental study to help businesses (especially SMEs) identify the best security architecture that will fit their Cloud architectures and performance requirements.

Our study is a pioneer work in analyzing the impact of the peculiar characteristics of hybrid Cloud architectures on the much-needed effective security solutions. We are conducting this experimental study on a real life operational hybrid Cloud infrastructure through a set of test scenarios depicting real-life situations of routine business environments. Our work will also stimulate new research directions in the area of Cloud security and its performance parameters; as the security solutions of the pre-Cloud era may not be simply ported to this novel paradigm without necessary improvements.

We are currently implementing usage control (UCON) security policy. Our future directions include implementation of more complex policy rules to better reflect the emerging security requirements. Examples include security policy rules for *reputation* component that grants or denies usage requests according to the client's reputation. We plan to explore such new policies to broaden the scope of the ExSec experiment beyond the single experimentation policy. We are also going to explore ways of compensating performance degradation by making use of available Cloud resources to fill the gap.

ACKNOWLEDGMENT

The research leading to the results presented in this paper has received funding from the European Union's seventh framework programme (FP7 2007-2013) Project BonFIRE under grant agreement number 257386. Authors would also like to express their gratitude to Damien Hubaux and Florian Schreiner who have provided their valuable help in conducting the ExSec experiment.

REFERENCES

[1] Data Center Post, How Network Performance and Security Monitoring are Useful in Today's Data Center, online article at Technorati.com, 26 April 2011 [retrieved: June 2012]

[2] W. Jansen, Directions in Security Metrics Research, NIST Interagency/Internal Report (IR) no. 7564, March 2009 [retrieved: June 2012]

[3] Cisco product: Security Monitoring, Analysis and Response System (MARS) [retrieved: June 2012]

[4] Hewlett Packard (HP) product: IT Performance Suite [retrieved: June 2012]

[5] J.M. Cary, Data security and performance overhead in a distributed architecture system, UMI Research Press, 1981 [retrieved: June 2012]

[6] European 7th Framework project BonFIRE: Building Service Testbeds on FIRE (Future Internet Research Experimentation)

[7] S. Naqvi and P. Mori, Security and Trust Management for Virtual Organizations: GridTrust Approach, IFIP International Conference on Trust Management (IFIPTM'09), West Lafayette, USA, June 2009

[8] European 7th Framework project RESERVOIR: Resources and Services Virtualization without Barriers

[9] The OpenNebula Cloud Management Engine [retrieved: June 2012]

[10] The ZABBIX Performance Monitoring Tool - www.zabbix.com

[11] Restfully: A general-purpose client library for RESTful APIs [retrieved: June 2012]

[12] The JavaScript Object Notation (JSON) data-interchange format [retrieved: June 2012]

[13] A.S. Ibrahim, J.H. Hamlyn-Harris, J. Grundy, and M. Almorsy, CloudSec: A security monitoring appliance for Virtual Machines in the IaaS cloud model, in proceedings of the 5th International Conference on Network and System Security (NSS), 2011, pp.113-120, 6-8 September 2011, Milan, Italy, ISBN: 978-1-4577-0458-1

[14] T. Garfinkel, and M. Rosenblum, A Virtual Machine Introspection Based Architecture for Intrusion Detection, in proceedings of the 10th Network and Distributed Systems Security Symposium (NDSS) 2003, pp 191-206, 6-7 February 2003, San Diego, CA, USA

[15] F. Van Staden, and H. Venter, Adding digital forensic readiness to electronic communication using a security monitoring tool, in proceedings of the IEEE Information Security South Africa (ISSA) 2011, 15-17 August 2011, Johannesburg, South Africa

[16] OASIS Extensible Access Control Markup Language (XACML)

[17] S. Kolahi, Z. Qu, B. Soorty, N. Chand, The Impact of Security on the Performance of IPv4 and IPv6 Using 802.11n Wireless LAN, in Proceeding of the 3rd International Conference on New Technologies, Mobility and Security 2009 (NTMS 2009), 20-23 December 2009, Cairo, Egypt, ISBN: 978-1-4244-4765-7

[18] J. Schonwalder and V. Marinov, On the Impact of Security Protocols on the Performance of SNMP, IEEE Transactions on Network and Service Management, pp.52-64, Vol. 8, Issue 1, 2011

[19] A. K. Agarwal and W. Wang, Measuring performance impact of security protocols in wireless local area networks, in Proceedings of the 2nd IEEE International Conference on Broadband Networks 2005 (BroadNets 2005), 3-7 October 2005, Boston, MA, USA

[20] S. Cavalieri, G. Cutuli, S. Monteleone, Evaluating impact of security on OPC UA performance, 3rd IEEE Conference on Human System Interactions 2010 (HSI 2010), 13-15 May 2010, Rzeszow, Poland

[21] R. Savola and P. Heinonen, Security-Measurability-Enhancing Mechanisms for a Distributed Adaptive Security Monitoring System, in proceedings of the Fourth International Conference on Emerging Security Information, Systems and Technologies 2010 (SECURWARE'10), 18-25 July 2010, Venice, Italy

[22] The Trusted Computing Group technologies [retrieved: June 2012]

[23] X. Li, C. Jiang, J. Li, B. Li, VMInsight: Hardware Virtualization-Based Process Security Monitoring System, in Proceedings of the IEEE Int. Conference on Network Computing and Information Security 2011 (NCIS 2011), 14-15 May 2011, Guilin, China

[24] B. Bertholon, S. Varrette, and P. Bouvry, CertiCloud: A Novel TPM-based Approach to Ensure Cloud IaaS Security, in Proceedings of the 4th IEEE International Conference on Cloud Computing 2011 (CLOUD 2011), 4-9 July 2011, Washington DC, USA

[25] Amazon Elastic Compute Cloud (Amazon EC2)

[26] Ponemon Institute study (sponsored by CA Technologies), Security of Cloud Computing Providers Study, April 2011 [retrieved: June 2012]

[27] C. Probst, A. Sasse, W. Pieters, T. Dimkov, E. Luysterborg, and M. Arnaud, Privacy penetration testing: How to establish trust in your Cloud provider, In: European Data Protection - In Good Health? S. Gutwirth, R. Leenes, P. De Hert, Y. Poullet (Eds.), 2012

Characterizing and Predicting Internet Users Hidden Behaviors with Markovian Models

Paulo Salvador, António Nogueira, Eduardo Rocha
Instituto de Telecomunicações, DETI, University of Aveiro
Aveiro, Portugal
E-mails: salvador@ua.pt, nogueira@ua.pt, erocha@av.it.pt

***Abstract*—The ability to characterize and predict Internet users behaviors in environments where only layer 2 statistics are available can be very important for a network operator. At network entry points, like Wi-Fi or WiMax access points or UMTS or LTE base stations, the operator can perform a low level monitoring of the communications independently of the data encryption level and even without being associated with the network itself. Based on this low level network data, it is possible to infer the user behavior, optimize the access service and offer new security threat detection services. The user behavior inference consists in identifying the underlying web application that is responsible by the layer 2 traffic at different time instants and characterize the usage dynamics of the different web applications. Many identification methodologies have been proposed over the last years to classify/identify IP applications, including port-based analysis, deep packet inspection, behavior-based approaches and learning theory, each one having its own advantages and drawbacks. However, all these methodologies fail when only low level statistics are available or under data encryption restrictions. We propose the use of multiscaling traffic characteristics to differentiate between different web applications and the use of a Markovian model to characterize the dynamics of the user actions over time. By applying this methodology to Wi-Fi layer 2 traffic generated by users accessing different common web services/contents through HTTP (namely social networking, web news and web-mail applications), it was possible to achieve a good matching and prediction of the users behaviors. The results show that the proposed multiscaling traffic Markovian model has the potential to identify, model and predict Internet users behaviors based only on layer 2 traffic statistics.**

Keywords - User profiling; wavelets; multiscaling behavior; multivariate Gaussian distributions; Markovian modeling; behavior prediction.

I. INTRODUCTION

Identifying different behaviors of Internet users by analyzing the application types they are running is the key issue of many crucial network monitoring and management tasks. Basic network management functions such as quality of service improvement, network equipment optimization and security threats detection are all based on the ability to accurately classify network traffic into the right corresponding application and describe users behavior over time. Most existing approaches are based on static information about the applications (such as the name and type of the application, its owner, the execution time, or the host on which the application was executed). However, such approaches are not applicable to scenarios involving low level monitoring, traffic encryption or under stringent confidentiality requirements, since they rely on analyzing specific fields of the packet header.

So, this paper proposes the use of multiscaling traffic characteristics to differentiate between different web applications and the use of a Markovian model [1], [2] to characterize the various dynamics of the user actions over time. This methodology will be able to identify and predict the different user behaviors, even if this information is somehow hidden when performing a classical statistical analysis of the generated traffic.

Besides, the proposed methodology can be applied to scenarios where existing identification approaches are not applicable or have limited efficiency, like low level monitoring and service optimization at Wi-Fi [3] or WiMax [4] access points and Universal Mobile Telecommunications System (UMTS) [5], [6] or Long Term Evolution (LTE) [7] base stations.

One of the first and most common forms of traffic classification is port-based classification, which relies on the port numbers employed by the application at the transport layer. However, since many modern applications use dynamic ports negotiation, port-based classification became ineffective [8], [9], with accuracy ranges between 30% and 70%. Chronologically, the next proposed classification technique was deep packet inspection (DPI) or payload-based classification, which requires the inspection of the packets' payload: this classifier extracts the application payload from the layer 4 data unit and searches for a signature that can identify the flow type. Although DPI is widely used by today's traffic classifier vendors, being very accurate [10], [9] for some scenarios, it is unable to deal with low level or encrypted data. Very efficient classification techniques that perform traffic identification without accessing user data were proposed [11], but they also rely on layer 3 and layer 4 traffic statistics that may not be available for the operator at specific network entry points, can be protected by encryption or restricted by confidentiality requirements.

In this work, we propose a methodology for the differentiation of Internet applications based on the multiscaling statistical analysis of low level traffic, together with a modeling approach of the user preferences over time. It is known that several frequency components are introduced by mechanisms operating at different scales of analysis, including user interactions, flow sessions and individual packets dynamics. This creates characteristic multiscaling signatures that can be used to perform an accurate differentiation of the different web applications. A wavelet scalogram [12], which describes the signal energy simultaneously on a frequency and time domain, is constructed based on the wavelet multiscale decomposition of a traffic counting process and can be used to create multiscaling statistical signatures for each web application. The wavelet scalogram communicates the time frequency localization property of the discrete wavelet transform, being possible to capture the correlation that exists between the time variability of the process and the different scales.

The results obtained by applying the proposed methodology to layer 2 traffic promiscuously captured in the vicinity of a Wi-Fi network access point (without authenticating) show that it is able to achieve a good identification accuracy. It was possible to identify, model and predict the behavior of users accessing three common web applications: social networking (without chatting and game interactions), news web journals and web-mail.

For validation purposes, the ground-truth of the data was created by asking a pre-determined set of users to replicate their traditional Internet behavior using a controlled environment (user terminals and network).

The remaining part of this paper is organized as follows: Section II presents some of the most relevant related work on statistical classification of web applications and user behavior modeling; Section III presents some important background on multiscaling analysis; Section IV presents the details of the proposed identification methodology and behavior model; Section V presents the results of a proof-of-concept of the methodology and, finally, Section VI presents some brief conclusions about the presented model and identification methodologies.

II. Related work

The statistical approach to classification is based on collecting statistical data of the network flow, such as the mean packet size, flow duration, number of bytes per time interval, number of packets per time interval, etc. The statistical paradigm relies on the assumption that each application has a unique distribution of properties that represents it and can be used to univocally identify it. This approach has been the subject of intensive research in recent years.

First of all, Paxson *et al.* [13] established a relationship between flow application type and flow properties (such as the number of bytes and the flow duration). In [14], the authors proposed a methodology for separating chat traffic from other Internet traffic using statistical properties such as packet sizes, number of bytes, duration and packets inter arrival times. In [15], Mcgregor *et al.* explored the possibility of forming clusters of flows based on flow properties such as packet size statistics (e.g., minimum and maximum), byte count, idle times, etc., using an expectation maximization (EM) algorithm to find the clusters' distribution density functions. A study focusing on identifying flow application categories rather than specific individual applications was presented in [16]. Although it was limited by a small dataset, the authors have been able to show that the k-nearest neighbor algorithm and other techniques can achieve good results, correctly identifying around 95% of the flows. In reference [17], the authors were able to obtain an average success rate of 87% in the separation of individual applications using an EM based clustering algorithm. In [18], Moore *et al.* studied the basic Navie Bayes algorithm, enhanced by certain refinements, showing that it is able to achieve an accuracy level of 95%.

In [19], realtime classification was addressed by studying the feasibility of application identification at the beginning of a TCP connection: based on an analysis of packet traces collected on eight different networks, the authors found that it is possible to distinguish the behavior of an application from the observation of the size and the direction of the first few packets of the TCP connection. Three techniques were applied to cluster TCP connections: K-Means, Gaussian Mixture Model and spectral clustering. Crotti *et al.* [20] presented a realtime classification mechanism based on three simple properties of the captured IP packets: their size, inter-arrival time and arrival order. Based on new structures called protocol fingerprints, which express these quantities in a compact way, and on a simple classification algorithm based on normalized thresholds, the proposed technique showed promising results on classifying of a reduced set of protocols. In [21], a traffic classification approach based on Support Vector Machines (SVM) was proposed: using a simple optimization algorithm, a statistical traffic classifier was able to perform correctly with only a few hundred samples for training. Note that these algorithms were tested only against basic application protocols. Encrypted applications communications add additional constraints to the detection problem by making the traffic packet headers and data inaccessible to network based monitoring systems. Therefore, the detection methods that rely on packets headers/data information are completely inappropriate in encrypted communications scenarios [8], [22].

Bar-Yanai *et al.* [23] introduces a hybrid statistical algorithm that integrates the k-nearest neighbors and k-means machine learning algorithms. The proposed algorithm is fast, accurate and is insensitive to encrypted traffic, overcoming several weaknesses of the DPI approach (like asymmetric routing and packet ordering). The strength of the algorithm was demonstrated on encrypted BitTorrent, which is known to use packet encryption, port alternation and packet padding (on initial flow packets) to avoid detection.

The BLINC [11] approach is based on observing and identifying patterns of host behavior at the transport layer, analyzing the social, functional and application level patterns. The fact that this approach relies on layer 3 and layer 4 traffic statistics makes it impossible to be used by an operator in certain entry points of the network where only low level data is available.

Rocha *et al.* [24] presented a methodology for the detection of security attacks and the classification of Internet flows that relies on multidimensional Gaussian distributions [25]. In this way, it is possible to account for the correlation between the values that are obtained for the different dimensions, allowing to infer even more accurate probability distributions. The proposed approach starts by performing a multiscale analysis to the sampled IP data-streams, obtaining multiscale estimators for all streams; the estimators are subsequently processed by mapping a dimension to each timescale, so that the multivariate distributions (for each protocol) can be inferred; an algorithm will then find the dimensions where the separation between the several distributions is most noticeable and each of the traffic streams is then classified according to the probability of belonging to each one of the inferred distributions.

III. MULTISCALING ANALYSIS

The inability of conventional Fourier analysis to preserve the time dependence and describe the evolutionary spectral characteristics of non-stationary processes requires tools that allow time and frequency localization. Wavelet transforms can provide information concerning both time and frequency, which allows local, transient or intermittent components to be elucidated [12]. Such components are often obscured due to the averaging inherent within spectral only methods, like Fast Fourier Transform (FFT) [26], for example.

Wavelets are mathematical functions that are used to divide a given signal into its different frequency components. They consist of a short duration wave that has limited energy. Wavelets enable the analysis of each one of the signal components in an appropriate scale. Starting with a mother wavelet $\psi(t)$, a family $\psi_{\tau,s}(t)$ of "wavelet daughters" can be obtained by simply scaling and translating $\psi(t)$:

$$\psi_{\tau,s}(t) = \frac{1}{\sqrt{|s|}}\psi(\frac{t-\tau}{s}) \tag{1}$$

where s is a scaling or dilation factor that controls the width of the wavelet (the factor $\frac{1}{\sqrt{|s|}}$ being introduced to guarantee preservation of the energy, $||\psi_{\tau,s}|| = |\psi|$) and τ is a translation parameter controlling the location of the wavelet. Scaling a wavelet simply means stretching it (if $|s| > 1$) or compressing it (if $|s| < 1$), while translating it simply means shifting its position in time.

Given a signal $x(t) \in L^2(\Re)$ (the set of square integrable functions), its Continuous Wavelet Transform (CWT) with respect to the wavelet ψ is a function of time (τ) and scale (s), $W_{x;\psi}(\tau, s)$, obtained by projecting $x(t)$ onto the wavelet family $\{\psi_{\tau,s}\}$:

$$W_{x;\psi}(\tau, s) = \int_{+\infty}^{-\infty} x(t)\frac{1}{\sqrt{|s|}}\psi(\frac{t-\tau}{s})dt \tag{2}$$

By analogy with the terminology used in the Fourier case, the energy components of the signal are given by the square of the CWT components of the signal and the (local) Wavelet Power Spectrum (sometimes called Scalogram or Wavelet Periodogram) is defined as the normalized energy over time and scales:

$$E_x(\tau, s) = 100\frac{|W_{x;\psi}(\tau, s)|^2}{\sum_{\tau'}\sum_{s'}|W_{x;\psi}(\tau', s')|^2} \tag{3}$$

Scalograms reveal much information about the nature of non-stationary processes that was previously hidden, so they are applied to a lot of different scientific areas: diagnosis of special events in structural behavior during earthquake excitation, ground motion analysis, transient building response to wind storms, analysis of bridge response due to vortex shedding, among others [27].

IV. MULTISCALING BEHAVIOR MODELING

A. Multiscale traffic data

Let us assume that process $x(t)$ represents a counting statistic of a layer 2 traffic trace to and from a specif user terminal (e.g., number of frames on the upload direction, number of bytes in the download direction, etc.). The user is identified by a layer 2 address depending on the underlying communications technology. It is possible to apply a multiscaling analysis to process $x(t)$ by calculating the scalogram using equation (3). We characterize the multiscale user behavior by the estimator of the standard deviation of that user's traffic energy within a time window for a set of timescales. Therefore, a traffic process energy standard deviation at time interval k and time scale s using a sliding time window of width W can be defined as:

$$\hat{D}_x(k, s) = \sqrt{\frac{1}{W-1}\sum_{\tau\in[k-W,k]}\left(E_x(\tau, s) - \overline{E_x(k, s)}\right)^2} \tag{4}$$

with $k = \{W, W+1, W+2, \ldots\}$ and

$$\overline{E_x(k, s)} = \frac{1}{W}\sum_{\tau'\in[k-W,k]} E_x(\tau', s) \tag{5}$$

Choosing J timescales ($\{s_1, s_2, \ldots, s_J\}$) of interest, it is possible to define a vector $B_{x,k}$ that describes the inferred localized multiscaling characteristics (at time interval k) of the traffic process x:

$$B_{x,k} = \{\hat{D}_x(k, s_j), j = 1, \ldots, J\} \tag{6}$$

B. Markov Modulated multivariate Gaussian Processes Model

The proposed discrete time Markov Modulated multivariate Gaussian Process (dMMGP) model characterizes position and mobility of a subject based on the following assumptions: (i) the multiscaling behavioral metrics for the use of a specific web application can be described by a multivariate Gaussian distribution, (ii) the time scales of importance can be pre-determined, (iii) a ground truth for the web applications usage multiscaling characteristics can be pre-established and (iv) the transition between applications can be described by an underlying (homogeneous) Markov chain where each state maps the multiscaling behavior characteristics of a specific web application usage.

The dMMGP can then be described as a J-dimensional random process (B) with a multivariate Gaussian distribution that characterizes the behavior of a user in an universe of A possible applications in a J-dimensional environment (for J time scales of importance), whose parameters are a function of the state (S) of the modulator Markov chain (B, S) with A states. The dMMGP model states will map the applications multiscale characteristics and the dMMGP model transitions will define the user behavior/dynamics on the usage of the different applications. The former will be inferred based on pre-established ground truth (set of known flows) for the web applications multiscaling characteristics and the later will be inferred based on the dynamics of the mapping of a set of flows of specific users to the application multiscale characteristics (i.e. model states).

More precisely, the (homogeneous) Markov chain

$$(B, S) = \{(B_k, S_k),\, k = 0, 1, \ldots\}$$

with state space $I\!R^J \times U$, with $U = \{1, 2, \ldots, A+1\}$, is a dMMGP if and only if for $k = 0, 1, \ldots$,

$$P(B_{k+1} = \mathbf{b},\, S_{k+1} = n | S_k = m) = p_{mn}\Gamma_n(\mathbf{b}) \qquad (7)$$

where $\mathbf{b} \in I\!R^J$ is a generic multiscale component in a J-dimensional environment, p_{mn} represents the probability of a transition from state m to state n of the underlying Markov chain in time interval $[k, k+1]$, and

$$\Gamma_a(\mathbf{b}) = (2\pi)^{-\frac{J}{2}} \mathbf{\Sigma}_a^{-\frac{1}{2}} e^{-\frac{1}{2}(\mathbf{b}-\mathbf{m}_a)^T \Sigma_a^{-1}(\mathbf{b}-\mathbf{m}_a)} \qquad (8)$$

is the multivariate Gaussian distribution of the multiscaling characteristics of application a flows, it is centered in m_a and has covariance matrix Σ_a.

Whenever (7) holds, we say that (B, S) is a dMMGP with a set of modulating states with size A and parameter matrices $\mathbf{P}$, $\mathbf{M}$ and $\mathbf{S}$. Matrix $\mathbf{P}$ is the transition probability matrix of the modulating Markov chain S,

$$\mathbf{P} = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1A} \\ p_{21} & p_{22} & \cdots & p_{2A} \\ \cdots & \cdots & \cdots & \cdots \\ p_{A1} & p_{A2} & \cdots & p_{AA} \end{bmatrix} \qquad (9)$$

while matrix $\mathbf{M}$ defines the mean values of each multiscaling Gaussian distribution:

$$\mathbf{M} = \begin{bmatrix} \mathbf{m}_1 & \mathbf{m}_2 & \ldots & \mathbf{m}_A \end{bmatrix} \qquad (10)$$

where $\mathbf{m}_a$ is a $J \times 1$ vector. Matrix $\mathbf{S}$ contains the covariance (sub-)matrices of each multiscaling Gaussian distribution:

$$\mathbf{S} = \begin{bmatrix} \mathbf{\Sigma}_1 & \mathbf{\Sigma}_2 & \ldots & \mathbf{\Sigma}_A \end{bmatrix} \qquad (11)$$

where $\mathbf{\Sigma}_a$ is a $J \times J$ matrix. Moreover, we denote by $\mathbf{\Pi} = [\pi_1, \pi_2, \ldots, \pi_A]$ the stationary distribution of the underlying Markov chain.

Matrix $\mathbf{P}$ will be unique for each user, and will characterize his/her behavior on the usage of the applications characterized by matrices $\mathbf{M}$ and $\mathbf{S}$. The overall multiscaling behavior of a user can be statistically described by a stationary probability density defined by a weighted sum of A multivariate Gaussian distributions:

$$f(\mathbf{b}) = \sum_{a=1}^{A} \pi_a \Gamma_a(\mathbf{b}), \mathbf{b} \in I\!R^J \qquad (12)$$

where $\mathbf{b}$ is a multiscale component that belongs to the J-dimensional domain of chosen timescales.

C. Model Inference Procedure

Assuming that we have a ground-truth for a set of A web applications, analyzed over F flows, over K time windows in J timescales of interest, we can define the multiscale profile of an application $a(a = 1, \ldots, A)$ as $G_{a,f,k}$, inferred using equation (6) considering that process $x(t)$ is the f-th flow of application a, with $a = 1, \ldots, A$, $f = 1, \ldots, F$ and $k = 1, \ldots, K$, i.e.:

$$G_{a,f,k} = B_{x,k}, x \leftrightarrow \text{flow } f \text{of application } a \qquad (13)$$

The $\mathbf{M}$ and $\mathbf{S}$ matrices of the dMMPGP model can then be inferred as

$$\mathbf{m}_a = \frac{1}{KF} \sum_{f=1}^{F} \sum_{k=1}^{K} G_{a,f,k} \qquad (14)$$

$$\mathbf{\Sigma}_a = \frac{1}{KF-1} \sum_{f=1}^{F} \sum_{k=1}^{K} \Big((G_{a,f,k} - \mathbf{m}_a)\,(G_{a,f,k} - \mathbf{m}_a)^T \Big) \qquad (15)$$

The final step of the inference procedure is to infer matrix $\mathbf{P}$, i.e. the transition probabilities between the states defined in the first step. This task is achieved by probabilistically mapping each multiscaling behavior of each unknown flow trace $x(t)$ $B_{x,k}, k = 1, \ldots, K$ to one state/application and then averaging the probabilistic transitions between states, according to a probability vector:

$$\mathbf{q}_k = \{\Gamma_1(B_{x,k}), \ldots, \Gamma_A(B_{x,k})\}, k = 0, 1, \ldots, K \qquad (16)$$

D. Behavior Prediction

Defining $\mathbf{c}_k = \{c_{k,a} : a = 0, 1, \ldots, A\}, k = 0, 1, \ldots, K$, where $\mathbf{c}_k$ is the probability vector defining that within time-window $[k-W, k]$ the user is using application a, and based on equation (12) we can define the multivariate distribution of the predicted multiscaling behavior of the user in a future time-window (z observations in the future) as:

$$\sum_{a=1}^{A} \mathbf{c}_{k+z} \Gamma_a \qquad (17)$$

with

$$\mathbf{c}_{k+z} = \mathbf{c}_k \mathbf{P}^z \qquad (18)$$

where $\mathbf{c}_{k+z}$ represents the probabilistic vector that quantifies the probability of a web application to be in use k time windows in the future.

V. Proof of concept

A. Data-set

The test data-set was obtained by capturing, in promiscuous mode, the layer 2 traffic having as source or destination a specific Wi-Fi network access point. The traffic capture was performed without authenticating to the network and consisted only of 802.11 frames. In a controlled environment, where all terminals were using a bare installation of Linux with a daemon that recorded all browse requests, a set of invited users were asked to access and use their usual web applications, maintaining their typical behavior. This approach allowed us to create the ground-truth of a mapping between layer 2 data traces and their originating users and web applications. Within the context of this paper and this proof of concept, we only used the data traces that were created by users accessing three general web applications: social networking, namely Facebook (without chatting and game interactions), news web journals and web-mail access. The total number of data sets was divided in two: the first half was used to infer the underlying dMMGP model of the behavior of each application and user, while the second half of the data sets was used to validate the inferred models by comparing the predicted multiscale behavior (and associated web application usage sequence) of each user. The raw statistical process used was the amount of bytes transmitted from the Wi-Fi access point to each user, sampled every 0.1 seconds. Sampling the raw statistics in 0.1 seconds allows our method to measure and incorporate some of the most characteristic multiscale dynamics of an application: (i) the lower timescales that are strictly related with the way that specific application handles the multiple data sessions, (ii) the medium timescales that are related with the application algorithmic dynamics and (iii) the higher timescales that reflect mainly the user interactions dynamics [28]. For the purpose of the model inference, we use time windows with a width of 120 seconds ($W = 1200$) and considered time windows in 20 seconds interval. The choice of these values is a tradeoff between the amount of (past) data necessary to fully characterize the traffic dynamics and the amount of data that can be process and analyzed in pseudo-real time. The heavier computational task that it is the construction and update of the behavior models which are made off-line and is not an issue. However, to perform the application and user identification the measured data must be matched with previously inferred models in pseudo real-time. The interval between windows of classification was chosen in order to minimize the delay between the moment of an user application change and its effective detection by our methodology. With an appropriate choice of parameters, namely window size and interval of processing, this methodology is fully scalable since the computation power required is proportional to the amount of traffic (number of users) under analysis.

Figures 1 and 2 depicted the 80% and 90% quantile frontiers of the inferred multivariate Gaussian distributions of the multidimensional characteristics of each application (using just 3 timescales) for all users. These distributions reveal that the multiscale characteristics of the three web applications are distinct and have a small overlap in the universe of the three dimensions/scales considered.

After inferring the underlying dMMGP model, we use the test data traces to test the precision of the model in identifying the current web application of an user every 20 seconds. In this test, we were able to obtain a precision of 72.4% of correctly classified windows and the identification results presented in Table V-A. The results show a very good agreement between the identified web application and the real application, considering the reduced amount of information (in terms of raw data and time span of the observation) used for the identification.

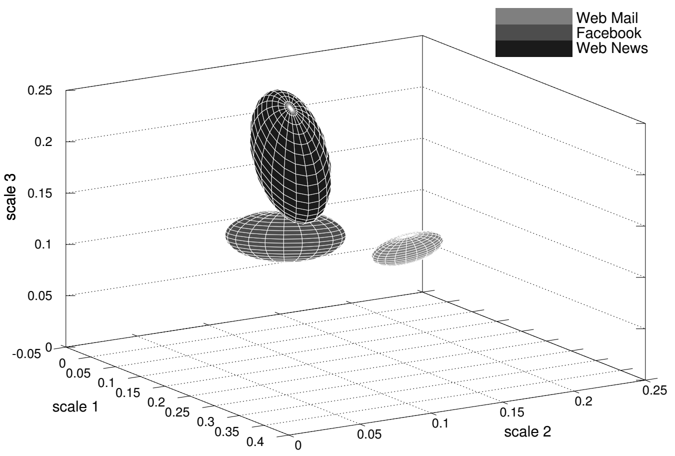


Figure 1. 80% quantile frontiers of the inferred multivariate Gaussian distributions of the multidimensional characteristics of each application.

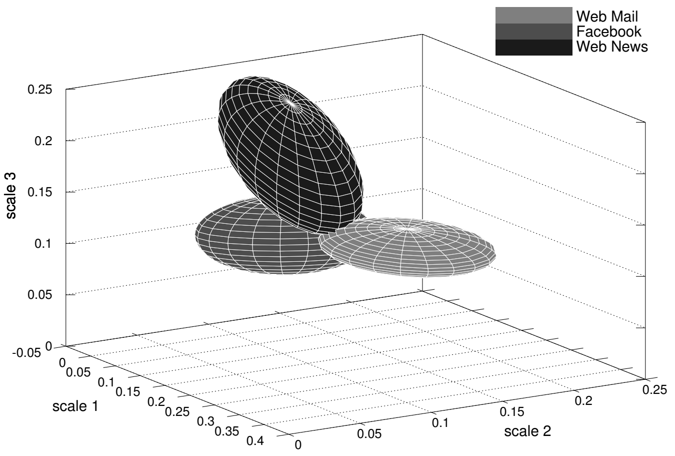


Figure 2. 90% quantile frontiers of the inferred multivariate Gaussian distributions of the multidimensional characteristics of each application.

Using the test data traces to test the precision of the model in identifying the web applications that are in use 60 seconds in the future we obtained a precision of 55.3% of correctly classified windows. The results show that the identification/predicting results are still significantly above the pure random guess.

The results show that our methodology was able to obtain very good classification and prediction results considering the reduced amount of information (only network layer 2 sampled statistics) and that the web applications under consideration may, in some particular cases, be very similar. Most of the

	Web-Mail	Facebook	Web-news
Web-mail	69.95%	7.84%	22.20%
Facebook	4.12%	83.13%	12.74%
Web-news	7.08%	24.35%	68.55%

Table I
IDENTIFICATION OF THE CURRENT WEB APPLICATION RESULTS.

errors can be explained by the fact that some Web-news pages are very similar to social networking applications pages and even incorporate social network features within its own Web-pages. Also, when the Web-news web pages have less content the user dynamics may get similar to Web-mail or Facebook interactions (i.e., small data chunks exchanged at small intervals).

VI. Conclusion and Future Work

We presented a novel approach that uses multiscaling traffic characteristics to differentiate between different web applications and a Markovian model that is able to characterize the dynamics of user actions over time. By applying this methodology to Wi-Fi layer 2 traffic generated by users accessing different common web services/contents through HTTP (namely, social networking, web news and web-mail applications), it was possible to achieve a good matching and prediction of the users behaviors. Our methodology may be applied to preallocate resources in network access points based on past user behavior and pseudo real-time predictions of short term requirements.

As future work, we plan to test our methodology incorporating more applications with completely different behavior (such as video streaming, P2P file transferring, online games, etc.). This will required the improvement of the inner algorithms of the methodology to accommodate multiple and dynamic timescales ranges. Moreover, our short term plans include the developing of a prototype and test in a 3G/4G network base station for optimal dynamic allocation of resources.

References

[1] P. Salvador, R. Valadas, and A. Pacheco, “Multiscale fitting procedure using Markov modulated poisson processes,” *Telecommunication Systems Journal, Kluwer Academic Publishers*, vol. 23, no. 1-2, pp. 123–148, 2003.

[2] A. Pacheco, L. C. Tang, and N. U. Prabhu, *Markov-modulated processes & semiregenerative phenomena*. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ, 2009.

[3] “IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)*, pp. 1 –1076, 12 2007.

[4] A. Ghosh, D. Wolter, J. Andrews, and R. Chen, “Broadband wireless access with WiMax/802.16: current performance benchmarks and future potential,” *Communications Magazine, IEEE*, vol. 43, no. 2, pp. 129 – 136, feb. 2005.

[5] M. van Nielen, “UMTS: a third generation mobile system,” in *Personal, Indoor and Mobile Radio Communications, 1992. Proceedings, PIMRC ’92., Third IEEE International Symposium on*, oct 1992, pp. 17 –21.

[6] E. Dahlman, B. Gudmundson, M. Nilsson, and A. Skold, “UMTS/IMT-2000 based on wideband CDMA,” *Communications Magazine, IEEE*, vol. 36, no. 9, pp. 70 –80, sep 1998.

[7] D. Astely, E. Dahlman, A. Furuskar, Y. Jading, M. Lindstrom, and S. Parkvall, “Lte: the evolution of mobile broadband,” *Communications Magazine, IEEE*, vol. 47, no. 4, pp. 44 –51, april 2009.

[8] H. Kim, K. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, “Internet traffic classification demystified: myths, caveats, and the best practices,” in *Proceedings of the 2008 ACM CoNEXT Conference*, ser. CoNEXT ’08. New York, NY, USA: ACM, 2008, pp. 11:1–11:12. [Online]. Available: http://doi.acm.org/10.1145/1544012.1544023

[9] S. Sen, O. Spatscheck, and D. Wang, “Accurate, scalable in-network identification of p2p traffic using application signatures,” in *Proceedings of the 13th international conference on World Wide Web*, ser. WWW ’04. New York, NY, USA: ACM, 2004, pp. 512–521. [Online]. Available: http://doi.acm.org/10.1145/988672.988742

[10] A. Madhukar and C. L. Williamson, “A longitudinal study of p2p traffic classification,” in *Proceedings of the IEEE MASCOTS*, 2006, pp. 179–188.

[11] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, “BLINC: multilevel traffic classification in the dark,” in *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, ser. SIGCOMM ’05. New York, NY, USA: ACM, 2005, pp. 229–240. [Online]. Available: http://doi.acm.org/10.1145/1080091.1080119

[12] J. Byrnes, K. A. Hargreaves, and K. Berry, *Wavelets and their Applications*. Springer, 1994.

[13] V. Paxson, “Empirically derived analytic models of wide-area tcp connections,” *IEEE/ACM Trans. Netw.*, vol. 2, pp. 316–336, August 1994. [Online]. Available: http://dx.doi.org/10.1109/90.330413

[14] C. Dewes, A. Wichmann, and A. Feldmann, “An analysis of internet chat systems,” in *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, ser. IMC ’03. New York, NY, USA: ACM, 2003, pp. 51–64. [Online]. Available: http://doi.acm.org/10.1145/948205.948214

[15] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, “Flow clustering using machine learning techniques,” *LNCS*, vol. 3015, pp. 205–214, 2004.

[16] M. Roughan, S. Sen, O. Spatscheck, and N. G. Duffield, “Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification.” in *Internet Measurement Conference’04*, 2004, pp. 135–148.

[17] S. Zander, T. T. T. Nguyen, and G. J. Armitage, “Automated traffic classification and application identification using machine learning.” in *LCN’05*, 2005, pp. 250–257.

[18] A. Moore and D. Zuev, “Internet traffic classification using bayesian analysis techniques.” in *ACM SIGMETRICS*, 2005, pp. 50–60.

[19] L. Bernaille, R. Teixeira, and K. Salamatian, “Early application identification,” in *Proceedings of the 2006 ACM CoNEXT conference*, ser. CoNEXT ’06. New York, NY, USA: ACM, 2006, pp. 6:1–6:12. [Online]. Available: http://doi.acm.org/10.1145/1368436.1368445

[20] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, “Traffic classification through simple statistical fingerprinting,” *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 5–16, January 2007. [Online]. Available: http://doi.acm.org/10.1145/1198255.1198257

[21] A. Este, F. Gringoli, and L. Salgarelli, “Support vector machines for tcp traffic classification,” *Comput. Netw.*, vol. 53, pp. 2476–2490, September 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1576850.1576885

[22] T. T. T. Nguyen and G. J. Armitage, “A survey of techniques for internet traffic classification using machine learning.” *IEEE Communications Surveys and Tutorials*, pp. 56–76, 2008.

[23] R. Bar-Yanai, M. Langberg, D. Peleg, and L. Roditty, “Realtime classification for encrypted traffic.” in *SEA’10*, 2010, pp. 373–385.

[24] E. Rocha, P. Salvador, and A. Nogueira, “Detection of illicit network activities based on multivariate gaussian fitting of multi-scale traffic characteristics,” in *2011 IEEE International Conference on Communications (ICC 2011)*, June 2011, pp. 1–6.

[25] K. S. Miller, *Multidimensional Gaussian Distributions*. John Wiley & Sons Inc, 1964.

[26] E. Brigham, *Fast Fourier Transform and Its Applications*. Prentice Hall, 1988.

[27] K. Gurley and A. Kareem, “Applications of wavelet transforms in earthquake, wind, and ocean engineering,” *Engineering Structures*, no. 21, pp. 149–167, 1999.

[28] M. Crovella and B. Krishnamurthy, *Internet Measurement: Infrastructure, Traffic and Applications*. Wiley, 2006.

An Empirical Study of Connections Between Measurements and Information Security

Rodrigo Sanches Miani*, Michel Cukier†, Bruno Bogaz Zarpelão*, Gean Davis Breda*, Leonardo de Souza Mendes*
*School of Electrical and Computer Engineering
University of Campinas, Campinas, SP, Brazil
Email: {rsmiani,bzarpe,gean,lmendes}@decom.fee.unicamp.br

†A. James Clark School of Engineering
University of Maryland, College Park, USA
Email: mcukier@umd.edu

***Abstract*—This paper presents an investigation of factors that are likely to affect the security of an organization, in particular, the number of security incidents. Using Intrusion Prevention Systems (IPS) data, provided by the University of Maryland, we derive three potential factors (attackers, corrupted computers and attack types) and their respective measurements. Based on empirical studies and information security literature, we examine the effects of selected factors on the number of security incidents. We use a regression model to test the hypotheses empirically and also to study how those factors are affected over time. We found that the number of potential corrupted computers is positively related to the security incidents while the number of potential attackers and range of attack types does not significantly affect the number of security incidents. We also found empirical evidence that factors could significantly change over time.**

***Keywords*-Network and Security Management; Security Metrics; Empirical Study; Security Incidents; Intrusion Prevention Systems.**

I. Introduction

In information security, questions such as “Is security improving over time?” and “Are we using effective controls?” could be used to derive measurements to facilitate decision making and improve performance and accountability. Researchers have been trying to measure security using models from other disciplines such as economics, risk, reliability engineering and statistics. However, as pointed out by Jansen [1] much of what has been written about security quantification is definitional, aimed at providing guidelines for defining a security metric and specifying criteria for which to achieve.

Currently, there are many suggestions in the security community for what measures organizations should collect in order to construct security measurement models [2], [3] and [4]. However, as noted by Verendel [5], for most cases it is unknown if the proposed models are valid or not in representing security for systems in realistic environments due to the lack of validation and comparison between such methods against empirical data. In other words, little work has been done to determine the value of these measures in real-world operational environments [6]. Research is needed to validate connections between measures and security, and determine and understand possible correlations.

In this paper, we study connections between metrics derived from intrusion prevention system (IPS) alert events and the number of security incidents. The number of security incidents is an important security indicator that in combination with other metrics can indicate the level of threats and effectiveness of security controls.

Since security incidents may be the result of several factors, from a computer infected with a virus to successful attacks against servers, our aim is to investigate some of these factors that could be derived from Intrusion Prevention Systems (IPS) data. We empirically examine, using a multiple linear regression model, the effects of potential attackers, potential corrupted computers and attack signatures on the number of security incidents reported by the Office of Information Technology (OIT) at the University of Maryland [7]. We also analyze how those factors are affected over time.

This paper is structured as follows. Section 2 describes the background on security incidents and intrusion prevention systems. Section 3 introduces our hypotheses about the relationship between factors and the number of security incidents. Section 4 presents the empirical modeling approach and the regression results. Section 5 discusses the threats to validity of our study. We provide conclusions and directions for future work in Section 6.

II. Background

This section describes the background on security incidents, intrusion prevention systems and security quantification that we will use in this paper.

A. Security Incidents

A security incident, according to the Center for Internet Security [2], results in the actual outcomes of a business process deviating from expected outcomes for confidentiality, integrity, and availability resulting from people, process, or technology deficiencies or failures. There are many interacting factors that affect the occurrence of a computer security incident, for instance, virus, vulnerabilities, characteristics of the population being attacked, distribution and prevalence of vulnerable operating systems and applications and intensity of attacks [8], [9]. Examining the factors that could lead to security incidents is important to improve forecasting and also to identify conditions which may result in the spread of new incident types.

Security incidents may be reported using operational security systems sources, such as anti-malware software and intrusion detection systems (IDS), host logs and also reports from users. In this work we are interested in studying the number of security incidents.

The number of security incidents indicates the number of detected security incidents that the organization has experienced during a time period. However, the use of this metric should be properly examined to avoid misinterpretation of data. In combination with other metrics, the number of security incidents can indicate the level of threats, incident detection capabilities and effectiveness of security controls and can be used as a security indicator of an organization. Understanding which factors are likely to affect the number of security incidents could begin to paint a picture of system security.

B. Intrusion Prevention Systems

An IPS is considered as an extension of an IDS that monitors malicious activity and reacts in real time by blocking a potential attack [10]. An IDS is a passive device that monitors activity whereas an IPS is an active device that blocks the potential malicious activity.

The investigated IPS device is a signature-based IPS where the blocking decision relies on a set of signatures that are regularly released by the vendor as attacks are newly discovered on the Internet. Basically, when the characteristics of an attack match the ones of a defined signature, the attack is blocked and an alert is recorded.

Based on the data provided by the IPS, it is possible to derive metrics to assess the volume and the nature of the malicious activity. For instance, a set of metrics that includes the number of alerts, number of distinct targets, number of distinct IPS signatures and number of blocked attackers was proposed by [11].

C. Security Quantification

The ISO/IEC 27004 [12] standard includes specific guidelines about information security measurement such as: measures and measurement development, measurement operation, data analysis and measurement results reporting, and also a template on which to describe a metric.

Jansen [1] provides an overview of the security metrics area and look at possible avenues of research that could be pursued to advance the state of the art. The author states that much of what has been written about security metrics is definitional, aimed at providing guidelines for defining a security metric and specifying criteria for which to achieve. However, relatively little has been reported on actual metrics that have been proven useful in practice.

Condon et al. [8], [9] describe the application of time series models and software reliability models on computer security incidents. They found that certain incidents are caused by well-defined vulnerabilities and might easily patched against but others may exhibit propagation behavior similar to contagious diseases in animals or still can result from economic incentives external to an organization or environment.

Considering other empirical analysis about security measurement, Chrun et al. [11] presents a method that ranks potentially corrupted computers using security metrics derived from imperfect IPS event data. However, nothing was said about the relationship between corrupted computers and computer security incidents.

Cukier and Panjwani [13] conducted an empirical analysis to quantify the link between vulnerabilities and malicious connections. They conclude that a high number of vulnerabilities on services do not necessarily imply a high number of malicious connections or successful attacks.

This study can be used as motivation to investigate another relevant security issues such as: does the high number of attacks on networks imply a high number of security incidents? Does the high number of corrupted computers on networks imply a high number of security incidents?

Our work focuses on extracting empirical relationships between IPSs and computer security incidents datasets and also how to use these results to improve the knowledge about system security.

III. Research Hypotheses

In this section, we investigate the relationship between the selected factors and their impact on the number of computer security incidents. From this investigation, we formulate hypotheses to guide us in the study.

A. Attackers

From models that predict the likelihood of burglary or other conventional crimes, it has been demonstrated that parameters such as a motivated offender, a suitable victim and the absence of a motivated guardian can have a major effect on the probability of crime [14].

In information security, the same analogy is hard to achieve. According to Schechter [15], the number of potential attackers is likely to be positively correlated with the rate of security breaches and the resulting security risk. In addition to that, certain variables such as means, motive and opportunity to attack, may also influences the number of potential attackers.

As pointed out by Chrun [16], the three major attacker's characteristics which influence on security are motivation, qualities and expertise. For this reason, it is difficult to predict that the number of attackers increases the number of incidents, for example. Besides, an attack could be initiated from an external or internal source and insiders do not generally demonstrate the same attack pattern that external attackers do [17].

With this in mind, we propose the following alternate pairs of exploratory hypotheses: i) H1(a) The number of attackers increases the number of security incidents and ii) H1(b) The number of attackers does not increase the number of security incidents.

B. Attack signatures

According to Chrun et al. [10], analyzing the range of attack types that target an organization might help the security team identifying popular attack types and making decisions for their network security. With this in mind, it is reasonable to assume that the number of attack signatures is a factor that might be linked to the number of security incidents.

In the simplest case, we can say that an increase in the number of attack signatures might be linked to the number of incidents due to the higher number of attack types that should be handled by the organization. We can also consider the severity of attack signatures. IPS vendors usually classify each attack signature based on the impact of attack on the network. More severe attacks could represent more security incidents. However, attack signatures may have limited impact on the number of security incidents due to the way that organizations react to different attack types. In other words it is difficult to predict that the number of attack signatures increases the number of incidents, for example.

Therefore, we propose the following alternate pairs of hypotheses, also considering the severity of signatures: i) H2(a) The number of attack signatures increases the number of security incidents, ii) H2(b) The number of attack signatures does not increase the number of security incidents, iii) H3(a) The severity of attack signatures increases the number of security incidents and iv) H3(b) The severity of attack signatures does not increase the number of security incidents.

C. Corrupted computers

A corrupted computer is a potential source of attack inside an organization. Malwares, computer viruses, worms and even unpatched applications are some feasible causes for computer corruption.

The relationship between corrupted computers and security incidents could be depicted using computer security data breach reports. A study conducted by Verizon [18] using 141 breach cases, revealed that 38% of studied security incidents were caused by a computer corrupted by malware. Other study conducted by PricewaterhouseCoopers [19] using the survey data of 539 respondent organizations showed that 14% of incidents were caused by viruses or malicious software corruption. Besides that, 23% was caused by systems failure or data corruption which could also involve corrupted computers.

A similar survey [20] based on the responses of 351 computer security practitioners in U.S. corporations, government agencies, financial institutions, medical institutions and universities, revealed that 67.1% of studied incidents were provoked by malware corruption. Despite the differences between the studies, the rate at which corrupted computers affected the investigated attacks could be considered significant.

In face of these findings we propose the following hypothesis: H4(a) The number of corrupted computers increases the number of security incidents.

IV. Empirical Modeling Approach

This section presents the description of the dataset and our empirical modeling approach.

A. Data and measurements

The dataset provided by the University of Maryland consisted of over 2615 security incidents and 6.687.874.770 IPS alerts recorded during a period of four years and four months (from September 4, 2006 to December 31, 2010).

The data were grouped in a weekly basis ($t = 226$ weeks), with Monday as the first day of the week. Grouping the data in a time window is a technique to minimize the effects of lag time between occurrence of an event and submission of an incident report. However, due to the human interaction in the incident reporting process, we cannot prove that an incident reported in certain week would have been related to IPS alerts from the same week.

The incidents dataset also included 21 different incident types. Only one type called "nethicsreq" was removed from the dataset. It represents the identification of an illegal use of copyrighted material like music and movies. This kind of incident cannot be detected by the IPS and, therefore, is out of scope of our investigation.

The IPS is located at the edge of the organization so it cannot detect traffic originating inside the organization and targeting computers inside the organization. This dataset includes two cases, represented by Figure 1: 1) where a computer outside the organization is targeting the organization, and 2) where a computer inside the organization is targeting computers outside the organization.

Chrun et al. [11] proposed metrics for both cases. When the organization is the target (case 1), the following metrics were proposed: number of alerts, number of distinct targets,

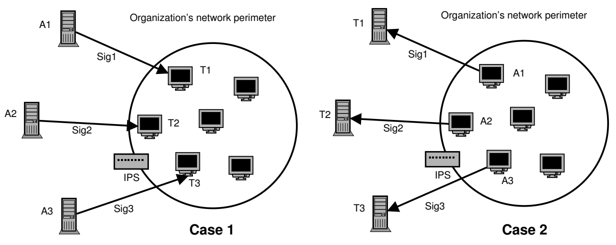


Figure 1. IPS location and attacker's perspective

number of distinct IPS signatures, number of alerts per target and number of attackers per target. When focusing on the traffic originating inside the organization and targeting computers outside the organization (case 2), the following metrics were proposed: number of alerts, number of distinct attackers, number of distinct IPS signatures, number of alerts per attacker and number of targets per attackers. We derive the investigated factors (attackers, corrupted computers and attack signatures) from these two sets of metrics.

We derive the number of security incidents (denoted as I_t, $t = 1, \dots, 226$) from the provided incidents dataset and the number of potential attackers, attack signatures and potential corrupted computers from the provided IPS alerts database.

We consider the number of potential attackers in this study as the number of distinct external blocked attackers (denoted as A_t, $t = 1, \dots, 226$) in the IPS as showed in case 1.

The number of attack signatures is extracted using two measures, one from case 1, the number of distinct signatures from external attackers (denoted as Ve_t, $t = 1, \dots, 226$), and one from case 2, the number of distinct signatures from internal attackers (denoted as Si_t, $t = 1, \dots, 226$). In this way it is possible to estimate the attack types associated with computers that are being exploited by external attackers and also the attack types associated with computers that are being used to launch attacks against external targets. The same applies for the severity of a attack signature where the severity level of a signature is classified by the vendor in three levels: critical (denoted as SCe_t and SCi_t), major (denoted as SMe_t and SMi_t) and minor (denoted as SNe_t and SNi_t).

The number of potential corrupted computers could be measured using two metrics from the IPS dataset: number of distinct targets in case 1 (denoted as Ce_t, $t = 1, \dots, 226$) and the number of distinct internal blocked attackers, showed in case 2 (denoted as Ci_t, $t = 1, \dots, 226$).

In case 1, it is clear that the number of distinct targets reflects the number of targeted computers and thus potentially corrupted computers. The number of distinct internal blocked attackers in case 2 could be seen in two different ways: if a computer inside the organization is launching an attack, it might the result of a willing attacker who launches an attack, or it may be due to an already corrupted computer launching attacks. Therefore, it is reasonable to assume that the number of external targets and the number of internal attackers could be used to approximate the number of corrupted computers on a network. Table I presents the descriptive statistics of these variables.

B. Empirical Modeling

In order to investigate how the variables defined in the previous section might be linked to the number of security incidents, we built a multiple linear regression model. Our dependent variable is the weekly number of incidents I_t and the independent variables are A_t, Ce_t, Ci_t, Se_t, SCe_t, SMe_t, SNe_t, Si_t, SCi_t, SMi_t and SNi_t. The general notation of the multiple regression functions can be written as:

$$I_t = \beta_0 + \sum_{k=1}^{j} \beta_k X_k + \epsilon_t \qquad (1)$$

where β_0 is the constant term, β_1 to β_j are the coefficients on the j^{th} independent variable, j is the total number of independent variables, ϵ_t the error term and X_k represents the set of independent variables that could include A_t, Ce_t, Ci_t, Se_t, SCe_t, SMe_t, SNe_t, Si_t, SCi_t, SMi_t, SNi_t. In a multiple linear regression model, we assume that the relationship between the variables is linear and the error terms are distributed normally.

With a multiple linear regression model, it is possible to detect the effect of the independent variables on the dependent variable using a variable selection approach, that is, the screening of the candidate variables to obtain a regression model that contains the 'best" subset of independent variables [21]. The main idea is to select the independent variables, run the regression model and study its significance through the p-value obtained in each variable. Further information about variable selection in multiple linear regression models can be found in [22] and [23].

Our goal is to investigate the effects of attackers, attack signatures and corrupted computers on the number of security incidents using several variables. Therefore, we propose five different regression models in order to investigate the differences between using the number of attack signatures or the severity of attack signatures, and also analyze the impact of each single severity level.

In the first model, we analyze the number of attackers, attack signatures and corrupted computers. In the second model we exclude the number of signatures, in order to analyze the signature severity levels. In models 3, 4, 5 we analyze the impact of each severity level. Model 3 includes only the critical attack signatures, model 4 includes only the major attack signatures and the model 5 includes only the minor attack signatures. Table II summarizes the regression models and results.

We noticed that in all proposed regression models, the weekly number of attackers, A_t, cannot significantly affect

Table I
DESCRIPTIVE STATISTICS OF VARIABLES (226 OBSERVATIONS)

Variables	Mean	Std. Dev	Min	1st Quartile	Median	3rd Quartile	Max
I_t Incidents	9.80	12.236	0	3.00	6.00	11.25	64
A_t External attackers	8674.53	57686.163	31	795.00	1424.00	2193.50	822698
Ce_t External targets	1532.70	767.803	78	1030.50	1262.00	1801.50	6244
Ci_t Internal attackers	103.87	76.991	11	56.00	83.50	131.25	504
Ve_t Ext_Signatures	86.40	124.119	15	62.00	72.00	82.25	1347
SCe_t Ext_Signatures_critical	58.25	81.797	10	43.00	50.00	56.00	1074
SMe_t Ext_Signatures_major	14.21	15.266	2	7.00	10.00	17.00	151
SNe_t Ext_Signatures_minor	14.04	36.583	0	8.00	11.00	14.00	535
Si_t Int_Signatures	29.00	32.132	8	20.00	24.00	29.25	426
SCi_t Int_Signatures_critical	16.22	5.794	4	12.00	15.00	19.00	40
SMi_t Int_Signatures_major	7.38	27.896	0	2.00	3.00	5.00	377
SNi_t Int_Signatures_minor	5.40	3.810	0	4.00	5.00	6.00	39

Table II
REGRESSION RESULTS - ENTIRE DATASET

Variables	Model 1	Model 2	Model 3	Model 4	Model 5
A_t	-0.00001 (0.000014)	-0.00001 (0.000014)	-0.00001 (0.000014)	-0.00001 (0.000014)	-0.00001 (0.000014)
Ce_t	0.002650* (0.00106)	0.00259* (0.01257)	0.00273* (0.001)	0.00266* (0.001)	0.002698* (0.001)
Ci_t	0.022947* (0.01083)	0.0157 (0.01257)	0.0121 (0.0117)	0.02394* (0.01)	0.0256* (0.0111)
Se_t	-0.00717 (0.00657)	-	-	-	-
SCe_t	-	-0.01432 (0.0185)	-0.00939 (0.0098)	-	-
SMe_t	-	0.075 (0.1162)	-	-0.03446 (0.055)	-
SNe_t	-	-0.0211 (0.03569)	-	-	-0.0254 (0.0222)
Sit	0.006619 (0.02578)	-	-	-	-
SCi_t	-	0.3259* (0.1634)	0.3363* (0.1564)	-	-
$VMit$	-	-0.00023 (0.036)	-	-0.0003 (0.0292)	-
$VNit$	-	-0.16 (0.27)	-	-	-0.1378 (0.2227)

* coefficient significant at 0.05	** coefficient significant at 0.01	( ): standard error

the number of security incidents, supporting H1(b). According to our previous discussion, attacker's characteristics such as motivation, expertise and qualities influence the number of potential attackers. Our results suggest that we cannot use the number of attackers as an indicator of security incidents. Besides that, we may also need to develop metrics to help characterize the attacker in order to refine the attacker's dataset. It may be necessary to study characteristics such as attacker geographical origin, time of attack, targets and type of attack to create a subset of possible relevant attackers.

According to Table II, the Model 1 shows that the number of attack signatures, represented by Si_t and Se_t, cannot significantly affect the number of security incidents, supporting H2(b) and also consistent with the results presented in [13]. While the number of attack signatures may be used to find corrupted computers [10] and reveals the range of attack types that target the organization, they are not related to the number of security incidents.

This fact can be due to the limitation of signature-based devices. The detection of new attacks in such devices depend on how fast the vendor will provide new signatures for them. In other words, new attacks will not be detected nor blocked and such unblocked attacks may be the origin of some security incidents, decreasing the impact of attack signatures on security incidents.

Regarding the severity of signatures, none of the distinct signatures from external attackers (SCe_t, SMe_t, SNe_t) significantly affect the number of security incidents.

The same thing occur with the distinct signatures from internal attackers (SMi_t, SNi_t), except with the critical signatures from internal attackers, SCi_t, that significantly affect the number of security incidents, as seen in models 2 and 3. Thus, we conclude that only critical signatures associated with computers from the organization that are being used to launch attacks against external attackers impact the number of security incidents, supporting H3(a), that is, the severity of attack signatures increases the number of security incidents.

Across all these five models, we found that the number of external targets, Ce_t, significantly impacts the number of security incidents. The other metric related to corrupted computers, number of internal attackers, Ci_t, significantly impacts the number of security incidents in models 1, 4 and 5, according to the severity of attack signatures.

In cases where the number of internal critical signatures is considered, the internal attackers are not significantly. Thus, hypothesis H4(a) is supported. In other words, the more corrupted computers, the more frequent the security incidents. Thus, in our case, the number of corrupted computers could be used as an indicator of security incidents.

The results also reveal that the impact of internal threats, such as the number of internal attackers and the number of internal critical signatures (Ci_t and SCi_t), on the number of security incidents are more significant than the external threats, such as external attackers and signatures (A_t, SCe_t, SMe_t, SNe_t). This result may indicate that external threats do not impact the number of security incidents or that the reported security incidents are not reflecting the actual external threats.

Using the same datasets, we can also study the behavior of the factors over time. Since attacks could change over time, it is reasonable to assume that the metrics should also change in order to follow the new trends.

As pointed out by [6], cyber technology is so dynamic that the meaning of metrics changes over time. There may be additional factors that influence the significance of the measure, as well as different relative importance for the existing factors [6]. Therefore, we investigate the variables over two periods of time: 2007-2008 and 2009-2010. We decide to exclude the four months of 2006 in order to preserve the entire year. We also chose the aggregation of two years in each analysis to maintain an acceptable number of observations per dataset (105 per dataset). Tables III and IV summarize the regression results.

Table III
REGRESSION RESULTS - 2007-2008

Variables	Model 1	Model 2	Model 3	Model 4	Model 5
A_t	-0.000005 ($7 \cdot 10^{-7}$)	-0.000005 ($7 \cdot 10^{-7}$)	-0.000005 ($7 \cdot 10^{-7}$)	-0.000005 ($7 \cdot 10^{-7}$)	-0.000005 ($7 \cdot 10^{-7}$)
Ce_t	0.0049** (0.000892)	0.0042** (0.000963)	0.004487** (0.0009)	0.0044** (0.0009)	0.00467** (0.00092)
Ci_t	0.022947 (0.01083)	0.00554 (0.011)	-0.003311 (0.01)	-0.001997 (0.01)	-0.00112 (0.01025)
Se_t	-0.003317 (0.003433)	-	-	-	-
SCe_t	-	-0.02552* (0.0103)	-0.0053 (0.0051)	-	-
SMe_t	-	0.215934** (0.076)	-	0.0012 (0.03)	-
SNe_t	-	-0.039 (0.01962)	-	-	-0.01152 (0.011)
Si_t	0.01318 (0.1048)	-	-	-	-
SCi_t	-	-0.01896 (0.1698)	0.0243 (0.1511)	-	-
SMi_t	-	-0.02117 (0.32511)	-	0.1421 (0.2845)	-
SNi_t	-	-0.36126 (0.3692)	-	-	-0.2217 (0.3711)
	* coefficient significant at 0.05	** coefficient significant at 0.01	( ): standard error		

Analyzing the Tables III and IV, we can note that in both datasets the number of attackers and the number of signatures cannot significantly affect the number of security incidents, the same as in the previous analysis.

Regarding the differences between the two periods, in 2007-2008 the number of external targets, reflecting the potentially number of corrupted computers, significantly impacts the number of security incidents. However, the other metric related to corrupted computers, number of internal attackers, does not significantly affect the number of security incidents as showed in the analysis with the entire dataset.

This metric became significant only in the next period, 2009-2010, showing that the effects of measures changes over time. This may be due to several factors as the increase or decrease of these metrics or changes in the reported type

Table IV
REGRESSION RESULTS - 2009-2010

Variables	Model 1	Model 2	Model 3	Model 4	Model 5
A_t	-0.000006 (0.00009)	0.000022 (0.00009)	0.000016 (0.00009)	0.000005 (0.00009)	0.00001 (0.000014)
Ce_t	-0.0004 (0.0039)	-0.0026 (0.0043)	-0.00085 (0.0039)	-0.00088 (0.00375)	-0.001 (0.004)
Ci_t	0.04247* (0.0146)	0.02879 (0.01743)	0.0349* (0.01659)	0.0442** (0.0141)	0.037* (0.015)
Se_t	-0.04219 (0.07)	-	-	-	-
SCe_t	-	0.08362 (0.1817)	-0.06287 (0.11221)	-	-
SMe_t	-	-0.3715 (0.33279)	-	-0.2019 (0.21867)	-
SNe_t	-	0.0117 (0.618)	-	-	-0.07624 (0.4617)
Si_t	-0.00353 (0.0281)	-	-	-	-
SCi_t	-	0.24 (0.229)	0.1978 (0.22121)	-	-
SMi_t	-	-0.0736 (0.0418)	-	-0.0133 (0.031)	-
SNi_t	-	0.7165* (0.3534)	-	-	0.3471 (0.27187)
	* coefficient significant at 0.05	** coefficient significant at 0.01	( ): standard error		

of security incidents. In other words, the measurement context is very important to improve the accuracy of measures and metrics and also to develop new metrics.

C. Discussions

Currently, there are many recommendations in the security metrics literature for what measures organizations should gather. However, as noted by Black [6], little work has been done to determine the value of these measures in real-world environments, including which measures are most supportive of particular metrics.

Our findings show that, in our case, IPS metrics might be linked to the number of security incidents. Based on the investigated hypothesis the following results were achieved: i) the number of attackers does not increase the number of security incidents, ii) the number of attack signatures does not increase the number of security incidents, iii) the severity of attack signatures increases the number of security incidents and iv) the number of corrupted computers increases the number of security incidents.

The number of attackers, number of attack signatures and severity of external attackers does not significantly affect the number of security incidents. Since there are some characteristics of attackers that might influence the rate at which a system is attacked, finding metrics that characterize attackers is a possible avenue to research, in order to understand the type of attackers that affects the number of security incidents.

A practical implication of the results could be the development of automated tools to analyze the relationship between IPS data and security measurements. Such tools would be used by the security team, as a first step towards understanding the organization's overall security posture. Another implication is the study of IPS metrics to build security incident prediction models. For example, evaluating whether the number of attackers, signatures and corrupted computers are predictive of security incidents. If so, security analysts can use this prediction to prioritize security inspection and to implement preventive measures.

Our study indicates that more than one significant IPS metric might be derived from the dataset. This finding suggests the use of different metrics instead of finding a single security metric. In other words, as noted by [24], since security is a set of attributes, we should also use a set of metrics for measuring it. In combination, metrics could begin to paint a picture of system security.

The results also provide indications that metrics behavior change over time. Given the dynamic nature of information security, it is not certain that events of the past will provide a trustworthy prediction to the future, since attackers actively work to change the threat environment. Therefore, security metrics must be able to reflect significant changes in the underlying assumptions about how the system changes over time.

Our findings might be restricted to networks like those of universities: with nodes that are not fully controlled by the IT department. Private organizations, for instance, have different concerns about information security. The way that the security perimeter of a university is secured is completely different from a private company. Therefore, a similar study, when conducted in such organizations, could show different results.

Finally, we studied the relationship between IPS data and security incidents, but, similar research could be conducted between another security data, for instance, security incidents and firewall logs, intrusion detection systems and network flow data and so on.

V. Threats to validity

The incidents used in our study were reported based on three sources of events: i) an IDS, ii) reports from users and iii) reports from other system administrators. Since recorded incidents led to the blocking of the suspected computer's IP address, the University Office of Information Technology (OIT) verified the authenticity of each incident. As a result, all incidents obtained from these sources were manually reviewed. OIT launched port scans and packet captures to validate the suspicious behavior of identified hosts. Because of the method used by OIT to validate the incidents, we can assume that all incidents used in our work are real. Thus, there are no false positives among the incidents reported. However, we cannot quantify the number of undetected attacks and intrusions that did not lead to a security incident.

The main issue with IPS event data is that the collected data are not perfect [10]. In other words, collected data might contain false positives and might not detect some malicious activity (false negatives). Moreover, since the IPS is a signature-based device, new attacks will not be detected nor blocked.

We have not evaluated the IPS and thus do not know how many false positives and false negatives the IPS produces. Besides, we cannot prove that a blocked attack would have been damaging to the targeted computer. In particular, for an attack to be successful, the targeted computer should have the associated vulnerability. We have scanned several computers for which an IPS alert was raised and noticed that in many cases the vulnerability associated with the alert was not present. This means that even without the IPS, the attack would not have been successful. This also indicates that the IPS identifies and detects an attack in its early stage preferring to block attacks that would not have been successful instead of not blocking a potentially successful attack.

As with all empirical studies, our results are limited to the datasets we investigated. In order to generalize our observations from this study to other environments, further studies should be performed.

VI. Conclusion and Future Work

In this paper, we investigated some factors that might be linked to the rate at which security is successfully breached and empirically examine how attackers, attack signatures and corrupted computers affect the number of security incidents of an organization. We use two datasets, security incidents and IPS alerts, provided by the University of Maryland to derive our measurements.

Our results reveal that from the set of 11 investigated variables, 3 of them are positively related to the number of security incidents: the number of external targets, number of internal attackers, and the number of critical signatures of attacks launched from computers inside the organization. Since the number of external targets and internal attackers are related to the number of potential corrupted computers in an organization, the following hypotheses were supported:

- The number of attackers does not increase the number of security incidents;
- The number of attack signatures does not increase the number of security incidents;
- The severity of attack signatures increases the number of security incidents;
- The number of corrupted computers increases the number of security incidents.

We also found empirical evidence that relevant metrics changes over time. These findings are consistent with the idea that the security of the overall system cannot be ensured using only a single metric. Since security is a set of attributes, measuring security implies the usage of a set of metrics.

Future research should be conducted to compare the results presented in this work. It would be useful to repeat the analysis for some other datasets and investigate the differences between them. For instance, since attack signatures could be associated with certain security vulnerabilities, it would be interesting to investigate whether severity of security vulnerabilities follows the pattern found in our study.

Additional research may be also conducted to evaluate the impact of security factors over other variables, such as network topology and certain security incidents categories. Understanding which factors are likely to affect the security of a network can help network security analysts extract relevant information about the organization security.

ACKNOWLEDGMENTS

The authors would like to thank the State of São Paulo Research Foundation (FAPESP) and the Coordination for the Improvement of Higher Education Staff (CAPES) that supports this work. We also thank Gerry Sneeringer and the Division of Information Technology at the University of Maryland for allowing and supporting the described research.

REFERENCES

[1] W. Jansen, “Directions in security metrics research,” National Institute of Standards and Technology (NIST), Tech. Rep., 2009.

[2] The Center for Internet Security, “The cis security metrics v1.1.0,” http://www.cisecurity.org/, November 2010, retrieved: 07, 2012.

[3] A. Jaquith, *Security metrics: replacing fear, uncertainty, and doubt*. Addison-Wesley Professional, 2007.

[4] M. Swanson, N. Bartol, J. Sabato, J. Hash, and L. Graffo, “Performance measurement guide for information security,” NIST Special Publication 800-55, Tech. Rep., 2003.

[5] V. Verendel, “Quantified security is a weak hypothesis: a critical survey of results and assumptions,” in *NSPW ’09: Proceedings of the 2009 workshop on New security paradigms workshop*. New York, NY, USA: ACM, 2009, pp. 37–50.

[6] P. Black, K. Scarfone, and M. Souppaya, “Cyber security metrics and measures,” 2008.

[7] “Division of information technology,” http://www.oit.umd.edu/, retrieved: 07, 2012.

[8] E. Condon, A. He, and M. Cukier, “Analysis of computer security incident data using time series models,” in *19th International Symposium on Software Reliability Engineering (ISSRE)*, 2008, pp. 77–86.

[9] E. Condon, M. Cukier, and T. He, “Applying software reliability models on security incidents,” in *18th IEEE International Symposium on Software Reliability (ISSRE)*, 2007, pp. 159–168.

[10] D. Chrun, M. Cukier, and G. Sneeringer, “Finding corrupted computers using imperfect intrusion prevention system event data,” in *Proceedings of the 27th international conference on Computer Safety, Reliability, and Security*, ser. SAFECOMP ’08. Springer-Verlag, 2008, pp. 221–234.

[11] ——, “On the use of security metrics based on intrusion prevention system event data: An empirical analysis,” in *HASE ’08: Proceedings of the 2008 11th IEEE High Assurance Systems Engineering Symposium*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 49–58.

[12] “Iso/iec 27004,” http://www.iso27001security.com/, retrieved: 07, 2012.

[13] M. Cukier and S. Panjwani, “Prioritizing vulnerability remediation by determining attacker-targeted vulnerabilities,” *Security & Privacy, IEEE*, vol. 7, no. 1, pp. 42–48, 2009.

[14] E. Mustaine and R. Tewksbury, “Predicting risks of larceny theft victimization: A routine activity analysis using refined lifestyle measures,” *Criminology*, vol. 36, no. 4, pp. 829–858, 1998.

[15] S. Schechter, “Toward econometric models of the security risk from remote attacks,” *Security & Privacy, IEEE*, vol. 3, no. 1, pp. 40–44, 2005.

[16] D. Chrun, M. Cukier, A. Mosleh, and G. Sneeringer, “Investigating the impact of humans in information technology security: A case study at the university of maryland,” in *10th International Probabilistic Safety Assessment and Management Conference (PSAM)*, 2010.

[17] E. Schultz, “A framework for understanding and predicting insider attacks,” *Computers & Security*, vol. 21, no. 6, pp. 526–531, 2002.

[18] Verizon, “2012 data breach investigations report,” http://www.verizonbusiness.com/us/about/events/2012dbir/, 2012, retrieved: 07, 2012.

[19] PriceWaterHouseCoopers, “Global state of information security survey 2012,” http://www.pwc.co.uk/audit-assurance/publications/uk-information-security-breaches-survey-results-2012.jhtml, 2012, retrieved: 07, 2012.

[20] R. Richardson, “Csi computer crime and security survey,” *Computer Security Institute*, pp. 1–44, 2010.

[21] R. Walpole, R. Myers, S. Myers, and K. Ye, *Probability and statistics for engineers and scientists*. Macmillan New York, 1989.

[22] A. Arora, R. Krishnan, A. Nandkumar, R. Telang, and Y. Yang, “Impact of vulnerability disclosure and patch availability-an empirical analysis,” in *Third Workshop on the Economics of Information Security*. Citeseer, 2004.

[23] J. Wang, N. Xiao, and H. Rao, “Drivers of information security search behavior: An investigation of network attacks and vulnerability disclosures,” *ACM Transactions on Management Information Systems (TMIS)*, vol. 1, no. 1, p. 3, 2010.

[24] S. Pfleeger, “Useful cybersecurity metrics,” *IT Professional*, pp. 38–45, 2009.

Semantic Matching of Security Policies to Support Security Experts

Othman Benammar, Hicham Elasri, Abderrahim Sekkaki
University Hassan II, Faculty of Sciences Ain Chock
Casablanca, Morocco
emails{othmanbenammar@gmail.com, h.elasri@ancfcc.gov.ma, a.sekkaki@fsac.ac.ma}

***Abstract*— Management of security policies has become increasingly difficult given the number of domains to manage, taken into consideration their extent and their complexity. Security experts has to deal with a variety of frameworks and specification languages used in different domains that may belong to any Cloud Computing or Distributed Systems. This wealth of frameworks and languages make the management task and the interpretation of the security policies so difficult. Each approach provides its own conflict management method or tool, the security expert will be forced to manage all these tools, which makes the field maintenance and time consuming expensive. In order to hide this complexity and to facilitate some security expert's tasks and automate the others, we propose a security policies aligning based on ontologies process; this process enables to detect and resolve security policies conflicts and to support security experts in managing tasks.**

Keywords—Security-based policy; security management; ontology alignment; ontology enrichment.

I. INTRODUCTION

The security management in Cloud Computing (CC), Grid Computing (GC) or Distributed Systems (DS) requires the use of a dynamic environment-based policy to overcome a set of problems like the number of domains to manage and the permanent modifications that could occur inside managed environments themselves. However, the main aims for the management-based policy are to optimize, minimize the human interventions and automate some management tasks during security lifecycle.

Several frameworks of security-based policy and models have been proposed in the literature. The present paper will focus on the languages associated to KAOS [10], REI [11] and Ponder [5].

The main issue to manage many heterogeneous security domains is to assure interoperability between the enforced Security Policies (SP), and also detect and resolve conflicts.

Conflicts can be categorized in two forms, vertical and horizontal conflicts. The vertical ones occur when the SP are expressed in a common language while sharing heterogeneous knowledge on shared resources, this type of conflict is typically semantic. The horizontal conflicts occur when security policies are expressed with different languages; this type of conflict is due to different syntax for used languages. However, to detect and resolve horizontal conflicts, a third-party canonical language can be used to uniformize and hide syntactic and lexical heterogeneity; as a result, we find the vertical form.

Ontologies are the most used approach to represent semantic knowledge, and thereby provide robust mechanisms to resolve semantic conflict. In this sense, we use the ontologies as a pivot between languages towards to facilitate interpretation and understanding for the security experts.

Our goal is to provide a helpful method and tool to hide the complexity to deal with multiple specification languages that security expert can find to manage several security domains like those present in DS, GC or CC. So, we define an ontology used as a common knowledge, called in the literature support or background ontology [3]. This ontology is a semantic consensus on the concepts of different security policy specification languages. It is used in a distributed environment to supply security domains with a mutual agreement on the SP which implies a secured semantic interoperability. This paper presents an aligning-based ontologies process that supports security experts to manage SP , assures the SP conformity and consistency, detects and resolves conflicts that may be occur when adding, changing or merging SP.

This work will be presented as follows. The second section presents some related researches. The third section explains briefly the management-based policy principle. The fourth section will display the most used security-based policy frameworks such as KAOS, REI and PONDER. The fifth section presents our SP alignment process approach by using examples for illustration and a prototype is given in the sixth section. Finally, the conclusion will be featured in the last section.

II. RELATED WORK

In this section, we briefly present some of the research literature related to some security-based policy frameworks.

On one hand, there are some works in detecting conflicts of security policy related to syntax conflicts. Ponder and XACML [27] are typically non-semantic policy framework. On the other hand, there are some representative research works on semantic conflict detection. KAOS and REI are approaches that enriched with semantics using RDF [28] and OWL [25] as standards for policy specification. KAOS adopt policy priority for the conflict resolution of semantic policy [29]. REI combined the method of meta-policy and priority for conflict detection and resolution of semantic policy [18]. A comparative analysis between semantic and non-semantic language is made by Nejdl et al. [30] to show the advantages of semantic policy approach.

Since each semantic approach provides a conflict management method, the security expert will be forced to manage all these tools. Some works was interested to detecting conflict between SP related to a specified framework [18][23]. Our goal is to interface with different frameworks in order to centralize the conflicts management and offer the possibility to compare SP from different frameworks in order to detect and resolve conflicts. This work cover the first step that present an ontology-based method in order to remove all semantic SP ambiguity, assist security experts to resolve semantic conflicts and enable an automatic resolution for semantic conflicts by adopting a conflict resolution based on rules strategy i.e., by combining a resolution rules set for each conflict type.

III. MANAGEMENT-BASED POLICY PRINCIPLE

The objective of the management-based policy is the optimization as well as possible of security experts efforts. Thus, it first consists in determining the strategies and the tactics reflecting the security expert's objective and also representing them in policies form. Then, these policies must be presented as a set of rules to be understood by the management entities and stored in a Policy Repository (PR). The distribution and the application of these policies require to communicate these rules to the PDP (Policy Decision Point) and the PEPs (Policy Enforcement Point) managed by this latter. The high-level policy defined by security expert, cannot be directly understood by equipments or applications. It is therefore necessary to translate it into the configuration rules. At each stage, several policies may conflict. There is no simple recipe for solving all conflicts encountered. Ultimately, the security expert must perform this task.

The term subject refers to users, principals, or automated manager components, which have management responsibility (i.e., have the authority to initiate a management decision). The term target refers to objects on which the action can be performed. The relations between subject and target [12] are well defined by management policies and depend also on the nature of these latter. Thus, obligation policies define what a Subject must perform or not on the level of a Target, whereas authorization policies specify the access rights that could have a Subject on the level of a Target.

The policies rules database is replaced by both a domain service and a policy service. In our approach, the policies specification will be based on the ontologies, to ensure a horizontal and vertical semantics between SP specification languages.

In the next section, we present the security policies semantic specification languages chosen through literature.

IV. POLICY SPECIFICATION LANGUAGES AND FRAMEWORKS

SP can be specified in several ways and different approaches have been proposed in various application fields [16]. However, there are some general requirements that any policy representation must satisfy independently of its application scope:

– A great expressiveness that can meets the managed system security requirements
– A simple and effective vocabulary to facilitate to the security experts policy development tasks with various expertise degrees.
– An execution space that provide a political specifications mapping in different platforms.
– Scalability to ensure appropriate performance, enable the defined policies analysis and achieve an appropriate balance between the expression objectives, its calculating docility and ease of use.

This section does not provide security policies specification overview, but to describe the semantic aspects of some approaches that have been specifically designed and tested for the distributed systems management. We first present some semantic security approaches KAOS, REI and Ponder.

A. KAOS (Knowledge-able Agent-oriented System)

KAOS is a framework that provides the policy services allowing the specification, management, conflicts resolution, and execution of policies within fields. Policies are represented in DAML + OIL [2] as ontology. Policy ontologies KAOS distinguish between authorizations (i.e., constraints that permit or prohibit certain actions) and obligations (i.e., constraints that require an action to perform or to waive such requirement).

KAOS detects potential conflicts between the policies at the specification moment, every time a user tries to add a new policy to the directory service [15]. The engine identifies conflicts between policies using the subsumption mechanisms between classes and tries to resolve these conflicts through political order based on their priority and, if necessary, create new harmonized policies [17][18].

B. REI

REI is a framework that includes specifying policies support, analysis and reasoning in computer applications [11][12].

His language policy based on ontological logic allows users to express and represent the rights concepts, prohibitions, obligations and exemptions. These concepts correspond, respectively, to the positive and negative authorization conditions, and to positive and negative obligations in Ponder and KAOS.
REI is based on an application-independent ontology to represent the rights concepts, prohibitions, requirements, exemptions and the politics rules. This allows different components from the same domain to understand and interpret REI policies in the right way. REI's policy specification can be in forms of Prolog predicates or RDF-S [31] statements. REI extends OWL with the expression of relations like role-value maps, which makes the language itself more expressive than original OWL.

C. Ponder

Ponder is a declarative object-oriented language that supports several types of distributed systems management policies specification. It's providing technical structuring policies to meet the policy administration complexity in large company information systems [5][8][9]. The basic types of political rights in Ponder are obligations and permissions.

Ponder has a conflict detection tool to detect overlaps and conflicts between policies. Similar to KAOS and REI, the tool can detect inconsistencies in the policy specification that may arise between policies with opposite signs modalities which refer to the same subjects, targets or actions (e.g., conflict between permissions [4]).

Following is an SP comparative table for semantic specification languages based on several criteria.

TABLE I. SPECIFICATION LANGUAGES COMPARISON OF SEMANTIC SECURITY POLICY

	KAOS	**REI**	**Ponder**	**Our approach**
ontology-based	yes	yes	no	yes
policy-type	Access control based	Access control based	Access control based	Access control based and management
Policy representation	Owl	Rei (Prolog like syntax+ RDF-S	Ponder language specification	OWL
Open security interoperability support	Applying mediating and proxy agents on to specific domains	Applicable in specific domains	Applicable in specific domains	Applicable in specific domains
Interoperability representation	Explicit	Explicit	Explicit	Explicit
Reasoning support	Java theorem	Prolog engine	Event calculus representation	Java theorem
Security ontology enrichment	no	no	no	yes

Our goal is to ensure unified presentation of different semantics languages and other SP specifications through a specification languages transformation set to a single ontological representation. This transformation can resolve vertical level conflicts by hiding the heterogeneous knowledge semantic heterogeneity and horizontal level conflicts by hiding the languages heterogeneity. Since each approach KAOS, REI, or Ponder provides a conflict management method, the security expert will be forced to manage all these tools from different approaches. In order to hide this complexity, we also manage the horizontal level conflicts between different languages to facilitate some security expert tasks and automate the others. In the present work we focus on the semantic conflicts resolution tasks between SP. We will present in this work an ontology-based method in order to remove all semantic SP ambiguity, assist security experts to resolve semantic conflicts and enable an automatic resolution for semantic conflicts by adopting a conflict resolution based on rules strategy: by combining a resolution rules set for each conflict type.

D. Types of conflicts

To harmonize the SP, we must lift any type of conflict that we enumerate into three categories: syntactic, structural and semantic conflicts. The first conflict type concerns identity or abbreviation conflicts introduced during SP designing or primitives like operations and attributes. The second one involves structural conflicts between classes. Here we distinguish between those related to association type between two elements and those related to the hierarchy relative to an abstraction level. The third conflict type concerns the conflicts that involve semantic elements like naming conflicts, measurement and confusion. These conflict types can occur in two forms vertical and horizontal form.

– Vertical conflicts occur when we have SP expressed in a common language while sharing heterogeneous knowledge on shared resources, this type of conflict is typically semantic.
– Horizontal conflicts, they occur when security policies are expressed with different languages, this syntactic type of conflict is due to different syntax languages.

V. Semantic Alignment Process Of Security Policies

In our study, we could see that the ontology languages facilitate understanding between interoperate entities. However, interoperability access controls cannot be implemented without the semantic compatibility establishment between the SP agents participating in the exchange. One of the problems is heterogeneity between different organizations entities, this heterogeneity makes the semantic compatibility establishing so difficult. The scientific community became interested in ontological models for the access control purposes. The most significant models in our view are KAOS [10] and REI [13][20].

In this section, we present our SP alignment process that can possibly be described with semantic specification languages like KAOS, REI or others.

A. Security policy semantic alignment process

The semantic alignment process allows detecting and resolving naming semantic conflicts type between candidate SP for cooperation in distributed systems, and it enable to enhance the security ontology used as support ontology during the alignment process.

The alignment process inputs are:

– A SP set, denoted SP_1, SP_2 ... SP_n, selected by the security expert for their alignment.
– A security ontology chosen by the security expert according to SP type used in the distributed system. It will serve as support ontology during the alignment process.

The alignment process provides as outputs:

– A new enriched security ontology by new semantic relations added due to two treatments that we apply in the process: ontologies alignment and enrichment, which can be used to update the initial security ontology and serve as new support ontology in future integration iterations thereby increasing the process efficiency.
– Correspondence ontology, security experts can use this ontology to detect and resolve semantic conflicts in a semi-automatic process.

The semantic alignment process is highlighted in Figure 1.

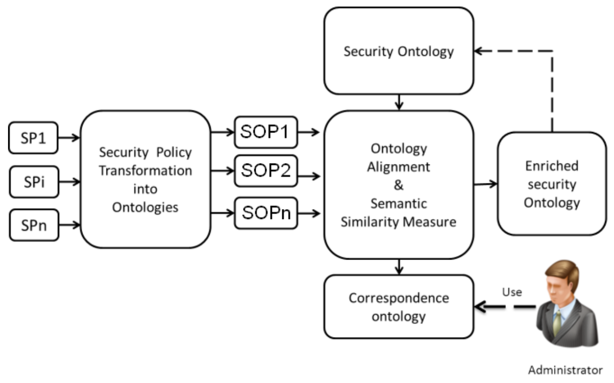

Security Ontology
SP1
SPi
SPn
Security Policy Transformation into Ontologies
SOP1
SOP2
SOPn
Ontology Alignment & Semantic Similarity Measure
Enriched security Ontology
Correspondence ontology
Use
Administrator

Figure 1. Matching security policy process.

Our proposal is based on several research projects results, especially those concerning the SP transformation described in the SP specification languages in ontologies description languages [7], and those related to the ontologies security alignment and enhancement [3][7][19][20]. The security policy transformation into ontologies relies on the work cited above to transform SP to Security Ontology Policies (SOP).

The integration process involves two steps:

1. The candidate security policies for alignment-based ontology

The candidates SP for alignment are conceptual safety rules set representations. They are described in a given modelling language (formal, semi-formal or natural).

Several studies have focused on the conceptual models transformation [7][21][22][24]. We rely on this work for the SP transformation into ontologies.

2. Aligning obtained ontologies based on security ontology

The ontologies alignment consists on establishing correspondences between two ontologies, with a priori on the same knowledge domain. It helps to find semantic relations between concepts defined in ontologies to align.

In order to achieve semantic SP alignment, we rely on an ontology aligning method [1] based on support ontology, also called in the literature “background ontology” [3].

This step takes as input:

– Ontologies obtained in the previous step.
– The security ontology.

Results:

– An ontology alignment results of all available ontologies as input.
– The ontology of security enhanced by new relationships.

Support Ontology is in our case, the SP ontology used by the distributed system.

B. Security ontology enrichment

The enrichment allows identifying new elements: concepts, terms and relations, then place them in an existing ontology. Enrichment and manual ontology construction prove to be tedious and expensive tasks [3]. Several studies have proposed automated methods or semi-automated enrichment and ontologies construction. Most of these methods rely on external sources from which new semantic knowledge are identified, evaluated and placed in the ontology to enrich it.

The method for measuring semantic similarity [1] uses security ontology enrichment process, when it has no semantic relation between concepts to align. To implement this process and demonstrate its feasibility, we chose, and that as an example, two rules among the various rules on semantic relationships:

– R1: Two concepts are similar if their neighbours are similar equivalents. According to Hassan et al. [24], two concepts are similar if their sub-concepts "son" are similar. This was confirmed in [6].
– R2: Two concepts are similar if their sub-concepts "son" are similar.

Rule R2 focuses on the composite concepts. The composite concepts represent concepts Father and sub-components concepts linked by a semantic relationship like “part-of”, are the concepts son.

Let C1 and C2 two concepts to align belonging to SOP_i; we distinguish three cases:

Case 1: C1 and C2 recognize a semantic relationship within SOP_i. This relationship is then injected into the Security Ontology (SO).

Case 2: C1 and C2 do not allow SOP_i in semantic relation while there are two concepts in SOP_i, C1’ and C2’ and two semantic relations of equivalence, the first between C1 and C1’ and the second between C2 and C2’.

From R1 we can deduce a new semantic relationship between C1 and C2 that is injected into the security ontology SO.

Case 3: C1 and C2 are concepts that do not allow composite semantic relation in SOP_i, while there are semantic relations between concepts in their respective son.

Let {C11, C12 ...C1n} the C1 son concepts set and {C21, C22 ... C2n} the C2 son concepts set, as is C1i and C2i admit a semantic relationship within SOP_i. From R2, we can deduce a new semantic relationship between C1 and C2 that is injected into the security ontology SO.

C. Semantic and syntactic similarity measuring

To achieve the ontologies alignment for the various SP, we reuse methods for measuring semantic and syntactic similarity proposed in [1], which proposes several methods for measuring semantic and syntactic similarity.

Ontologies alignment process product aligned ontology, which defines the semantic relationships between several sources ontologies concepts. Obviously, we need a model that combines semantic relationship to each one or more rules, they can be different types: transformation rules, merging, mapping, renaming or deleting...

D. Conflict resolution rules

To demonstrate how to use correspondence ontology, we present resolution rules for naming conflicts derived from semantic relation: homonym and synonym existing in correspondence ontology result of our process.

Conflict Resolution Rule 1: if we have a semantic relation type synonym in the correspondence ontology between concepts of sources ontologies, we offer security experts to rename the concepts with same name.

Conflict Resolution Rule 2: if we have a semantic relationship type homonym in the correspondence ontology between concepts of sources ontologies, we offer security experts to rename the concepts with different names.

Figure 2 shows a namely conflict resolution assisted by security experts based on a set of conflict resolution rules stored in a catalogue.

Based on correspondence ontology and the conflict resolution rules, we offer security expert decisions set represented by derived operations set. For example, in case of type synonym relationship in correspondence ontology then find in the catalogue the resolving conflicts (conflict resolution rule 1), then propose to security expert an operation "rename" one of concepts in conflicts.

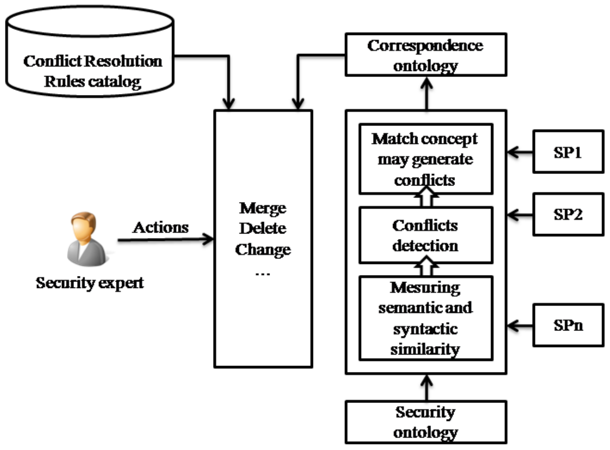

Conflict Resolution Rules catalog
Correspondence ontology
Match concept may generate conflicts
SP1
Conflicts detection
SP2
Actions
Merge
Delete
Change
...
Security expert
Mesuring semantic and syntactic similarity
SPn
Security ontology

Figure 2. Matching security policy process with actions.

E. Matching security policies in Cloud Computing environments

Cloud computing providers can build large datacenters at low cost due to their expertise in organizing and provisioning computational resources. The economies of scale increase revenue for cloud providers and lower costs for cloud users.

Several entities intend to take part in the benefits of Cloud Computing by outsourcing their infrastructure while keeping their policy and operational security system. To illustrate our process of matching the SP, we present the case of several entities wishing to outsource their infrastructure to a Cloud Computing and using different security policies expressed in different SP specification languages. Our approach offers to the Cloud Computing security experts a tool to manage these different SP while ensuring consistency with the Cloud Computing security policy if necessary. For this, we propose to align the different SP to provide a unified global vision and thus detect potential conflicts between these SP. The security expert can rely on an ontology correspondence as a decision support to detect synonyms and homonyms concepts; this example is highlighted in Figure 3. This process is essential toward a centralized management tool in CC environments.

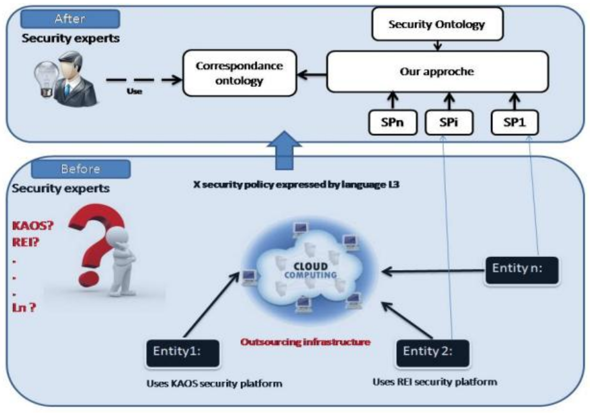


Figure 3. Matching security policy example

In the follow section, we show an example to illustrate the ontologies alignment process presented by Figure 2, and validate the function of this process. For example, as shown in Figure 4, we suppose that the collaborative environment is composed by two virtual domains. So, the input contains two cases which are obtained from domain A and domain B, each case describes the positive authorization policy in REI Prolog format, which specifies that an entity X (P in case 1 and Q in case 2) can use the printing service when it is an IT department member. The part of the instances describing two cases is as follow:

Case 1 from domain A

The following is a positive example for authorization policy in REI Prolog format [32].
has(P,permit(usePrintingService,[member(P, ITDepartment)]).

The example above specifies that an entity P can use the printing service when it is an IT department member in domain A.

Case 2 from domain B

We consider the same proposals described in case 1 by replacing the entity P by Q and domain A by domain B.

has(Q, allow(usePrintingService,[member (Q, ITDepartment)]).

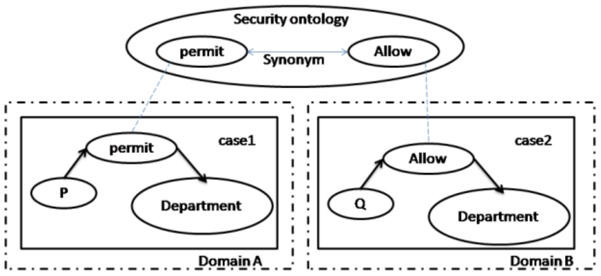


Figure 4. The two cases represented fragments ontologies

We apply the matching process of aligning two cases 1 and 2 from domain A and B, we obtain a correspondence ontology that represents the tow concepts in conflict(synonym(permit, allow)) and based on correspondence ontology to derive the rule of conflict resolution to use. In this example it is the conflict resolution rule 1 then proposes to security experts an operation “rename” one of the component concepts by the same concepts.

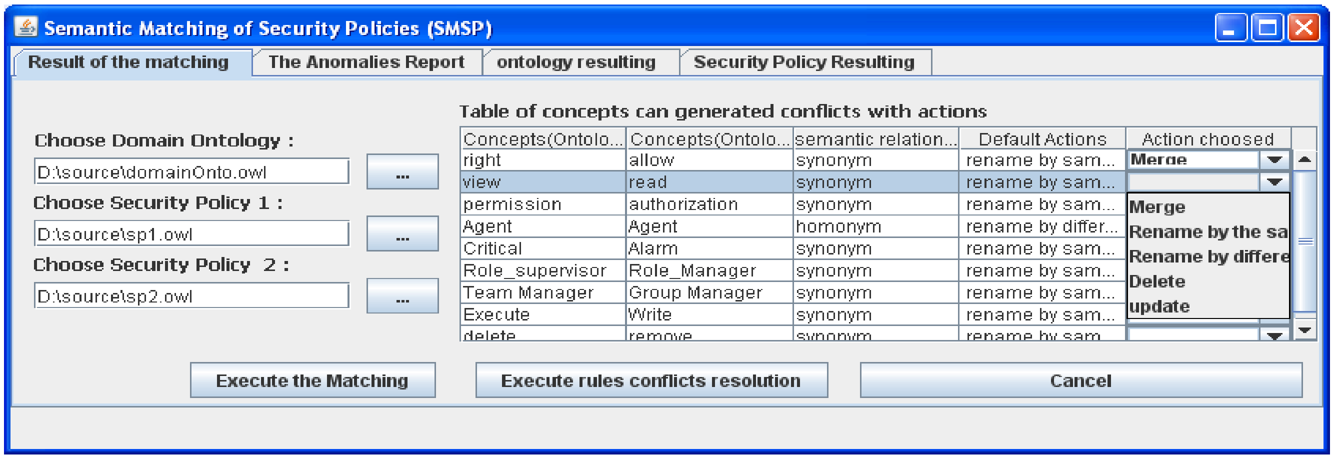
The example, shown in Figure 4, allows presenting how to solve the naming conflict between the concepts from the two cases specified by the security policy: REI. Our approach can solve all conflicts types: horizontal and vertical knowing that we define the resolution rules for each relationship between concepts in the correspondence ontology.

VI. Prototype

In order to validate and to evaluate our process of semantic matching of security policies, we have developed a prototype baptized Semantic Matching of Security Policies (SMSP). The prototype SMSP takes first in entry the domain security ontology and the security policies to match; and transforms them into ontologies described in OWL [26].

Security expert chooses the appropriate actions to perform through a list of rules already established in order to resolve the semantic conflicts. Then, SMSP applies various treatments associated with each step of our proposed method, in particular similarity measurement and security ontology enrichment. After those treatments, domain security ontology were be enriched with new semantic relationships toward to be used in similar situations.

Figure 5 presents a screen capture of SMSP that shows a use case for matching security policies; user can load domain ontology and the two security policies to match, and perform an action by merging, renaming or deleting conflicting concepts.



Choose Domain Ontology : D:\source\domainOnto.owl

Choose Security Policy 1 : D:\source\sp1.owl

Choose Security Policy 2 : D:\source\sp2.owl

Table of concepts can generated conflicts with actions

Concepts(Ontolo...	Concepts(Ontolo...	semantic relation...	Default Actions	Action choosed
right	allow	synonym	rename by sam...	Merge
view	read	synonym	rename by sam...	
permission	authorization	synonym	rename by sam...	
Agent	Agent	homonym	rename by differ...	
Critical	Alarm	synonym	rename by sam...	
Role_supervisor	Role_Manager	synonym	rename by sam...	
Team Manager	Group Manager	synonym	rename by sam...	
Execute	Write	synonym	rename by sam...	
delete	remove	synonym	rename by sam	

Figure 5. A prototype of Semantic Matching of Security Policies

VII. Conclusion And Future Work

We were interested in this work to conflict resolution semantic naming type when aligning SP in distributed multi-systems security policies. Our solution is based on ontologies mediation for SP cooperation and understanding. We propose a helpful process for security experts, this process permit to mask heterogeneity and resolve conflicts between SP, using different steps. The first and last steps concern the processing of conceptual ontological policies representations. The second step, which constitutes the fundamental part of this work, is a detecting method for the semantic conflicts between SP. We expect to continue this work first by a formal validation of the solution, and then by looking for opportunities to extend it to solve other types of semantic conflicts, including conflicts of measurement and confusion. We plan to further our research on the use of this approach in CC environments toward a centralised management of security policies, we will propose a framework that can interface with other security-based frameworks to ensure the interoperability and also enrich them with semantic relationships.

REFERENCES

[1] H. Elasri, A. Sekkaki, and L. Kzaz, “An ontology-based method for semantic integration of business components”, IEEE NOTERE, 11th annual International Conference on New Technologies of Distributed Systems, Paris, France, 2011, pp. 01-07.

[2] F. van Harmelen, P.F. Patel-Schneider, and I. Horrocks. “Reference description of the DAML+OIL”, Ontology Markup Language, 2001.

[3] B. Safar and R. Chantal, “Alignement d’ontologies basé sur des ressources complémentaires: illustration sur le système TaxoMap”, In Revue TSI, 2009, pp. 1211-1232.

[4] A. K. Bandara, E. Lupu, and A. Russo, “Using event Calculus to Formalise Policy Specification and Analysis”, Proceedings of 4th IEEE workshop on Policies for Distributed Systems and Networks, Lake Como, Italy, 2003, pp. 26-39.

[5] N. Damianou, N. Dulay, E. C. Lupu, and M. Sloman, “Ponder: A Language for Specifying Security and Management Policies for Distributed Systems”, Imperial College, Research Report Department of Computing, UK, 2001.

[6] L. Ben Ghezaiel, C. Latiri, M. Ben Ahmed, and N. Gouider-Khouja, “Enrichissement d'ontologie par une base générique minimale de règles associatives”, Conférence en Recherche d'Information et Applications (CORIA), Sousse, Tunisia, 2010, pp. 289-300.

[7] K. Barrett, S. Davy, J. Strassner, B. Jennings, S. Van Der Meer, and W. Donnelly, “A Model Based Approach for Policy Tool Generation and Policy Analysis Global Information Infrastructure Symposium”, GIIS, Marrakech, Morocco, 2007, pp. 99-105.

[8] M. Sloman, “Policy Driven Management for distributed Systems”, Plenum Press Journal of Network and Systems Management, Vol. 2, No. 4, 1994, pp. 333-360.

[9] N. Damianou, N. Dulay, E. Lupu, and M. Sloman, “The Ponder Policy Specification Language”, In proceeding of Workshop on Policies for Distributed Systems and Networks, Springer-Verlag, LNCS, Bristol, UK, 2001, pp. 18-38.

[10] A. Uszok, M. J. Bradshaw, J. Matthew, J. Renia, A. Tate, J. Dalton, and S. Aitken, “KAOS policy management for semantic web services”, IEEE Intelligent Systems, 2nd International IEEE Conference, 2004, pp. 32-41.

[11] L. Kagal, “REI: A Policy Language for the Me-Centric Project”. HP Labs Technical Report, 2002.

[12] L. Kagal, T. Finin, and A. Johshi, “A Policy Language for Pervasive Computing Environment”, IEEE 4th International Workshop on Policies for Distributed Systems and Networks, Lake Como, Italy, 2003, pp. 63-74.

[13] R. Masuoka, M. Chopra, Z. Song, Y. K. Labrou, L. Kagal, and T. Finin, “Policy-based Access Control for Task Computing Using Rei”, In Policy Management for the Web Workshop, Chiba, Japan, 2005, pp. 37-43.

[14] P. Shvaiko and J. Euzenat, “A survey of schema-based matching approaches”, Journal on Data Semantics (JoDS), 2005, pp. 146-171.

[15] http://www.ksl.stanford.edu/software/JTP/ [retrieved: July, 2012]

[16] S. Wright, R. Chadha, and G. Lapiotis, “Special Issue on Policy Based Networking”, IEEE Network, Vol. 16, No. 2, 2002, pp. 8-56.

[17] J. M. Bradshaw, A. Uszok, R. Jeffers, and N. Suri, “ Representation and reasoning for DAML-based policy and domain services in KAoS and Nomads”, The Autonomous Agents and Multi-Agent Systems Conference, Melbourne, Australia, New York, NY: ACM Press, 2003, pp. 835-842.

[18] A. Uszok, J. Bradshaw, R. Jeffers, and N. Suri, “KAoS Policy and Domain Services: Toward a Description-Logic Approach to Policy Representation, Deconfliction, and Enforcement”, IEEE 4th International Workshop on Policies for Distributed Systems and Networks, Lake Como, Italy, 2003, pp. 93-96.

[19] U. Visser, H. Stuckenschmidt, Schlieder, H. Wache, and I. Timm, “Terminology Integration for the Management of distributed Information Resource”, Kunstliche Intelligenz (KI), 2001.

[20] F. Fürst, “Contribution à l'ingénierie des ontologies: une méthode et un outil d'opérationnalisation”, thèse de doctorat à l’université de Nantes: Nantes, France, 2004, pp. 31-34.

[21] B. Tsoumas and D. Gritzalis “Towards an Ontology-based Security Management”, Advanced Information Networking and Applications, AINA, Vienna, Austria 2006, pp. 985-992.

[22] K. Verma, R. Akkiraju, and R. Goodwin, “Semantic Matching of Web Service Policies”, Proceedings of the Second Workshop on SDWP, Orlando, Florida, USA, 2005, pp. 79-90.

[23] G. Tonti, J. M. Bradshaw, R. Jeffers, R. Montanari, N. Suri, and A. Uszok, “Semantic Web Languages for Policy Representation and Reasoning: A Comparison of KAoS, Rei, and Ponder”, Second International Semantic Web Conference, Sanibel Island, FL, USA 2003, pp. 419-437.

[24] R. Hassan, M. Eltoweissy, S. Bohner, and S. El-Kassas, “Goal-Oriented Software Security Engineering: The Electronic Smart Card Case Study, Computational Science and Engineering”, IEEE International Conference, August 2009, pp. 213-218.

[25] T. Phan, J. Han, J. G. Schneider, T. Ebringer, and T. Rogers, “A Survey of Policy-Based Management Approaches for Service Oriented Systems”, Software Engineering Conference, Australian, 2008.

[26] D. L. McGuinness and F. V. Harmelen, “Owl web ontology language overview”, W3C Recommendation, 2004.

[27] T. Moses, “eXtensible Access Control Markup Language TC v2.0 (XACML), Organization for the Advancement of Structured Information Standards (OASIS), 2005, pp. 392-401.

[28] F. Manola and E. Miller, RDF primer, W3C Recommendation, 2004.

[29] A. Bandara, S. Calo, J. Lobo, E. Lupu, and A. Russo, “Toward a Formal Characterization of Policy Specification & Analysis”, Proceedings of Annual Conference of ITA, Maryland, US, 2007, pp. 01-09.

[30] W. Nejdl, D. Olmedilla, M. Winslett, and C. Zhang, “Ontology-Based Policy Specification and Management”, Proceedings of Second European Semantic Web Conference, Crete, Greece, 2005, pp. 290-302.

[31] http://www.w3.org/TR/rdf-schema/ [retrieved: July, 2012].

[32] Swedish Institute of Computer Science. SICStus Prolog. http://www.sics.se/sicstus/ [retrieved: July, 2012].

Statistical Disclosure: Improved, Extended, and Resisted

Navid Emamdoost, Mohammad Sadeq Dousti, Rasool Jalili
Data and Network Security Lab
Department of Computer Engineering, Sharif University of Technology
Tehran, Iran
{emamdoost@ce., dousti@ce., jalili@}sharif.edu

***Abstract*—Traffic analysis is a type of attack on secure communications systems, in which the adversary extracts useful patterns and information from the observed traffic. This paper improves and extends an efficient traffic analysis attack, called "statistical disclosure attack." Moreover, we propose a solution to defend against the improved (and, *a fortiori*, the original) statistical disclosure attack. Our solution delays the attacker considerably, meaning that he should gather significantly more observations to be able to deduce meaningful information from the traffic.**

Keywords-Privacy; Anonymity; Mix-Net; SDA.

I. INTRODUCTION

Anonymity is a mechanism for providing privacy in communications systems. Research on anonymous communications systems began by the seminal work of Chaum in 1981. He introduced the concept of *mixes* as a way of providing *unlinkability*, i.e. removing any association between entities in the system [1]. Since then, many anonymity protocols were proposed, each of which has different characteristics for various requirements. Besides designing these protocols, research is conducted on vulnerability analysis of anonymous communications systems. As a result, numerous "privacy-violating" attacks were discovered. The ultimate goal of such attacks is to eliminate or reduce the amount of anonymity. While some attacks exploit the vulnerabilities in the protocol design or implementation—such as timing attacks, tagging attacks, blending attacks, etc. [2], there are other kinds of attacks which assume that the protocol design or implementation has no vulnerabilities. Here, the attacker merely observes protocol execution, while being oblivious to the internal mechanism of anonymity protocol. If the attack is successful, the attacker will be able to find associations between users; that is, he can tell which users where communicating during the attack. These types of attacks are named *Intersection Attacks*. Statistical Disclosure Attack (SDA) is one of the most *efficient* intersection attacks applicable on mix-nets [8].

In this paper, we improve SDA, so as it requires fewer observations to reduce anonymity. For any "target" user Alice, we identify users whose behavior affects Alice's anonymity, and name them *cloak users*. To uncover Alice partners (those associated with her), it is sufficient to merely estimate the behavior of cloak users, rather than all users.

In addition, we extend SDA to cover a non-threshold mix named "SG-Mix". To this end, we use the queueing theory to model message delays in SG-Mix.

We finally suggest a solution to counter both the improved and the original SDA. Our solution delays the adversary considerably, since he needs to gather a great deal of observations before he can effectively mount the attack.

The rest of this paper is organized as follows. In Section II, we present a brief description of mix-nets and traffic analysis attacks, such as disclosure and statistical disclosure attacks. In Section III, we present our contributions: the improved SDA is demonstrated in III.A; III.B includes our extension to the attack to cover a low-latency protocol III.B, and III.C describes our proposed method to resist intersection attacks. We show the effectiveness of the improvement and resistance methods by simulations in Section IV. Finally, Section V concludes the paper.

II. RELATED WORK

Chaum [3] introduced anonymous communications by defining the concept of "mix." A mix is a special router that provides anonymity by changing the bit pattern of messages, and by reordering them, so that no arriving message can be linked to a leaving one. In the Chaum's mix, the message bit pattern is changed by cryptographic functions, that is, every message is decrypted and will then be sent to its receiver in the alphabetic order. If the mix gets compromised, no anonymity is provided. Hence, instead of using just one mix, usually a chain of mixes is used. The anonymity of messages is guaranteed as long as at least one mix in the chain is honest.

The first model of the mix, introduced in [3], is called "Threshold Mix," as it waits until a constant number of messages arrives, and then decrypts and flushes them out. Other types of mixes, based on their flushing methods include timed mixes, pool mixes, and Stop-and-Go (SG) mixes. A good survey of these flushing algorithms is available in [2]. Here, we review SG-Mix in more detail, as we are going to extend an attack on this protocol.

Kesdogan *et al.*[4] proposed a mix design which does not use batch processing. In this design, the sender derives—from an exponential distribution with parameter μ—a t_i delay for the *i*th mix in the chain, puts this delay in the encrypted message, and sends the message to the SG-Mix. The *i*th SG-Mix decrypts the message and obtains t_i. The message will be flushed after a t_i delay. As there is no batch processing, SG-Mix is considered a low-latency mix, and can be used in applications with sensitive timing constraints like web surfing. In [5], Danezis showed that SG-Mix provides the optimal mixing strategy among low-latency anonymity protocols.

Thus far, we have considered mixes as a strategy to provide anonymity in communications networks. Next, we will examine attacks against such anonymous systems.

The ultimate goal of these attacks is to reduce the anonymity of users. Good surveys of such attacks are available in [2][6]. Some attacks exploit vulnerabilities in the protocol design or implementation. The attacker can compromise protocol nodes, trace a message, delay a message or manipulate it to distinguish its relevance to other messages. In addition, the attacker can flood protocol nodes to learn a specific message destination. Moreover, it can replay captured messages from previous executions of the protocol. On the other hand, some other attacks are oblivious to the internal operation of anonymity protocol, and assume there is no vulnerability in the protocol design and implementation. Here, the attacker just observes the protocol execution, and concludes links between users from these observations. The latter type of attack is termed "intersection attacks."

Agrawal and Kesdogan [7] presented an *intersection attack* named "Disclosure Attack." In this attack, the anonymity system is abstracted as follows: a subset of senders sends their messages to a subset of receivers in several rounds. The passive attacker is able to observe messages entering to and leaving from the anonymity system. In each round, two sets of users can be identified: the set of users who send messages (sender-set), and the set of users who receive messages (receiver-set). The attack's aim is to reveal partners of a "target" user, named Alice. It is assumed that Alice has *m* partners. Moreover, she sends at most one message in each round. If Alice is going to send in a round, she selects her receiver uniformly from her *m* partners. The batch size is *b*; that is, in each round, *b* users participate. There are a total of N recipients, and each sender (except Alice) selects its recipient uniformly from all the recipients.

Unfortunately, the attacker in the above model has to do an NP-complete computation to identify Alice partners. As a solution, Danezis [8] proposed a statistical version of the disclosure attack. *Statistical Disclosure Attack* (SDA) is more efficient than the naïve disclosure attack, as it does not have to solve an NP-complete problem. However, SDA cannot identify Alice's partners for certain. There is always some uncertainty in the solution obtained by SDA.

In SDA, users' behavior is modeled by vectors $\vec{u}$, $\vec{v}$, and $\vec{o}_i$ of N elements. In these vectors, the ith element corresponds to the probability of receiving some specified message by ith user. Vector $\vec{v}$ models Alice's behavior. The ultimate goal of the adversary is to find this vector. In $\vec{v}$, all m partners of Alice have probability $\frac{1}{m}$ and the remaining *N-m* elements are zero. Vector $\vec{u}$ is used to model the behavior of users other than Alice. Initially, all the N elements of this vector are $\frac{1}{N}$. The attacker observes protocol execution and derives observation vectors $\vec{o}_i$ from the receiver-set of round i. The jth element of $\vec{o}_i$ denotes the probability that the jth user be Alice partner in round i. So, in each round that Alice participates, every user in the receiver-set has probability $\frac{1}{b}$ of receiving the message from Alice. Based on the law of large numbers, the attacker can take arithmetic mean of a large number of these observation vectors:

$$\overline{O} = \frac{1}{t}\sum_{i=1}^{t} o_i \approx \frac{\vec{v} + (b-1)\vec{u}}{b}. \qquad (1)$$

Accordingly, it can derive $\vec{v}$:

$$\vec{v} \approx \frac{b}{t}\sum_{i=1}^{t} o_i - (b-1)\vec{u}. \qquad (2)$$

In [9], authors extended this attack to *Pool Mixes*, which do not flush all messages, and always retain a constant number of messages in the pool. In [10], authors tried to use all observations, and find a better estimation for other users' behavior. They used arithmetic mean of observations in which Alice did not participate, as an alternative to vector $\vec{u}$.

Several methods have also been proposed to counter (i.e. delay) intersection attacks, most of which consider sending dummy messages by users or mixes [10][11][12].

III. CONTRIBUTIONS

In this section, we express our improvement in SDA's effectiveness, the way to extend it to SG-mixes, and also our proposal to resist against.

The classical SDA has many limiting assumptions; it assumes users (except Alice) can select their partners from all recipients with equal probability. In addition, Alice selects the receiver uniformly from her *m* partners. The mixing process is assumed to be batch processing.

It is more realistic to assume that every user in the anonymity system is associated with a set of partners and the user does not necessarily select the receiver uniformly from this set. Also incorporation of batch processing is a significant merit.

The bottleneck in SDA is the number of observations required to identify Alice partners. So, improving the attack means decreasing the number of such required observations. To do so, we use an estimation to model other users' behavior. SDA was proposed to be applied on anonymity protocols such as mix-nets which use batch processing. The attack was extended [9] to a probabilistic variation of mixes named *Pool Mix.* We extend it to SG-Mix, which is a low-latency anonymity protocol, whilst it delivers the most degree of anonymity among low-latency protocols [5]. Our proposed new method of resisting SDA neither requires the mixing strategy to change, nor requires all users to send dummy messages.

A. *Improving SDA*

We propose a more efficient way to estimate the behavior of the other users. To this end, we identify effective users on Alice's anonymity, and then estimate their behavior. In the classical SDA, Alice selects her recipients from the set M, with equal probability, and other users select their recipients uniformly from all *N* recipients. The attacker uses the vector $\vec{u}$ to model the behavior of other users. Therefore, the vector $\vec{v}$, which is the goal of the attack, is an N-ary vector having *m* components of $\frac{1}{m}$ and *N-m* zero components. None of these assumptions are realistic; every user can have her own set of recipients, and is not compelled to select her recipient uniformly from the set.

If we consider a specific set of partners for every user, we cannot use the vector $\vec{u}$ anymore; using $\vec{u}$ implies that users select their recipients uniformly from all the other users. Therefore, estimating the behavior of the other users is necessary. In [10], a solution is proposed to achieve such estimation. They used the *median of observations* from rounds in which Alice has not participated, to model the behavior of other users. The estimation includes all users in the anonymity system; whilst for an effective attack, we should consider only users who are effective in hiding the Alice's communication. Therefore, the method in [10] suffers from hiding of the behavior of users who affect Alice's anonymity. On the other hand, when the number of observations increases, this estimation becomes very similar to $\vec{u}$, and so is not an appropriate model.

In the rest of this paper, we use the notations in Table 1 to illustrate our method.

TABLE I. SYMBOLS USED IN THIS PAPER

Symbol	Descriptions
A	Set of all senders
A'_i	Sender-set in round i
G_i	Set of rounds in which user i participated
B	Set of all receivers
B'_i	Receiver-set in round i
$\vec{v}$	Alice's send vector
$\vec{u}$	Uniform send vector
$\vec{o}_i$	Observations vector in round i
$o_{i,j}$	jth element of ith observation vector
a_r	Number of Alice's messages in round r

To be more precise, the following definitions are considered.

Definition 1: S is a Boolean function demonstrating if the user *e* has participated in round *i*:

$$S(e,i) = \begin{cases} 1 & \text{if user } e \text{ has participated in round } i \\ 0 & \text{otherwise.} \end{cases}$$

Using *S*, the sender-set of the *i*th round can be defined as follows.

Definition 2: A'_i is the set of users participating in round *i*:

$$A'_i = \{e \in A \mid S(e,i) = 1\}$$

Supposing the protocol to be executed for *t* rounds, we can define the set G_i as follows.

Definition 3: G_i is the set of all rounds in which user e_i has participated:

$$G_i = \{j \mid 1 \le j \le t \wedge e_i \in A'_j\}$$

From Definition 3, we can divide rounds into two categories: those including Alice (G_{Alice}) and those excluding Alice ($\bar{G}_{Alice}$). We need an estimate on the behavior of users who affect Alice's anonymity.

When Alice participates in a round, her messages become anonymous among the other messages sent in that round. So, all users participating in the round affect the anonymity of Alice's messages. From the attacker point of view, such effective users must be identified and be considered in disclosing Alice partners.

In each round, attacker notices a set of senders who send their messages to a set of receivers. So, the sender-set and the receiver-set can be constructed for the round. If Alice participates in the *i*th round, every user in the *i*th sender-set affects the Alice's anonymity. To identify such users, we define cloak users as follow:

Definition4: for the user e_i, the set of cloak users include users who have participated in at least one round with e_i.

$$Cloak_i = \{e_j \mid e_j \in \bigcup_{k \in G_i} A'_k - \{e_i\}\}$$

The set of Alice's cloak users is $Cloak_{Alice}$. We need an estimate on the behavior of users in $Cloak_{Alice}$. To this end, we extract such rounds from $\bar{G}_{Alice}$ in which at least one user from $Cloak_{Alice}$ has participated. Such rounds are collected in the set P_{Alice} .

Definition5: The set P_i includes those rounds of $\bar{G}_i$ in which the cloak user of e_i has participated:

$$P_i = \{j \in \bar{G}_i \mid Cloak_i \cap A'_j \ne \emptyset\}$$

The average of the observations from rounds in P_{Alice} is used as the estimation for the behavior of cloak users ($\overrightarrow{CloackUsr}$) calculated as:

$$\overrightarrow{CloakUsr} = \frac{1}{|P_{Alice}|} \sum_{i \in P_{Alice}} \vec{o}_i \,. \tag{3}$$

Using this estimation and the *law of large numbers*, the vector $\vec{v}$ can be calculated as:

$$\vec{v} = \frac{b}{\bar{a}t'} \sum_{i \in G_{Alice}} \vec{o}_i - \frac{(b-\bar{a})}{\bar{a}} \overrightarrow{CloakUsr}. \tag{4}$$

Where $t' = |G_{Alice}|$ and $\bar{a}$ is the average of Alice's share in the number of messages sent in each round:

$$\bar{a} = \frac{1}{t'} \sum_{i \in G_{Alice}} a_i \,. \tag{5}$$

The improved SDA can be summarized as the following steps:

1. Observe all rounds of the protocol execution
2. Calculate observation vectors for each round
3. Divide rounds into two sets: G_{Alice} and $\bar{G}_{Alice}$

4. Identify cloak users
5. Construct P_{Alice}
6. Calculate $\overrightarrow{CloackUsr}$
7. Calculate $\vec{v}$

To this end, the attacker uses all available observations to identify Alice partners. Using this improved attack, a better estimation of behavior of effective users on Alice's anonymity will be achieved; and no constraint will be imposed on any other users' behavior. In Section IV, we explain how this method reduces the number of required observations.

B. *Extending SDA*

SDA has been applied to anonymity systems with batch processing. In [9], authors extended SDA to cover Pool Mixes, which is a probabilistic mix. In this section, we extend the attack to cover a low-latency anonymity protocol named SG-Mix [4]. We chose SG-Mix due to its provision of the most optimal mixing strategies among low-latency protocols [5].

The model of anonymity system in SDA is a simplified form of mix-net, where execution of protocol is divided into rounds. In each round, a subset of users sends their messages to another subset of users. So, in protocols such as mix-nets, which collect input messages into batches and processes them after a threshold, each batch is equivalent to a round. However, batch processing imposes a great delay to the delivered messages. As most of online applications have timing constraints, low latency anonymity protocols would be of more interest.

For simplicity, we extend the attack to one SG-Mix, and assume whenever Alice sends a message to SG-Mix, she will not send another one until the first one has left the SG-Mix ($a_r=1$).

To apply SDA to any anonymity protocol, the protocol must be mapped onto the attack model. To extend the attack to SG-Mix, it is necessary to identify sender-sets and receiver-sets of each round. To do so, we model the delay of a SG-Mix.

1) *SG-Mix Delay Model*

As mentioned in Section II, the distribution of message delays is exponential with parameter μ. Assuming the message arrival distribution is Poisson with parameter λ, then the SG-Mix can be modeled as an $M/M/\infty$ queue. The maximum delay of a message (we refer to it as τ) can be found using such a modeling.

As distribution of delay in SG-Mix is exponential with parameter μ, the mean and standard deviation of the delay is $\frac{1}{\mu}$. Using one-sided Chebychev's inequality, the maximum delay of a message in SG-Mix can be estimated. The inequality says:

$$\Pr(X + \mathrm{E}(X) \geq k\sigma) \leq \frac{1}{1+k^2}. \tag{6}$$

where X is a random variable with mean $E(X)$ and standard deviation σ. The parameter k determines the amount of confidence. For example, for $k=3$, at least 90% values of X are less than or equal to $E(X)+k\sigma$. In such a case, with the confidence of $1-\frac{1}{1+k^2}$, it can be said that a message will leave SG-Mix at most $\frac{k+1}{\mu}$ after its entrance. In the other words:

$$\tau = \frac{k+1}{\mu}. \tag{7}$$

To find τ, μ is required; which is the parameter of delay distribution. In an M/M/∞ queue, the number of messages in the queue, upon entrance of a new message is $\frac{\lambda}{\mu}$. To provide anonymity for messages, it is necessary that at least one message be in the queue upon its entrance; i.e. $\frac{\lambda}{\mu} \geq 1$. Otherwise, SG-Mix behaves as a first-in first-out queue and no anonymity will be provided [5], i.e:

$$\mu \leq \lambda. \tag{8}$$

The parameter λ can be anticipated from the rate of incoming messages. So, λ is used as an upper bound for μ in our relations. To estimate λ, incoming messages to SG-Mix can be observed for a period of time *T*. Considering the number of entered messages in this period as x, $\lambda = \frac{x}{T}$; having λ as the message entrance rate in the unit of time.

2) *Construct Sets*

Using the maximum delay from (7), we can build sender-set and receiver-set. The set of all messages to which Alice's message blend forms receiver-set; and the set of all potential senders of messages in receiver-set, forms sender-set. The following theorems illustrate formation of these two sets.

Theorem 1: If Alice's message enters SG-Mix at time *t*, the receiver-set is comprised of all users receiving at least one message in the interval $W_1 = [\mathrm{t}, \mathrm{t}+\tau]$.

Proof: As the maximum delay of a message in SG-Mix is τ, Alice's message will leave the Mix at most at time$\mathrm{t}+\tau$. So, every user receiving at least one message from Mix in $[\mathrm{t}, \mathrm{t}+\tau]$ interval, can be Alice's partner, thus must be in the receiver-set. □

Theorem 2: If Alice's message enters SG-Mix at time *t*, all users who send a message to the Mix in the interval $W_2 = [\mathrm{t}-\tau, \mathrm{t}+\tau]$ will be in the sender-set.

Proof: From Theorem 1, the receiver-set is formed. So we try to relate messages in the receiver-set to senders. As the maximum delay is τ, so the message leaving the Mix at time *t*, could have been sent at worst at $\mathrm{t}-\tau$ (with maximum delay). The message which leaves Mix at time $\mathrm{t}+\tau$, could have been sent at $\mathrm{t}+\tau$ (with no delay). Thus, every user who sends a message to Mix in the interval $[\mathrm{t}-\tau, \mathrm{t}+\tau]$ interval, is a member of the sender-set. □

To perform the attack, observation vectors ($\vec{o}$) are necessary. As mentioned earlier in Section II, these vectors can be calculated from the receiver-sets. Cloak users are identified from sender-sets. The only remaining parameter prior to performing the attack is the batch size (b). In SDA, this parameter indicates the number of sent messages in each batch; but in SG-Mix there is no batch processing. The batch size determines number of messages which are blended together to provide anonymity. The size of the sender-sets can be used as parameter b. The number of users participating in each sender-set differs from round to round; and so we use the mean size of sender-sets as parameter b.

The extension of SDA to SG-Mixes can be summed up through the following steps:

1. Observe the protocol execution
2. Estimate the parameter λ
3. Calculate the intervals W_1 and W_2
4. Determine the sender-set and the receiver-set for Alice's messages
5. Obtain parameter b from the size of sender-sets
6. Calculate the vectors $\vec{o}$ from receiver-sets
7. Identify cloak users from sender-sets
8. Determine the sender and receiver-sets for cloak users
9. Calculate $\vec{o}$ vectors for cloak users
10. Calculate $\overrightarrow{CloakUsr}$
11. Calculate $\vec{v}$

C. *Resisting Against the Improved SDA*

There is no way to fully defend a protocol against intersection attacks such as SDA; whereas some ideas have been proposed to delay the attack; that is, increasing the number of required observations to succeed. Almost all of such methods include sending *dummy messages* by mix or by users in each round. However, the methods suffer from imposing of a great communication load due to sending of dummy messages. In some methods, users must participate in each round despite their willingness. The mix strategy must also be changed (to send/drop dummy messages) and thus the mixing process takes longer.

As mentioned in Section IV, increasing the value of two parameters b and m leads to an increase in the number of required observations. Sending dummy messages by users or mix tends to benefit from the role of parameter b. This parameter determines the number of messages blending together in each round. So, sending dummy messages is an attempt to hide Alice's message among more cloak users. Our Sybil defense benefits from the role of parameter m in the attack difficulty. The parameter determines the diversity of Alice's selection. More divert selections leaks less information about each recipient's identity.

In our method, Alice participates with n pseudonyms in the anonymity protocol. These pseudonyms are not linkable to Alice's identity in any way. Every pseudonym has its own set of recipients (different m). In each round where Alice intends to send a message, she sends a message by any of her pseudonym as well. For example, if Alice participates in the protocol as her own identity as well as another identity like A'; whenever Alice wants to send a message, she selects her recipient from a set of m members, and A' selects her recipient from a set of m' members. In each round where Alice is included in the sender-set, A' is also included; accordingly the attacker cannot distinguish partners of A' from partners of Alice. This can be considered as a scenario in which a user selects her partners from a set of size $m + m'$.

More pseudonymous identities lead to more complexity for the attacker. We refer to our method as Sybil defense, because of using multiple and unlinkable identities.

In Section IV, we will describe the implementation and evaluation of our defense method against the standard and improved SDA. As advantages of our defense method, there is no need to overwhelm the network by dummy messages and also, each user can use such protection by her willing and based on the amount of required anonymity.

IV. SIMULATION

Our simulator is inspired from [10]. Similar simulators was also used in [12] and [13].

A. *Simulation Plan*

For simulation, we generated observations and applied the standard and improved SDA to a unique system *configuration* under assumptions of original SDA. By configuration, we mean specific values of system parameters *N, b,* and *m*. To create observations for a configuration, we determined the set of partners for all users. To simulate each round, a sample of size b was obtained with replacement from the set of all users. This sample denotes the set of senders in that round. The sampling was performed *with replacement* to allow each user to send more than one message per round. For each sender, the corresponding receiver(s) was (were) determined based on the sender's partner set.

To show effectiveness of our improvement, we applied both attacks to several configurations and compared the number of required observation for each successful attack. We continued an attack until it revealed at least 80% of Alice's partners or passed over the limit of 5000 observations. We repeated the experiment 100 times per configuration before reporting the attack results. The pseudo code of the simulation is shown in Figure 1.

B. *Results*

We run the simulation with default values of $N=20000$, $b=50$, and $m=20$. In each comparison, we changed one parameter, whereas the other two were kept unchanged.

Figure 2 depicts the effect of changing N (number of system users) on the attacks. As N increases, the number of required observations decreases for both the attacks; with the need of fewer observations in the case of the improved SDA.

Figure 3 demonstrates the effect of batch size on the attacks. As parameter b increases, the number of required observations increases. As increasing the batch size results in increasing the number of messages which blend with Alice's ones, the problem become more difficult. As batch size increases, the improved attack needs less observation than the standard SDA.

Figure 4 shows the role of parameter m on the attacks. This parameter defines the number of Alice partners. By increasing m, Alice sends messages to more different users, so Alice's behavior becomes more diverse. For SDA, the number of observations increases extraordinarily when $m \geq 20$, but for the improved SDA, the number of observations increases gently. In the other words, when $m \geq 20$, SDA becomes inapplicable; while the improved SDA is still applicable. Accordingly, the main advantage of our improvement is where Alice increases her behavior diversity.

```
#Generate Observations
for$i$ = 1 .. $t$
    $senderSet$ ← a sample of size $b$ from $N$
    for each $k$ in $senderSet$ do
        Choose a receiver from $k$'s send vector
        and insert it into $receiverSet$
    end
end

# Statistical Disclosure Attack
$rounds$ ← rounds that $Alice$ participated in it
t ← |rounds|
for each $r$ in $rounds$
    for each k in receiverSet(r)
        O(k) += 1/b
    end
end
$\bar{O} \leftarrow \sum_k \frac{O(k)}{t}$
$\vec{v} \leftarrow b.\bar{O} - (b-1).\vec{u}$

# Improved SDA
Calculate $O$ vectors for all rounds, as before.
for each $r$ in $rounds$
    if $Alice$ is a sender in $r$
        then $G \leftarrow G \cup r$
        else $\bar{G} \leftarrow \bar{G} \cup r$
    end
for each $r$ in $\bar{G}$
    add all senders of $r$ except $Alice$ to $Cloak_{Alice}$
end
for each $r$ in $\bar{G}$
    if $Cloak_{Alice} \cap A'_r \neq \emptyset$
        then $P \leftarrow P \cup r$
end
$\overrightarrow{CloakUsr} \leftarrow \frac{1}{|P|} \sum_k (O(k) | k \in P)$
$\vec{v} \leftarrow \frac{b}{|G|} \sum_k (O(k) | k \in G) - (b-1)\overrightarrow{CloakUsr}$
```

Figure 1. Simulation pseudo code

To show the effect of the Sybil Defense introduced in Section III.C, we changed the way of generating of observations. In all the related simulations, we used just one pseudonym (A') for Alice. To determine the senders of a round, if Alice is included as a sender, we inserted the user A' to the set of senders and for every user in the sender-set, the corresponding recipient is selected from the partner set. The simulations confirmed that no attack could finish in none of the configurations before reaching the observation limit. This means that the attacker is unable to identify at least 80% of Alice's partners, whereas prior to employing the Sybil defense, both attacks were finished before reaching the 5000 observation limit. So the Sybil defense can effectively delay the attacker's success.

While we examined our resistance method on SDA, it is effective on all intersection attacks, such as the Hitting Set Attack [14], as it can unsettle the uniqueness of hitting set.

V. CONCLUSION AND FUTURE WORKS

In this paper, we improved the Statistical Disclosure Attack (SDA) in a way that it needs fewer observations to succeed, and extended it to cover a non-threshold mix protocol called SG-Mix. Finally, we proposed a resistance method to delay attackers' success.

We relaxed some limiting and unrealistic assumptions of SDA. In the relaxed assumptions, other users can have their specific sending vectors, and are not compelled to select their receivers uniformly from all users. Moreover, Alice can select her receivers non-uniformly from her m partners.

To model the message delay in SG-Mix we employed the queueing theory. An SG-Mix can be modeled as an $\mathrm{M/M/\infty}$ queue with Poisson arrivals and exponential delays. From there the maximum delay of a message was calculated and then observation vectors were constructed.

Our resistance method resulted in a great effect on the attack. In the normal scenario, the attacker was able to identify at least 80% of Alice's partners, while by employing just one pseudonymous identity by Alice, the attack became inapplicable. We believe that our method is effective on all intersection attacks such as the hitting set attack (by unsettling the uniqueness of hitting set), but this must be evaluated in more details as future work.

REFERENCES

[1] A. Pfitzmann, and M. Köhntopp, "Anonymity, unobservability, and pseudonymity—a proposal for terminology," *Lecture Notes in Computer Science, Volume 2009/2001,* vol. 2009/2001, pp. 1-9, 2001.

[2] M. Edman, and B. Yener, "On anonymity in an electronic society: A survey of anonymous communication systems," *ACM Computing Surveys (CSUR),* vol. 42, pp. 1-35, 2009.

[3] D. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM,* vol. 24, pp. 84-90, 1981.

[4] D. Kesdogan, J. Egner, and R. Büschkes, "Stop- and- Go-MIXes Providing Probabilistic Anonymity in an Open System," in *Information Hiding*. vol. 1525, ed: Springer Berlin / Heidelberg, 1998, pp. 83-98.

[5] G. Danezis, "The Traffic Analysis of Continuous-Time Mixes," in *Privacy Enhancing Technologies*. vol. 3424, D. Martin and A. Serjantov, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 35-50.

[6] A. Back, U. Möller, and A. Stiglic, "Traffic Analysis Attacks and Trade-Offs in Anonymity Providing Systems," *Information Hiding,* vol. 2137, pp. 245-257, 2001.

[7] D. Agrawal, and D. Kesdogan, "Measuring anonymity: The disclosure attack," *IEEE Security & Privacy,* vol. 1, pp. 27-34, 2003.

[8] G. Danezis, "Statistical disclosure attacks: Traffic confirmation in open environments," presented at the Security and privacy in the age of uncertainty: IFIP TC11 18th International Conference on Information Security (SEC2003), Athens, Greece, 2003.

[9] G. Danezis, and A. Serjantov, "Statistical Disclosure or Intersection Attacks on Anonymity Systems," in *Information Hiding*. vol. 3200, J. Fridrich, Ed., ed: Springer Berlin / Heidelberg, 2005, pp. 293-308.

[10] N. Mathewson, and R. Dingledine, "Practical Traffic Analysis: Extending and Resisting Statistical Disclosure," in *Privacy Enhancing Technologies*. vol. 3424, D. Martin and A. Serjantov, Eds., ed: Springer Berlin / Heidelberg, 2005, pp. 17-34.

[11] O. Berthold, and H. Langos, "Dummy Traffic against Long Term Intersection Attacks," in *Privacy Enhancing Technologies*. vol. 2482, R. Dingledine and P. Syverson, Eds., ed: Springer Berlin / Heidelberg, 2003, pp. 199-203.

[12] N. Mallesh, and M. Wright, "Countering Statistical Disclosure with Receiver-Bound Cover Traffic," in *Computer Security – ESORICS 2007*. vol. 4734, J. Biskup and J. López, Eds., ed: Springer Berlin / Heidelberg, 2008, pp. 547-562.

[13] D. Kedogan, D. Agrawal, and S. Penz, "Limits of anonymity in open environments," in *Proceedings of Information Hiding Workshop (IH 2002)*, 2002, pp. 53-69.

[14] D. Kesdogan, and L. Pimenidis, "The hitting set attack on anonymity protocols," in *Proceedings of 6th Information Hiding Workshop (IH 2004)*, 2004, pp. 326-339.

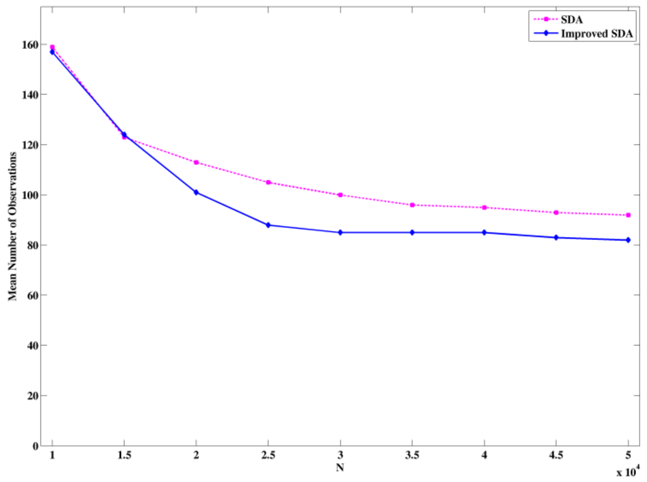


Figure 2. Effect of parameter N (number of users) on attacks

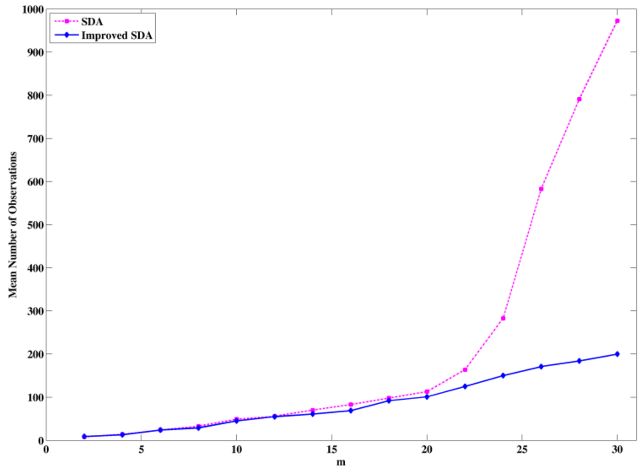


Figure 4. Effect of parameter m (number of partners) on attacks

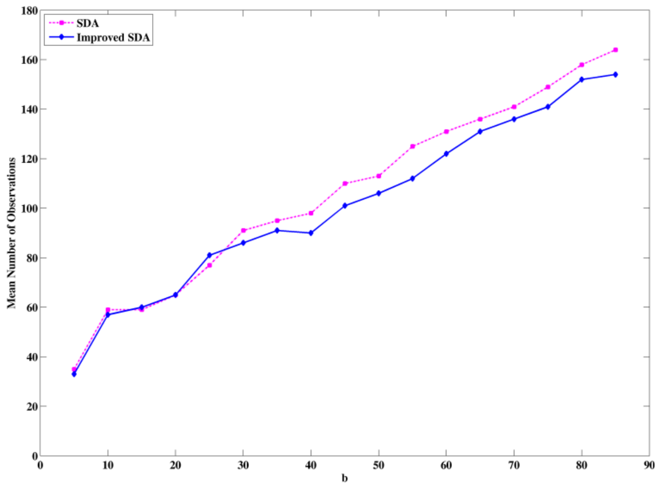


Figure 3. Effect of parameter b (batch size) on attacks

Compacting Security Signatures for PIGA IDS

Pascal Berthomé, Jérémy Briffaut, Pierre Clairet
ENSI de Bourges – Université Orléans – LIFO 18020 Bourges Cedex, France
Email: {pascal.berthome,jeremy.briffaut,pierre.clairet}@ensi-bourges.fr

***Abstract*—PIGA (Policy Interaction Graph Analysis) is a tool that detects malicious process behaviours by analysing the operating system activities. This tool uses signatures that represent illegal activities of some malicious user. These signatures are generated from a graph that models the performed operations at operating system (OS) level. For usual security properties, the number of signatures is large and they are stored in the memory during the detection process. In this paper, we present a way to reduce the memory required to store the signatures while preserving their quality. The methodology is derived from the modular decomposition of graphs. We investigate the impact of such an approach for the confidentiality property. The efficiency of the methodology is evaluated on interaction graphs of real operating systems. The number of signatures is divided by 20 for the tested confidentiality property.**

***Keywords*-security; operating system; modular decomposition.**

I. Introduction

Security in the Information Technology domain relies on global properties that need to be ensured. Integrity and confidentiality are two of the most important ones. Many others have been defined in the literature but they are mainly based on the noninterference principle [1][2]. PIGA [3] is a tool that can be used as an intrusion detection system (IDS) or an intrusion prevention system (IPS). An intrusion is an action that breaks a security property that is declared in the security policy.

In order to prevent illegal activities, PIGA [3] monitors the kernel activities (system calls) and examines the traces of the different processes. In [4], the authors present an overview of the different techniques of intrusion detection based on externally specified rules. For example, Remus [5] or BlueBoX [4] have a policy-based approach. The goals of these examples are different. The aim of Remus is to minimise the performance impact whereas BlueBoX tries to minimise the impact on the kernel. However, these systems detect some intrusion without taking into account the previous operations performed in the OS. PIGA defines on the contrary, signatures that describe the potential attacks on the operating system, by considering the temporal order of the operations.

One important feature of PIGA is to detect security property breaks by using a set of signatures. Many intrusion detection system are signature-based. For example, the Argos System also computes signatures; however, these signatures are generated from the examples of attacks that have already occurred [6]. The particularity of PIGA is that these signatures are generated off-line directly from the expression of the security properties. The drawback of PIGA is that the signature base is large for real systems and requires a significant memory space allocation for the detection process; furthermore, this base has to be restricted to small signatures because the computation requires a huge amount of memory, i.e., exponential with the length of the signatures [3]. For a complete system based on Fedora using graphical user interface the signatures base takes more than 500 MB.

This paper proposes an improvement of the PIGA system. In order to reduce the memory space needed, we propose a way to compact the information required for the intrusion detection. This paper focusses on the confidentiality property since it is the property that generates the largest number of signatures [3]. PIGA uses a graph to represent the system and its interactions, thus, this improvement is based on the reduction of this graph. For this, we exploit the modularity property of the input graph, i.e., the fact that some groups of nodes have the same behaviour considering the remaining part of the graph.

In order to present this contribution, the paper is organised as follows. In Section II, a brief presentation of PIGA is given and some security definitions are recalled. Section III presents the problem tackled in this paper. Section IV presents the major contribution of this paper, i.e., the use of the modular decomposition of graph to lower the size of the memory required to store signatures. It also provides a theoretical evaluation of the efficiency of our method. Section V shows that our strategy can be used to decrease the size of the information stored for detecting confidentiality threat on real systems. Section VI draws some conclusions and the direction of further works.

II. General Context

This section summarises the study provided for the PIGA system in [3]. The goal of PIGA is to monitor all the system activities and detect those that break rules defined by the administrator. This section proposes to recall the definitions of the main terms of security used in this paper through the prism of the PIGA system.

We present the process for generating a set of signatures from the security policy and the interaction graph. This process is explained throughout the following subsections. In Section II-A, the security properties are presented. In Section II-B, we describe the fundamental elements of a system, i.e., the security contexts. In Section II-C, we recall how these contexts are connected through the interactions of the system. In Section II-D, we glue these elements together in order to represent the system by a directed graph. Finally in Section II-E, we present how the signatures are computed for the confidentiality property.

A. System and security properties

A system can be seen as a state machine, where a state is characterised by the state of all the available resources [1]. The state of the system changes at each system call. In order to ensure the security of the system, the administrator defines the rules that represent the frontier between the *safe* and the *unsafe* state of the system. These rules are called security properties and the set of these security properties is called security policy. They are mainly based on the noninterference principle [7].

Integrity and *confidentiality* are the most common security properties [8]. The aim of the *confidentiality* property is to ensure there is no illegal information transfer. For example, a *confidentiality* property of the file `/etc/shadow` for the users prevents any information transfer from `/etc/shadow` to any user. The goal of the *integrity* property is to ensure there is no unwanted modification. For example, an *integrity* property of `/etc/shadow` for the users prevents any modification of `/etc/shadow` by a user.

B. Security contexts

PIGA is implemented upon SELinux, but it could be defined upon other mandatory access control (MAC) systems. Thus, it uses SELinux security contexts to label the resources of the system (file, process, resource, network port, ...). By extension, we say that two resources belong to the same security context if they have the same label.

In their simplest format, the security contexts in SELinux are defined by three elements: the user, the role and the type of the entity. For example, the security context of `/etc/shadow` is `system_u:object_r:shadow_t`. Several entities playing the same role can have the same security context. For example, `user_u:object_r:user_home_t` contains all the files in the home directory of the user.

C. Interactions and sequences

Any elementary system operation involves two resources, thus implying two security contexts [1]. This defines an *interaction* between two contexts. In SELinux, only allowed interactions are described explicitly. However, as shown in [3], using these allowed interactions cannot ensure that the system would remain in a safe state. Some security properties can be broken by performing successive operations. A *sequence* is then defined as a set of successive interactions, e.g., an information flow between two security contexts.

The aim of the PIGA system is to automate the generation of the sequences that break the security properties, defining the signatures as explained in the following sections.

D. Interaction and flow graphs

As shown before, PIGA models the system with a directed graph that represents the security contexts and the set of allowed interactions. More precisely, the system is represented by the interaction graph $G = (V, A)$, such that:

- V represents the security contexts, and
- A the set of interactions between those contexts.

Furthermore, any arc in this graph is labelled. The label of the arc from the security context sc_1 to the security context sc_2 represents the set of operations from sc_1 to sc_2 allowed by the MAC policy of the system. For example,

```
user_u:user_r:xserver_t
    system_u:object_r:mtrr_device_t
            file  { write read };
```

represents the possible interactions from the context `user_u:user_r:xserver_t` to `system_u:object_r:mtrr_device_t`. It means that any element having `user_u:user_r:xserver_t` security context can perform read and write operations on the files of the context `system_u:object_r:mtrr_device_t`. SELinux provides a wide range of permissions that can be allowed between different contexts. Figure 1 shows a simplified interaction graph.

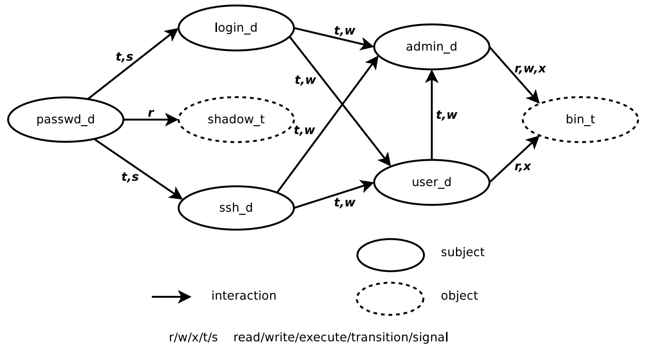


Fig. 1. A simplified interaction graph

From the interaction graph, we derive other graphs that are used in the generation of signatures. These new graphs are obtained by filtering the labelling function. These filtering operations may have different consequences on the arcs: an arc can be removed from the graph if the resulting label is empty; an arc also can change its direction by considering any property to be preserved with this operation. In the case of the confidentiality property, it uses the flow graph $FG = (V, A)$ where V is the set of the security contexts and A represent all the possible information transfers between the security contexts. The flow graph can be deducted from the interaction graph by removing the labels that do not imply flow transfer, maybe removing some arcs at this step. The remaining arcs deal with `read` and `write` operations. If a write operation occurs from context sc_1 to sc_2, there is a flow transfer from sc_1 to sc_2 and leads to an arc in the flow graph from sc_1 to sc_2. If a read operation occurs from context sc_1 to sc_2, then there is a flow transfer from sc_2 to sc_1 and leads to an arc in the flow graph from sc_2 to sc_1. Figure 2 presents the resulting flow graph derived from the interaction graph of Figure 1.

E. Signatures

A signature represents a set of actions, ordered or not, that breaks a security property. Note that a signature is derived

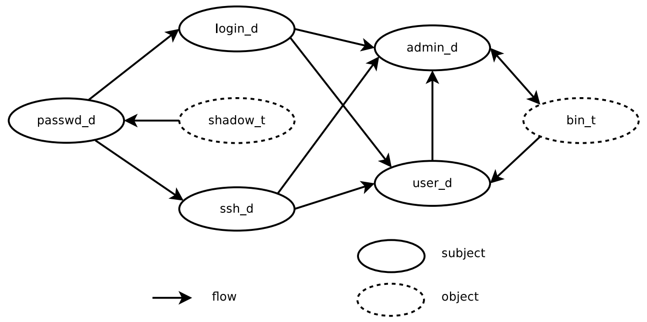


Fig. 2. The corresponding flow graph of the interaction graph of Figure 1

from a security property, but a security property generates several signatures. In PIGA, a signature represents an interaction, a sequence or a combination of sequences and/or interactions.

This paper focusses on confidentiality. The experiments of [3] show that this property generates the greatest number of signatures. In this case, a signature is a sequence that represents a succession of possible information transfers. For two contexts sc_1 and sc_2, $sign(sc_1, sc_2)$ represents the set of signatures between sc_1 and sc_2 and can be computed as the set of simple paths between sc_1 and sc_2 in the flow graph.

For example, from a system corresponding to Figure 1, the administrator would like to ensure that any file in the `shadow_t` security context cannot be accessible to any user (having security context `user_d`). Considering the flow graph in Figure 2, this would be possible that information from `shadow_t` arrives to `user_d` in only three steps without breaking the SELinux policy. The analysis of the flow graph results in 4 ways of breaking this security property :

```
shadow_t-> passwd_d-> ssh_d-> user_d
shadow_t-> passwd_d-> login_d-> user_d
shadow_t-> passwd_d-> ssh_d-> admin_d-> bin_t-> user_d
shadow_t-> passwd_d-> login_d-> admin_d-> bin_t-> user_d
```

When the PIGA system is active, all the interactions are audited and the system memorises the progression of all the signatures. If an interaction completes a signature, the system will forbid the last operation if it is in prevention mode and it will warn the user in permissive mode. Indeed, PIGA can be used either as an IDS or as an IPS.

III. Problem definition

As seen in the previous section, PIGA computes all the signatures off-line and stores them in the system. As shown in [3], the number of signatures may be large even for systems having a small number of services. These signatures are required for the detection process. However, they take a huge place (several hundreds of megabytes) in the memory. Furthermore, the interaction graph may be so large that it is unrealistic to compute all the signatures. The problem comes first from memory excess, since the administrator of the PIGA system is not really time restricted to compute the signatures. The solution chosen in this case is to bound the length of the signatures. This approximation is acceptable for some systems, since the probability of defining an effective attack from a signature become more and more smaller with its length.

It is thus a challenge to find a good way to compress the signatures in order to lower the size of the storage required by the signatures. In the PIGA system, a simple compression is performed: the signatures are stored using a tree. After this compression, the size of the input signatures remains large.

Another challenge is to obtain a compression not impacting the efficiency of the detection process. For example, if the system only stores the pairs (source, target), the detection would be preponderant and slow down the system significantly.

This paper tackles the first problem of finding a generic way to compress the number of signatures. An important point is to ensure that any signature generated by the original policy is included in the set of compressed signatures. We will see that the reverse property is not required (any compressed signature only represents valid signatures). This comes from the specificity of the objects manipulated in this context. However, this paper tries to find the compressed set of signatures that minimises the number of unrealistic signatures in a compressed signature. Finally, the solution presented in this paper follows the philosophy of PIGA: the generation of a set of signatures using a graph that represents the system.

Since the signatures are simple paths in the flow graph, reducing the size of this graph should reduce the number of the signatures. In order to reduce the size of this graph, the idea is to group the nodes having the same behaviour into a single *meta-node*. For example, a simple analysis on the flow graph in Figure 2 shows that both contexts `ssh_d` and `login_d` have the same behaviour considering the other contexts. The idea developed in the remaining of this paper is to consider these two contexts as a single entity called `module`. Assuming this in this example, only two signatures should be considered.

```
shadow_t-> passwd_d-> module-> user_d
shadow_t-> passwd_d-> module-> admin_d-> bin_t-> user_d
```

This kind of property for a graph is known as the modular decomposition. To obtain an interesting compression, the input interaction graph must contain a significant number of modules. The modules should not be too large in order to maintain the detection process efficient. The goal of this paper is to evaluate the compression ratio of the application of the modular decomposition to realistic flow graphs. The following section presents the theoretical base of our technique.

IV. Simplification of the input graph

The reader may refer to [9] for advanced concepts in graph theory. In this section, we first recall the definition of the modular decomposition on general undirected graphs in Section IV-A. Since the input graphs are directed, we show how to deal with this problem in Section IV-B. Section IV-C presents the general methodology for compressing signatures. In the original algorithm for computing the signatures, the signatures are defined as the set of the paths between the source and target security contexts. However, the decomposition process compacts some nodes into some *meta-nodes*, called *modules*. Thus, some source or target security contexts may disappear in this process. Section IV-D describes the process to make these contexts reappear. Finally, in Section IV-E, we provide an evaluation of the reduction of the number of signatures using this technique.

A. Modular decomposition

The modular decomposition of a graph is a useful tool to represent graphs in a compact way by grouping nodes that have the same behaviour. Theoretical and algorithmic aspects of this decomposition are explored in [10]. This decomposition has several applications. For example, it has been used in graph drawing, where a module can be seen as a single node and indeed simplify the drawing [11]. This decomposition has been used in biology to simplify the protein-protein interaction network and obtain comprehensive view of this network [12]. All these examples show that the modular decomposition is used to simplify the input graph and then lower the complexity of solving initial the problem. In this paper, we use this type of graph decomposition to lower the size of the flow graph on which the signatures are computed.

Definition 1: Let $G = (V, E)$ be an undirected graph. A **module** M of G is a subset of V such that:

$$\forall x \in V \setminus M \begin{cases} \forall y \in M, (x, y) \in E \text{ or} \\ \forall y \in M, (x, y) \notin E \end{cases}$$

Let G_M be the subgraph of G such that $G_M = (M, E')$ such that: $E' = \{(x, y) \in E \mid x, y \in M\}$. As shown in [10], module M can have one of the three following types, depending on the connectivity of G_M:

- M is series if $\overline{G_M}$ is not connected;
- M is parallel if G_M is not connected;
- M is prime otherwise.

A module M is **strong** if for any other module M', either one is included in the other ($M \subseteq M'$ or $M' \subseteq M$) or they are totally disjoint ($M \cap M' = \emptyset$). The set of strong modules can be organised in a tree, called the **modular decomposition tree**, giving some hierarchical relationship between the modules. The root of this tree is the trivial module V and the leaves are the other trivial modules: the single nodes. The modular decomposition tree of G is designated by $MDTree(G)$.

Figure 3 represents H the symmetrised version of the flow graph in Figure 2, where every arc between two nodes has been replaced by an edge between these nodes. The application of modular decomposition on H generates the modular decomposition tree $MDTree(H)$ represented on Figure 4.

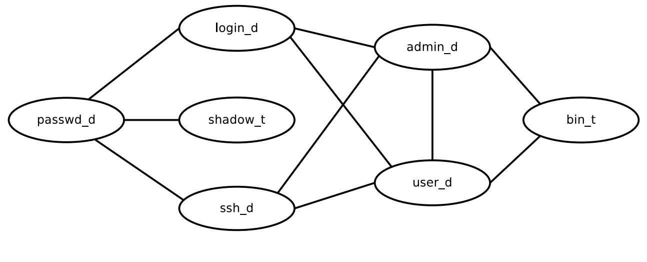


Fig. 3. The flow graph after symmetrization $H = (V, E)$

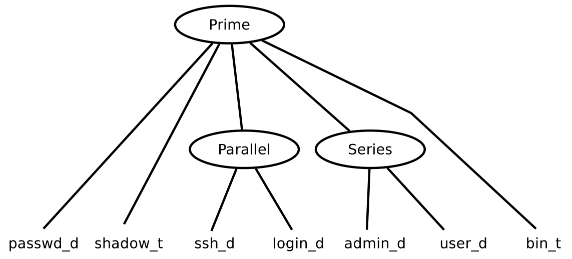


Fig. 4. The modular decomposition tree associated to H, $MDTree(H)$

A **quotient graph** can be obtained by grouping the nodes contained in the same module into a single node in the graph. The quotient graph of G is called $Quot(G)$, e.g., the graph $Quot(H)$ represented on Figure 5 is the quotient graph of H. This operation makes the input graph smaller and thus easier to understand. In the following definition, we extend the notion of quotient graph to any partition of the vertices. This latter definition is also valid for directed graphs.

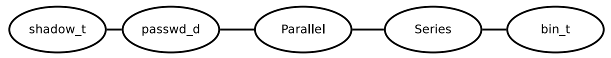


Fig. 5. The quotient graph associated to H, $Quot(H)$

Definition 2: Let $G = (V, E)$ be a graph and $\mathcal{V}$ a partition of V. The graph $Quot(G, \mathcal{V})$ is defined as follows:

- The vertex set is $\mathcal{V}$;
- The edge set $\mathcal{E}$ is given by:

$$\mathcal{E} = \left\{ \begin{array}{c} (V_1, V_2) \mid V_1, V_2 \in \mathcal{V} \text{ and } V_1 \neq V_2 \\ \text{and } \exists\ v_1 \in V_1, v_2 \in V_2\ (v_1, v_2) \in E \end{array} \right\}$$

Using this definition, the quotient graph in the modular decomposition context is simply $Quot(G, \mathcal{V})$ where $\mathcal{V}$ is the partition obtained from the modular decomposition tree considering the nodes at level 1.

The partition $\mathcal{V}$ obtained by selecting the nodes at the first level of the $MDTree(H)$ represented by Figure 4 is:
$\mathcal{V} =$ (`passwd_d`, `shadow_t`, (`ssh_d`, `login_d`), (`admin_d`, `user_d`), `bin_t`).

B. Dealing with directed graphs

The graphs considered in this paper, i.e., the policy and flow graphs, are directed. The modular decomposition can also be applied to directed graphs. F. De Montgolfier [13] uses 2-structure to define the module in this context. However, we only use the undirected concepts for several reasons.

First, as shown in Section II-E, the interaction and flow graphs are used for generating the signatures. They have the same vertices, however, some edges have different directions. Applying directed modular decomposition on both graphs may generate two different modular decompositions. Consequently, the quotient graphs would be different and it would be more difficult to generate the set of signatures.

Second, the sizes of the modules in directed modular decomposition are smaller than in the undirected case, since any directed module remains a module after transformation of the arcs into edges. Considering the initial example given in Figure 2 and its associated flow graph, the series module in the undirected decomposition is not a module in the directed case: the neighbourhood of `admin_d` and `user_d` are not the same in both graphs.

Finally, as shown at the end of this section, this action can be minimised by recovering a directed quotient graph, our process generates new dummy signatures that are useful to efficiently compress the signatures.

C. Overall strategy

In this section, we present our global methodology. The input of our algorithm is the interaction graph $G = (V, A)$.

In order to use the modular decomposition, we consider the associated undirected graph $G' = (V, E)$ obtained by modifying any arc (i, j) in G into the edge $\{i, j\}$ in G' and $Quot(G')$ its quotient graph. Since the generation of the signatures uses a directed graph, we compute the directed quotient graph $Quot(G, \mathcal{V})$ using the partition $\mathcal{V}$. $\mathcal{V}$ is generated by the computation of $Quot(G')$, i.e., a module of $Quot(G')$ is considered as a module of $Quot(G, \mathcal{V})$. The main difficulty is to compute the useful quotient graph depending on the source and target security contexts. Figure 6 represents the quotient graph obtained from the directed graph G in Figure 2.

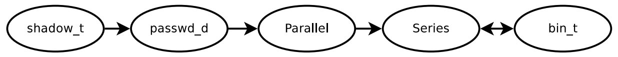


Fig. 6. The directed quotient graph associated to the flow graph Figure 2

In Section V, we show that real graphs in security domain contain several modules. They are mainly parallel modules. We show in the following section how to exploit this property for the computation of compressed signatures.

D. Computing the quotient graph for one pair source/target

In this section, we detail the extraction of the compacted signatures from the original graph. As described in Section II-E, the signatures are computed as the set of the simple paths from the source context to the target context. Using the modular decomposition, we compute the new signatures using the quotient graph. However, the source (respectively, target) context may be contained in modules. Thus, the endpoints of the paths should be considered as a module by themselves, breaking the modules in which they are contained into smaller modules. For this, we can see that the entire module can be removed and all the elements inside considered as trivial modules. However, as shown in the Algorithm 1 some submodules can be preserved.

Note that a node appears in the quotient graph if it is a direct child of the root of the modular decomposition tree, i.e., if it appears as a singleton in the corresponding partition. The aim of the algorithm is indeed to find a partition where both endpoints are singletons while keeping the largest number of modules. We called this partition $\mathcal{P}_{s,t}$ where s is the source and t the target. Thus, to extract a node from a module we need to transform the modular decomposition tree such that the

Algorithm 1 Extraction of a node

```
Input: n the node to extract
Output: decomposition tree with the extracted node
  for each: node m in path(n) do
    removeEdge(Parent(m),m)
    if m is not Root and (ChildCount(m)<=2 or m is Prime)
    then
      for each: node o in child(m) do
        removeEdge(m,o)
        addEdge(Root,o)
      end for
      removeNode(m)
    else
      addEdge(Root,m)
    end if
  end for
```

endpoints are direct children of the root and the tree remains a modular decomposition tree. Algorithm 1 solves this problem in the following way. Every node in the path from the root to the extracted node is a module. We can separate these modules into two categories. The first contains modules with more than two children in the tree. The second contains modules with exactly two children. Modules of the first category are put as a direct child of the root and they keep all their children but the child included in the path. Modules of the second category are removed from the tree and every child is put as a direct child of the root. Finally, the node that we want to extract is put as a direct child of the root. Special care has to be taken for prime modules. Indeed, a parallel (respectively, series) module having more than two elements can be decomposed in a parallel (respectively, series) module having exactly two elements, one node and a parallel (respectively, series) module that contained the remaining nodes. This property is not preserved for prime modules and a prime should be totally broken and its children must be put at the root level. The partition $\mathcal{P}_{s,t}$ is obtained by applying Algorithm 1 twice, with s and t.

Note that Algorithm 1 is very efficient since the modifications of the tree are located along the path between the root and the leaf corresponding to the node to extract. Thus, the worst case complexity is linear in the size of the tree. Consequently, the main time consuming task remains the computation of the signatures from the quotient graph. Nevertheless, the global computation time is reduced significantly.

As an example, we apply the modular decomposition on the graph depicted in Figure 7. This undirected graph contains 11 vertices and 29 edges. The algorithm generates the modular decomposition tree of Figure 8. This tree is composed of four levels and contains one prime, two series and three parallel modules. From this tree, we generate the quotient graph, in Figure 9, that contains the nodes of the first level of the tree. Computing the compressed signatures between nodes 1 and 5, we can use this quotient graph, since these nodes are not contained in non-trivial modules. There are two

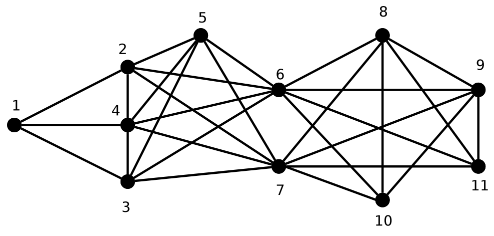


Fig. 7. Initial graph GE

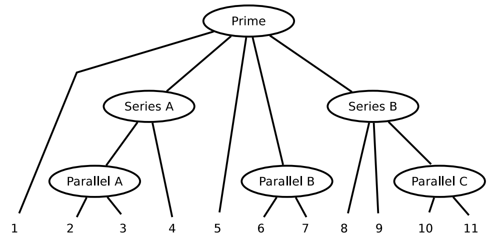


Fig. 8. Decomposition tree $MDTree(GE)$

signatures from Node 1 to Node 5 in the initial graph, there exists 2027 signatures for the same source/target pair. If the security property involves Nodes 2 and 8, this quotient graph cannot be used. Algorithm 1 computes the modified modular tree (Figure 10) and the resulting quotient graph is given in Figure 11. This graph contains 8 nodes and 15 edges. It results 50 compressed signatures instead of 2610 in the initial graph.

E. Reduction of the number of signatures

In this section, we evaluate the compression rate: the number of signatures represented by a **modular signature**, i.e., a signature that contains modular contexts.

We assume in this section that the compression process using the modular decomposition does not add unwanted arcs.

Definition 3: Let $G = (V, A)$ a directed graph and $Q = (M, A')$ a quotient graph of G. An **arc** (m_1, m_2) **is clean** iff

$$\forall x \in m_1, y \in m_2, (x, y) \in A.$$

The **quotient graph** Q **is clean** if all its arcs are clean.

A **signature** s **is clean** if it only contains clean arcs.

Lemma 1: A compressed clean signature s that contains one modular context M of size k represents at least $f(k)$ non-modular signatures, where $f(k)$ is the sum of:

- k, i.e., the size of the module, and
- the number of simple paths between two distinct elements in the module within G_M, i.e., considering only the paths inside the module.

Proof: Let s be a compressed signature that contains one modular context, i.e., $s = c_1, \ldots, c_i, M, c_{i+1}, \ldots, c_l$, where M is a module in G, $M = \{c'_1, \ldots, c'_k\}$. Using these notations, all the c_i and c'_i are distinct. Otherwise, there would be a loop in the signature or one of the c_i would belong to the module and should have been replaced by M in s.

From this signature we can derive those in the original graph in two different ways. First, we can replace M by one of the elements of the module. This lead to k signatures of the same length as s. Second, we can replace M by a single path between two distinct elements in M, leading to longer signatures and to the second part of the sum. ■

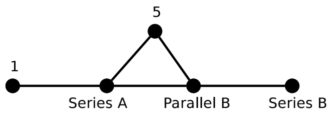


Fig. 9. Quotient graph $Quot(GE)$

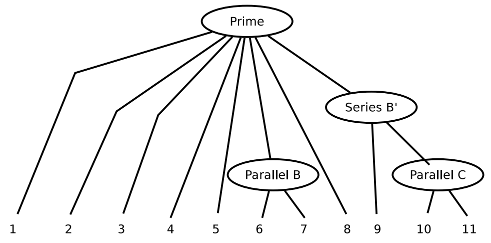


Fig. 10. Decomposition tree with source and target extract

In order to compute the computation ratio of a single signature for a general module, we need to make some complex computations. However, for parallel and series *pure* modules, this can be solved easily. We say that a module is *pure* if all its children in the modular decomposition tree are leaves.

Consequence 1: A compressed clean signature s that contains one modular *pure* parallel context M of size k represents at least k signatures in the original policy.

Proof: In this case, no path between two distinct elements exist within the module. ■

Consequence 2: A compressed clean signature s that contains one modular *pure* series context M of size k represents at least N signatures in the original policy, where

$$N = \sum_{i=0}^{k-1} \prod_{j=0}^{i} (k - j)$$

Proof: In this case, the nodes contained in M form a clique. Then the number of simple paths between two elements of M is at least the number of paths inside the clique, including the paths of length 0. Defining a simple path of length i consists in choosing the first element within k, then the second within $(k - 1)$, and so on. Thus, the number of simple paths of length i is $\prod_{j=0}^{i}(k - j)$. By summing all these terms, we obtain the desired result. ■

If a signature contains several modules, the number of uncompressed signatures corresponds to at least the product of the number of compressed signatures for each module.

Note that a compressed non-clean signature represents some signatures that are not possible in the original graph. These unrealistic signatures are not significant for the detection process. Indeed, such uncompressed signature cannot be activated since some transitions are not allowed by the MAC policy (e.g., SELinux) of the system.

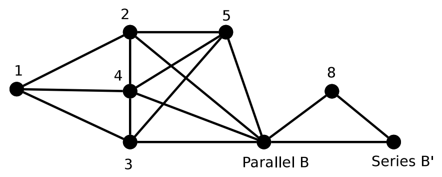


Fig. 11. Quotient graph $Quot(GE, \mathcal{P}_{2,8})$

	Nb sig	ρ	Min ρ	Max ρ
Uncompressed	103411	/	/	/
Compressed without loop	1436	98.6%	91.3%	99.9%
Compressed with loops on modules	5780	94.4%	81.7%	99.7%

TABLE I
COMPRESSION RATIOS FOR THE EXAMPLE GRAPH

```
p1: user_u:user_r:user_t --> system_u:object_r:shadow_t
p2: user_u:user_r:user_t --> system_u:object_r:etc_t
p3: user_u:user_r:user_t --> user_u:object_r:user_tmp_t
p4: system_u:object_r:shadow_t --> user_u:user_r:user_t
p5: system_u:object_r:etc_t --> user_u:user_r:user_t
p6: user_u:object_r:user_tmp_t --> user_u:user_r:user_t
```

TABLE II
AUDITED PAIRS

	p1	p2	p3	p4	p5	p6	Total
Uncompressed	1	2	14006	85510	42756	10238	152513
Compressed	1	2	477	4026	2014	350	6870
ρ	0%	0%	96.6%	95.3%	95.3%	96.6%	95.5%

TABLE III
COMPRESSION RATIOS FOR THE FLOW GRAPH

V. EXPERIMENTS

The experiments have been performed on one example graph and two graphs generated from a realistic security policy. The following sections present the structure of the studied graph and the results of our experiments for each studied case. In order to evaluate the efficiency of our method, we used the following compression ratio:

$$\rho = 1 - \frac{\#\text{compressed signatures}}{\#\text{uncompressed signatures}} \qquad (1)$$

A. Results on the example graph

The example graph (Figure 7) contains 11 nodes. This example has no application in the security domain and comes from papers dealing with modular decomposition theory [10]. However, it contains all the types of modules (Figure 8) and appears to be important in order to validate the global strategy for computing the signatures and compressed signatures.

We compute the number of signatures generated by the application of the modular decomposition for each source/-target pair. The results, presented in Table I, show that the compression is very high. Note that a compressed signature represents on average more than 70 uncompressed signatures.

We also consider more complex signatures: we allow the signature to contain loops for modular nodes. These signatures take into account the fact that a module contains several nodes. However, we use the following rule: a signature cannot contains more than k occurrences of a same module if the size of this module is k. Even in this case, the compression ratio remains high. This investigation shows that the modular decomposition strategy has a great impact even if the sizes of the modules are not large and for any pair of source/target nodes.

Since for each pair, our strategy computes a new graph on which the signatures are generated. It appears that the graphs contain between 5 and 9 nodes.

B. Results on a global SELinux policy

In order to evaluate the efficiency of our method on a real case, we study the following flow graph. It represents the possible information flows in a gateway using a Gentoo Linux distribution studied in [3]. It contains 43 nodes and 163 arcs. The decomposition tree has a special structure: it only contains three levels. The root is a prime node; at level 1, there exists 10 parallel modules and 16 leaves and the last level contains 27 leaves. Each parallel contains from 2 to 5 nodes. From a security point of view, each parallel has some semantic consistency. For example, one parallel contains `system_u:object_r:shadow_t`, `user_u:object_r:shadow_t` and `root:object_r:shadow_t`.

We compute the number of signatures generated on 6 source/target pairs given in Table II. The graph and the pairs were used by an instance of PIGA-IDS deployed on the gateway of a honey-pot [3]. The aim of this honey-pot was to understand the behaviour of any user considered *de facto* as an attacker. The first pair (p1) in Table II would detect all the attempts of changing the password of the user, whereas p4 detect all the attempts of the user to obtain information from the shadow file. All the pairs are using the context `user_u:user_r:user_t`, thus there are only four contexts used as endpoints. From these four contexts, only the context `system_u:object_r:shadow_t` is contained in a parallel module. Thus, all the pairs p2, p3, p5 and p6 will use the basic quotient graph for the computation of the compressed signatures, the two remaining ones use a quotient graph having 27 nodes and 91 edges. The quotient graphs are clean (see Definition 3). This implies that there are only clean signatures generated by the compression process.

Table III shows the number of signatures generated with and without using the modular decomposition process. This table also gives the compression ratio of the application of our method. A further analysis of the distribution of the compression ratios shows that a modular signature compresses between 2 and 717 non-modular signatures. Moreover, the number of modules in a signature varies between 1 and 3, some modules cannot been reached by the source element.

C. Results on the transition policy

As described in Section II-D, from the interaction graph, we can derive several others by filtering the labels of the arcs. The transition graph is obtained by removing the arcs that are not labelled by a transition and removing the isolated nodes. This section provides a study of the transition graph obtained from an interaction graph having 577 nodes.

This transition graph has 381 nodes and 21074 arcs (density 15%). The application of the modular decomposition generates a tree with a series module as root. This tree is deeper and more complex than the modular decomposition tree of the flow graph. At level 1, the leaves represent contexts that have some connection with all the other nodes. They are of the form `*:sysadm_r:sysadm_t` and they are related to the administrator of the system that has all the rights. Then the quotient graph is very simple and contains 5 nodes and 20 arcs corresponding to a complete graph. Note that a prime module at level 4 contains 108 of the 112 nodes at the level 5. The decomposition tree present another characteristic: the series modules contain from 3 to 7 nodes. The nodes contained in the same module have some similarities in their label. For example, all the contexts `*:*:passwd_t` form one series module of size 7. It shows that the modular decomposition reveals some logical coherency of the transition graph.

We compute the number of signatures from `system_u:system_r:sshd_t` to `user_u:user_r:user_t` on the transition graph using the modular decomposition. This computation defines all the possibilities of connections from `ssh` to a user account. We obtain 3546 signatures by limiting their length to 4. The corresponding quotient graph contains 115 nodes and 1515 arcs, 92 of them being not clean. In the original graph, it was not possible to compute all the paths of length 4 using our generation program due to memory overflow. This is a consequence of the search of the paths in the series modules that generate many signatures, as shown in Consequence 2. In this example, it was not possible to compute all the compressed signatures of length 5. To complete our experiment, we restrict the length of the paths to 3, we obtain 190 compressed signatures and 1571 uncompressed signatures, leading to a compression ratio of $\rho = 87.9\%$.

We also made the same experiment restricted to the nodes included in the prime module. In this case, the initial graph has 350 nodes and 5930 arcs, while the quotient graph used for the computation of the signatures has 108 nodes and 506 arcs, all of them being clean. The computation of the compressed signatures have been performed up to paths of length 10 where 111,704 signatures have been found, while the uncompressed signatures have been computed up to length 4. For paths of length 4, there exists only 96 compressed signatures and the compression ratio is 99%.

VI. Conclusion and future work

In this paper, we have presented a way for compressing the signatures base used in PIGA IDS. We experiment this procedure on different input graphs. These experiments show that our method is efficient for compressing signatures expressing the confidentiality property: the compression ratio is usually large, over 90%. This reveals a characteristic of the interaction graph and its derivatives. Indeed, many of the modules have a logical coherency and labels inside a same module have some similarity. In some extent, we can say that modules represent some meta-contexts and that the signatures are defined at this level.

This method can be also applied to other security properties.The study on the transition graph shows that any property implying this particular graph would lead to a very high compression ratio since all the types of modules are found and the decomposition tree is deep.

The compression technique provides another result: the expressiveness of compressed signatures is very high. Indeed, compressed signatures of small size can represent very long signatures. Furthermore, since the quotient graph is smaller than the original graph, our method can compute longer signatures than the initial computing method.

The main perspective to this work is to define the detection counterpart. Indeed, the process has to be adapted in order to deal with modules in order to eliminate the false positive induced by the compaction. The additional cost due to the compression has to be clearly evaluated before being installed in a real operating system.

References

[1] J. Goguen and J. Meseguer, "Security policies and security models," in *IEEE Symposium on Security and Privacy*, 1982, pp. 11–20.

[2] H. Mantel, "A uniform framework for the formal specification and verification of information flow security," Ph.D. dissertation, University of Saarland, 2003.

[3] J. Briffaut, J. Rouzaud-Cornabas, C. Toinard, and Y. Zemali, "A new approach to enforce the security properties of a clustered high-interaction honeypot," in *Workshop on Security and High Performance Computing Systems*, R. K. Guha and L. Spalazzi, Eds. Leipzig, Germany: IEEE Computer Society, June 2009, pp. 184–192.

[4] S. Chari and P.-C. Cheng, "BlueBoX: A policy-driven, host-based intrusion detection system," *ACM Transactions on Information and System Security*, vol. 6, no. 2, pp. 173–200, May 2003.

[5] M. Bernaschi, E. Gabrielli, and L. Mancini, "Remus: a security-enhanced operating system," *ACM Transactions on Information and System Security*, vol. 5, no. 1, pp. 36–61, Feb. 2002.

[6] G. Portokalidis, A. Slowinska, and H. Bos, "Argos: an emulator for fingerprinting zero-day attacks for advertised honeypots with automatic signature generation," in *Proceedings of the 1st ACM SIGOPS/EuroSys European Conference on Computer Systems 2006*, ser. EuroSys '06. Leuven, Belgium: ACM, 2006, pp. 15–27.

[7] H. Mantel, "The framework of selective interleaving functions and the modular assembly kit," in *Proceedings of the 2005 ACM workshop on Formal methods in security engineering*, ser. FMSE '05. New York, NY, USA: ACM, 2005, pp. 53–62.

[8] Department of Defense, "Trusted Computer System Evaluation Criteria," Department of Defense, Technical Report DoD 5200.28-STD, 1985.

[9] J. Bondy and U. Murty, *Graph Theory*, ser. Graduate Texts in Mathematics. Springer, 2008, vol. 244.

[10] M. Habib and C. Paul, "A survey of the algorithmic aspects of modular decomposition," *Computer Science Review*, vol. 4, no. 1, pp. 41–59, Feb. 2010.

[11] C. Papadopoulos and C. Voglis, "Drawing graphs using modular decomposition," *Journal of Graph Algorithms and Applications*, vol. 11, no. 2, pp. 481–511, 2007.

[12] J. Gagneur, R. Krause, T. Bouwmeester, and G. Casari, "Modular decomposition of protein-protein interaction networks," *Genome Biology*, vol. 5, no. 8, p. R57, 2004.

[13] R. McConnell and F. de Montgolfier, "Linear-time modular decomposition of directed graphs," *Discrete Applied Mathematics*, vol. 145, no. 2, pp. 198–209, Jan. 2005.

A Data Centric Security Cycle Model for Data Loss Prevention of Custodial Data and Company Intellectual Property

Mathew Nicho
College of Information Technology
University of Dubai
mnicho@ud.ac.ae

Avinash Advani
Boole Server
Milano, Italy
a.advani@booleserver.com

Abstract: **Review of data breach trends in the last five years reveal that data at rest, in use, and in motion, inside and over the extended network, is being increasingly affected. While organisations primarily focus on protecting sensitive customer financial information, the protection of custodial data and company secrets has been a back burner issue. Moreover, errors, mistakes and accidents on the part of the employees working/ travelling/ residing onsite and off-site with company media/data, have worsened the situation such that current technical and socio-technical controls are not adequate in preventing theft of media or the accidental or intentional misuse/loss of portable data. To overcome this issue, the security action cycle model of Straub and Welke (based on the general deterrence theory) is used as a theoretical lens to build a data centric security cycle model to safeguard the data that are "at rest, in motion and in use". Finally, the paper discusses how the model can be further empirically validated using the updated IS success model of DeLone and McLean.**

Keywords-IS Security; data breaches; data centric security.

I. INTRODUCTION

Security and privacy has remained one of the top ten key issues for Information Systems [IS] executives since 2003 [1] and crucial to the continuous wellbeing of modern organisations [2] with the result that organizations need to protect information assets against cyber crime, denial-of-service attacks, web hackers, data breaches, identity and credit card theft, and fraud [3]. A firm's information related assets are now among its most valuable assets [4]. Thus, the protection of this asset through the process of information security, is of equal importance [5]. The application of existing technical IS security frameworks and IS controls have been effective in preventing attacks from external entities into the organizational networks, but the mobility of the organizational staff and information technology (IT) assets along the extended network have posed serious risks to organizational data.

Inside organizations, valuable and critical data may exist in laptops, portable storage media, storage media at distant locations, and in emails in online and offline mode. In this instance, sensitive data is not only out of the organizational security defenses but that the existing organizational IS controls and assurance cannot be monitored and thus becomes voluntary on the part of the employees to protect it from theft, accidental loss and misuse. While information security controls and models exist for securing the organisational network, data loss prevention is an area of scant research.

The object of this paper is data, which primarily refers to custodial data (protection required by regulation/law) and company secrets (high value intellectual property and assets). The objective of the paper is to propose a model to protect and control this data. *Protection*, focus on the data in its raw form as it rests in the file system, flows through the network and while it is being used. The *controlling* aspect considers how the data or information is used once authorized users have gained access to it and even revokes the access to it. The output of this paper is a data centric security cycle model to protect the data (hereinafter refer to data that are "at rest, in motion and in use") in real time. The theoretical basis of the model rest on the security action cycle model of Straub and Welke [6], which in turn is derived from the general deterrence theory.

The paper is divided into five sections. The current state of IS security breaches is analyzed from statistics to highlight the relevance of data loss prevention (Section 2). This is followed by analyzing the high profile data breaches occurred during the last five years from 2007 to 2011 to find out the gaps in the socio-technical security structure (Section 3). Then, the current literature on IS security is evaluated to ascertain the existence and appropriateness of relevant IS security models for data protection and to select an appropriate theoretical basis for the model (Section 4). Finally, the selected theory is employed to build up the proposed Data Centric Security Cycle (DCSC) model, as well as provide the rationale for using the updated IS success model of DeLone and McLean [7] for evaluating the success of the model (Section 5).

II. DATA BREACH ANALYSIS

The central objective of any security system is the ability to prevent undesired access, while still allowing authorized access to information [8] but with cyber incidents growing in intensity and severity [9] the risks related to information security have become a major challenge and a top management priority for many organizations [10]. Thus, despite the critical role and relevance of information and information security in an organisation, unauthorized breaches into organisational internal and the extended networks occur with greater frequency and severity [6] [9] [11] [12]. IS security have remained within the top ten key issues in information systems since the advent of the Internet in the early 1990s [13], and maintained this position in subsequent studies [1] [14] [15] [16] [17]. This section analyses data breaches from a statistical as well as from a methodological perspective to ascertain the severity and cause of breaches.

A. Data Breaches – a Statistical Perspective

As organisations rely heavily on information to conduct their daily activities [5], any disruption or intrusion poses grave threat to the organisation and its extended enterprise. Recent statistics taken from different sources reveal the seriousness of the issue. In the annual 2010/2011 CSI computer crime and security survey (285 respondents), 41.1% of those surveyed reported a security incident in their company [18]. Similarly, a 2011 sector wise study by Deloitte on 138 organizations revealed that 75% experienced an IS security breach, which was an increase over the previous year by 62% [19]. Also the average total cost of a data breach rose to $6.75 million in 2009, with major increase in number of records lost per incident [20] where stolen laptops were cited as the number one cause of a data breach in 2009.

From a numerical perspective, in 2011, the US Identity Theft Resource Centre (ITRC) reported a total of 419 breaches resulting in 22,918,441 records being compromised [21]. Compared with ITRC, Datalossdb [22] reported 890 breaches in 2011 with insiders (accident and malicious) accounting for 39% of the breaches. Among the three types of breaches namely reported abuse, discovered abuse and undiscovered abuse [23], the above figures mainly represent only the reported abuse all of which involve substantial loss to the stakeholders. The breaches identified by the various sources are not uniform as they use newspaper articles, copies of letters reporting a breach to consumers, notification lists of state agencies, direct entry of incidents by the public, and other web sites as sources for the breaches [24].

ITRC statistics that are based on publicly available breaches in United States provides categorized statistical data on breaches (see Figures 1 and 2). Considering accidental exposure and data on the move, statistics from 2007 till 2011 does not indicate any drastic reduction in either the number of breaches or records breached.

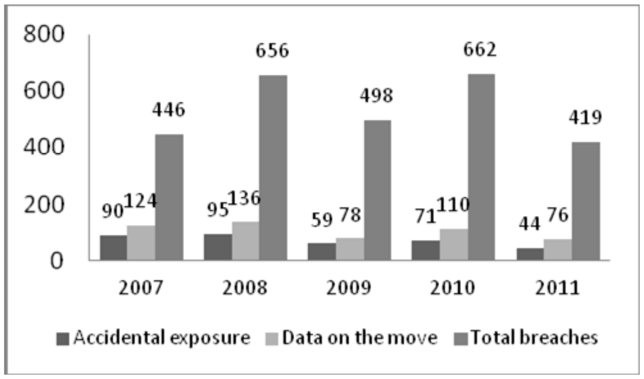


Figure 1. Total number of reported/known breaches in US from 2002 to 2011 (Source: ITRC, 2007 - 2011)

While payment card information and authentication credential are still the most sought after data, the largest breaches (in terms of the number of records) reveal that the personal information along with company secrets are a potential target (Figure 3). The above statistical figures reveal the increase in intensity and frequency of threat to all financial and organizational data from within and external to the organization.

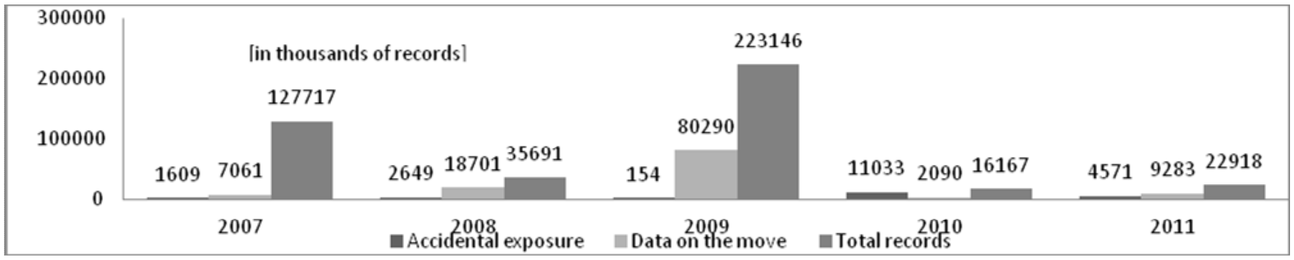


Figure 2. Total number of records breached from 2007 to 2011 from reported/known sources in US I (Source: ITRC, 2007, 2008, 2009, 2010, 2011)

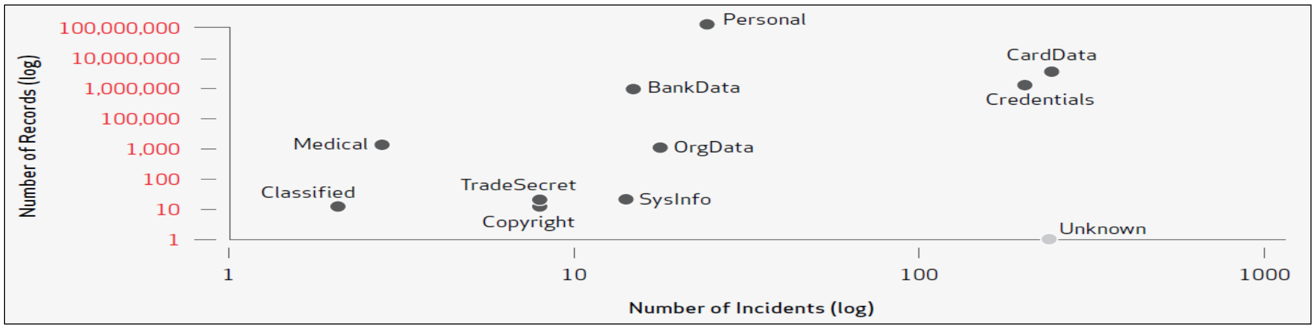


Figure 3. Varieties of data compromised by number of breaches and records [25]

Information can be categorized into custodial data and secrets [26]. It has been noticed from the above statistics that hackers have shifted from financial information to the wider custodial and company secrets (see TABLE I). Irrespective of the type of date, the cost incurred by the organisation comes at a price.

B. *Cost of Data Breach*

While IS security breaches cost dearly for the organizations, the question is not whether organizations need more security, but to look at cost-benefit methods to evaluate IT security so as to 'optimize' security countermeasure investments and reduce spending without sacrificing protection (Arora et al., 2004). The financial loss suffered by US companies average $ 5.1 million for a single data breach and the cost incurred for one compromised record comes to $214 [27]. Taking the 2011 ITRC reported breaches into consideration, this would amount to $ 2136 million (at the rate of US $ 5.1 million x 419 breaches). Calculating the loss from a records compromised, the loss amounts to $ 4904 million (22.9 million records based on ITRC x $ 214). Verizon (2012) reported 855 incidents with 174 million compromised records in 2011 from the small global sample from the 33 participating countries. If this statistic is taken to calculate the loss, it comes to an annual loss of $ 37,236,000,000 from this small sample.

C. *Human Factor and Mobility in Data Breaches*

The human being remains the weakest link in the control and security of systems and networks [28] and "frequently security violations involve those who are authorized or have access to the sensitive data of concern" [29] (p. 26). Moreover, it is estimated that at least half of the breaches to IS security are unauthorized system access made by internal personnel [30]. Thus, the involvement of humans in information security is equally important and many examples exist where human activity can be linked to security issues [2]. As noted by Schultz [31], information security is primarily a people problem and technology is designed and managed by people, leaving opportunities for human error. It has been observed that organisational security target the prevention of external threats, such as hackers and viruses, leaving organizations open to breaches from the inside [30]. Hence, the occurrence of IS security breaches by internal personnel may be reduced if greater emphasis were placed on internal threats to IS security that can occur when employees handle information in their day-to-day jobs (ibid).

The environment of today's worker is evolving from centralization and control to mobility and performance [32]. With the rapid adoption of mobile office, modern organizations are exploiting mobile media and their infrastructure in a more strategic manner, thus developing work styles and office designs that are evolving around new technologies like mobile phone, laptop and email address [33]. While these technological advances provide advantages to organizations [34], all of these pose significant threat to information. This mobility highlights the relevance of *protection* to the data, focusing on the data itself and evaluate how sensitive data and information can be securely delivered and *shared/transferred* beyond the organisational network. This calls for a data focused security model that can incorporate the policies and *controls* into the technical security architecture to protect data from unauthorised access, wherever it resides (fixed infrastructure, end points and mobile devices).

III. CAUSES OF DATA BREACH

The year 2011 witnessed a spate of high profile cyber-attacks, and the top ten reported cases (number of records breached) in the US and a few organizations in Europe is taken to analyze the gaps in IS security. Since the ITRC and the Privacy Rights Clearinghouse (PRC) [35] documents publicly available breaches in US, these sources were combined and analyzed to come up with a list of top ten data breaches in 2011.

TABLE I. EVALUATION OF THE TOP TEN BREACHES **(**Source: ITRC, 2012; PRC, 2012)

	Organisation	**Records**	**Nature of breach**	**Evaluation**
1	Sony Playstation	70,000,000	Stolen information include - name, address, country, email address, birthdate, PlayStation Network/Qriocity password and login.	External attack - the hacker used spear phishing rather than highly sophisticated hacking to break into the network.
2	Zappos (owned by Ama-zon.com)	24,000,000	Customer information - names, email, billing and shipping addresses, phone numbers, the last four digits of credit card numbers, and crytographically scrambled passwords were stolen	External attack – where the hacker gained entry into the company server (Methodology is not known as the probe is still ongoing).
3	TRICARE-SAIC	4,900,000	Backup tapes containing SAIC SAIC data stolen from the car of a Tricare employee.	Non-technical – employee error; procedure not followed
4	Texas Comptroller	3,500.000	Unencrypted information transferred and kept in a server accessible to the public	Correct procedure not followed when trans-ferring information across servers
5	Betfair	2,300,000	SQL injection attack on a code vulnerability	*Technical breach*, procedure not followed especially network segregation and file integrity monitoring.
6	Health Net IBM	2,000,000	Nine server drives went missing from the data centre of the California office of Health Net	Appropriate policies and procedures have not been followed by those responsible for both the physical and logical protection of critical data.
7	Jacobi Medical Centre	1,700,000	The files (cassette tapes in a box) was stolen from a van operated by GRM Information Management Services, when the driver left the van unattended and unlocked.	Correct procedure not followed prior to and when transporting storage media
8	Nemours Children Health System	1,600,000	Unencrypted computer backup tapes containing patient billing and employee payroll data stolen from a Nemours facility in Wilmington, Dela-ware	Correct procedure not followed when stor-ing storage media. As per the control, the tapes were supposed to be safely locked
9	Oregon Department of Motor Vehicles	1,000,000	USB or CD containing personal information lost from the department. Thief caught.	Correct procedure not followed when stor-ing/disposing redundant data
10	Eisenhower Medical Centre	514,330	The computer used to check-in patients at the Center in Rancho Mirage was stolen from the open lobby area	Correct procedure not followed. The computer was not protected with drive encryption nor physically locked.

Table I reveals theft/accidental loss of media as a common cause of breach rather than hacking into the company network. In the above ten cases, the data stolen mostly contained personal and company information, rather than credit cards or financial data which can be sold to third parties or used for further social engineering attacks. Secondly, in all of these top ten breaches, only cases 1, 2 and 5 attacks came from external parties into the organizational network for which the IS security manager have control of. In these three cases, a simple analysis of the IS security defenses and improved IS security can prevent further attacks. Regarding the rest seven cases (3, 4, 6, 7, 8, and 9), data loss/theft occurred with ease, and since the data was in media out of the organizational perimeter defense it would have been impossible to track with the normal internal controls. These cases prove that the IS security manager need to have more control over the data that are at rest, in motion and in use. Thus, IS security should not only be built like a staircase of combined measures in order for information security measures to become effective [36] but mutually dependent on each other [37] Berghel, 2005 cited in [36] Hagen, Albrechtsen and Howden.

The cases analyzed in this section highlights the relevance of, *control* and *changing access* to information in real time even after it has been sent out to the extent of even restricting functions on secured documents, e.g. print, copy, save as, print screen. If the data is outside the authorized perimeter then the manager should have the right to *revoke* access instantly, even after delivery has been taken, no matter where it has ended up (except in the case of back up tapes). This can be equated to digital rights management

(DRM), which broadly refers to the set of policies, techniques and tools used to manage the use of digital content. While there are solutions in digital rights management, this are limited to a few media and does not have the time bound or rights revoke methods.

IV. INFORMATION SECURITY MODELS

The prime goals of information security is to provide confidentiality, integrity, authentication, non-repudiation (to data) and the key factor in getting value from security is to insure that technology investments protect the right things [38]. Based on this objective, numerous models and frameworks have been proposed for securing information systems in an organisation, apart from a few studies to secure information. But, very few researches have been done for securing data “at rest, in motion and in use”. While it is impossible to totally secure any information systems, much is known today about how "systems risks" can be substantially reduced through effective management practices [6].

Lehman [29] presented a detection model using audit trails in tracking potential security violations. Deterrence theory was used for modeling sanctions by Siponen et al, and Starub [39, 40]. The use of marketing campaigns in security breach prevention has been proposed by McLean [41] along with training and education model by [42]. Straub and Welke, [6] proposed the security planning model that addresses deterrence, prevention, detection and remedies using the general deterrence theory. A five level information security management model by Solms, et al, [43] encompass deterrence, prevention, detection and discipline. Straub’s [40] model for detection and discipline of computer abuse was focused more on evaluating investment in IA security, while Trcek’s [44] layered multi-planes model for information systems security focus on e-business systems security. Ganame et al [45] proposed a distributed SOC (Security Operation Center) which is able to detect attacks occurring simultaneously on several sites in a network, and a six view perspective of system security was presented by Yadav [12]. The model of Siponen and Vance [46] uses neutralization technique and deterrence theory to develop and implement security policies. Analysis of these models reveal that they target organisational and security systems rather than data. In the realm of data centric security, the first and only attempt to provide security at data level resulted in the data centric security model proposed by IBM [47]. The purpose of the IBM model was to directly align business strategy and IT security through the common thread of data. This model focuses only on authentication, authorization and disclosure control, thus only partly complying with the security action cycle (deterrence, prevention and detection, but not remedial actions).

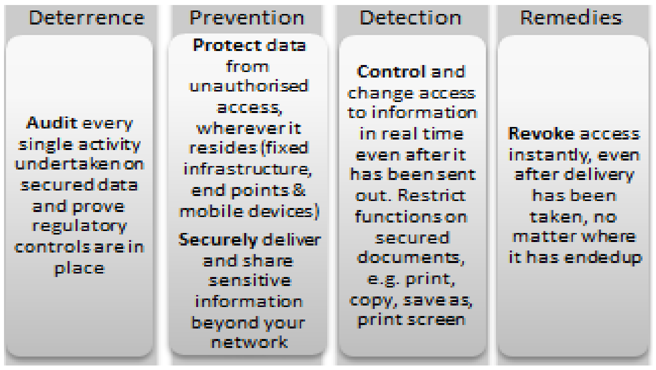
A. *The Security Action Cycle Model*

An analysis of the security models reveals that either one or more of the four components of the security action cycle (Figure 4) namely, deterrence, prevention, detection and the discipline (remedies) element is used to provide a comprehensive protection to the information systems as a whole, rather than focus on the data itself. Since hackers and unauthorized personnel target data, the four components of the security action cycle is used to built a model to focus and protect the data. The general deterrence theory have been used in IS security models [40, 46]. The security action cycle model [6] which is based on the general deterrence theory outlines four components for preventing ‘systems risk’ (information systems damage or loss) such as deterrence, prevention, detection and remedies. Two classes of countermeasures—deterrents and preventives— have been found to be effective [23]. Deterrents are passive, administrative controls that take no active role in restricting the use of system resources. Examples include computer security awareness training sessions and distributed policy statements that specify conditions for proper use of the system. Preventives, on the other hand, screen access to the system to admit authorized users only and include physical restraints such as locks on computer equipment room doors and programmed restraints such as software locks on accounts, files, transactions, and data items [40]. Detection is the process of monitoring the events in a network. Remedies have been described by Starub and Welke as punishment and recovery procedures. The application of these four elements involves the use technical as well as non-technical *information systems controls* which is referred to as audit.

B. *Role of Controls in IS Security*

Internal controls are policies, procedures, practices, and organizational structures put in place to reduce risks [48]. Appropriate controls are necessary to protect organizations from suits against negligent duty and compliance to computer misuse and data protection legislation [49]. While a “control framework is a recognized system of control categories that covers all internal controls expected in an organisation” (IIARF 2002, cited in [50], an internal control provides reasonable assurance regarding the achievement of objectives in the area of effectiveness and efficiency of operations, reliability of financial reporting and compliance with regulations [51].

The analysis derived from the three sections direct the researcher to the need for audit, protect data, transfer data across networks, control the access of data and revoke access even after delivery. These analysis viewed through the security action cycle is given in Figure 4.



Deterrence	Prevention	Detection	Remedies
Audit every single activity undertaken on secured data and prove regulatory controls are in place	**Protect data** from unauthorised access, wherever it resides (fixed infrastructure, end points & mobile devices) **Securely** deliver and share sensitive information beyond your network	**Control** and change access to information in real time even after it has been sent out. Restrict functions on secured documents, e.g. print, copy, save as, print screen	**Revoke** access instantly, even after delivery has been taken, no matter where it has ended up

Figure 4. Data centric components embedded into the security action cycle

V. DATA CENTRIC SECURITY CYCLE MODEL

Data centric security is a relatively new field and despite the fact that there exist a few security ontologies in the literature, none has yet touched on the topic of data-centric security [52]. Data-centric security starts with a hard look at what data the business must protect and its classification where the focus is on the data instead of the network [53]. Data sharing among the extended stakeholders requires complex procedures in the form of data sharing agreements, contracts, access privileges, usage and routing control infrastructure in the form of security policies [52], but despite these controls, the extent of data breaches, data loss and data pilferages could not be contained.

The data centric security cycle (DCSC) model revolves around five dimensions namely protection of data, securely deliver and share sensitive information, audit every single activity doen by users to information, control and change acceess on a continual basis, and if necessary revoke access even after the delivery of information has been taken by the user. Security of information here refer to where, when, who and how security can be used as an enabler of business, not as a restriction. This is done through storing information in encrypted format wherever needed, protecting through strong encryption, digital rights management and business-focused data leakage prevention. This ensures sharing data solely amongst authorised users, both internal and external to the organization and controlling it in real time of all user, administrator and file activity. The ISO 27001 IS security standard reiterates the cyclical nature and thus incorporates the Plan – Do – Check - Act (PDCA) cycle of Edward Deming.

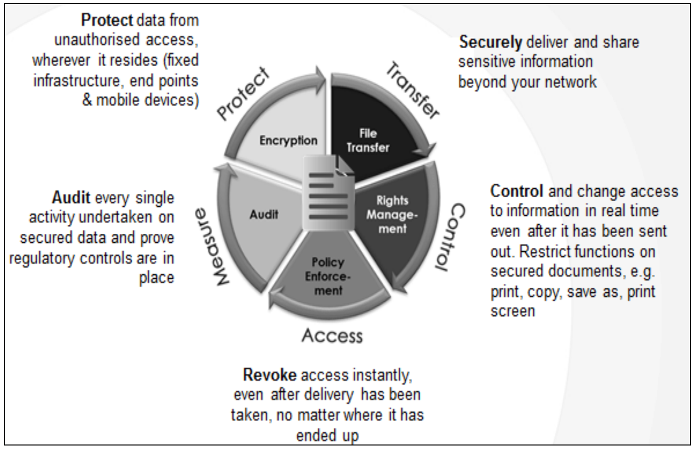

Protect data from unauthorised access, wherever it resides (fixed infrastructure, end points & mobile devices)
Securely deliver and share sensitive information beyond your network
Protect
Transfer
Encryption
File Transfer
Audit
Rights Manage-ment
Policy Enforce-ment
Measure
Control
Access
Audit every single activity undertaken on secured data and prove regulatory controls are in place
Control and change access to information in real time even after it has been sent out. Restrict functions on secured documents, e.g. print, copy, save as, print screen
Revoke access instantly, even after delivery has been taken, no matter where it has ended up

Figure 5. Data centric security cycle model

Also, taking into account the cyclical nature of the security action cycle, the model presented in Figure 4 can be refined further to the data centric security cyclical model (Figure 5). A data centric security solution provides persistent protection (persistent -at rest, in transit and in use); Strong encryption that is dynamic and granular data centric rights management where the rights can be managed at any time and in real time and it can be assigned in a granular way (administrators, users and data). The concept puts data control back in hands of data owners by separation between administrators and owners/users (Independent management) thus making it flexible for both internal and external users, for any type of file, at any storage location, for any size and type of solution, customization and integration. This data centric security concept is easy to use, simple, easy to administer, implement, maintain and is complete in terms of securing data. Since the control of data use is in the hands of the administrator, the human factor in data breaches due to accident and negligence can be substantially reduced, but not fully contained.

Next, the application of the model is discussed. When converted into a tangible solution, the model consists of three components namely the server, the web client and the agent. The server component manages all operations namely data protection, encryption, all information related to user profiles and their rights to access and use of shared files. The web client users are able to access all the functionality of the server via any Internet browser, for both Windows and Mac. The agent provides added security functions like block screen capture, print screen, video streaming, revoke access any time, create encrypted local disks, synchronize a local disk with the centralized resources through the web client, encrypt local resources making them accessible even without connection to the server and even provide time bound offline access. Once cases are selected this will be provided to them for implementation and thereafter for empirical evaluation, the researchers plan to use the updated IS success model of DeLone and McLean [54] using the variables namely information quality, systems quality and service quality from a IS security perspective. Once the users use the model, feedback (interview questions) will be framed based on these above variables.

VI CONCLUSION

Regulation and compliance are increasingly important where **c**ompliant doesn't mean secure and secure doesn't mean efficient. While organisataions need to secure networks and financial information, the concept of focussing on the data that are in use, in motion and at rest has gained relevance as is evident from the data breaches.. With this objective, this paper propose a model that focus on the real time protection/control of data. The model that was based on the security action cycle model can audit, protect, control, secure and even revoke rughts to data in online and offline mode by the data custodian. Further research can focus on protecting/controlling media and digital backup tapes using RFID technology.

REFERENCES

[1] J. Luftman and T. Ben-Zvi, "Key Issues for IT Executives 2011: Cautious Optimism in Uncertain Economic Times," *MIS Quarterly Executive*, vol. 10, pp. 203 - 212, 2011.

[2] H. A. Kruger and W. D. Kearney, "Consensus Ranking – An ICT Security Awareness Case Study," *Computers & Security*, vol. 27, pp. 254-259, 2008.

[3] S. Smith, D. Winchester, and D. Bunker, "Circuits of Power: A Study of Mandated Compliance to An Information Systems Security De Jure Standard inaA Government Organization," *MIS Quarterly Executive*, vol. 34, pp. 463-486, 2010.

[4] L. A. Gordon, M. P. Loeb, and T. Sohail, "Market Value of Voluntary Disclosures Concerning Information Security," *MIS Quarterly Executive*, vol. 34, pp. 567-594, 2010.

[5] K.-L. Thomson and R. v. Solms, "Information Security Obedience: A Definition," *Computers and Security*, vol. 24, 2005.

[6] D. Straub and R. Welke, "Coping with Systems Risk: Security Planning Models for Management Decision-Making :Working paper version," *MIS Quarterly*, vol. 22, pp. 441-469, 1998

[7] W. H. DeLone and E. R. McLean, "The DeLone and McLean Model of Information Systems Success: A Ten-Year Update," *Journal of Management Information Systems*, vol. 19, pp. 9-30, 2003.

[8] G. V. Post and K.-A. Kievit, "Accessibility vs. Security: A Look at the Demand for Computer Security," *Computers & Security*, pp. 331-344, 1991.

[9] M. Kjaerland, "A Taxonomy and Comparison of Ccomputer Security Incidents from the Commercial and Government Sectors," *Computer & Security*, vol. 25, pp. 522 – 538, 2006.

[10] B. Bulgurcu, H. Cavusoglu, and I. Benbasat, "Information Security Policy Compliance: An Empirical Study of Rationality-Based Beliefs and Information Security Awareness," *MIS Quarterly*, vol. 34, pp. 523-548, 2010.

[11] M. E. Whitman, "In Defense of the Realm: Understanding the Threats to Information Security," *International Journal of Information Management* vol. 24, pp. 43-57, 2004.

[12] S. B. Yadav, "A Six-View Perspective Framework for System Ssecurity: Issues, Risks, and Requirements," *International Journal of Information Security and Privacy*, vol. 4, pp. 61-92, 2010.

[13] R. T. Watson, G. G. Kelly, R. D. Galliers, and J. C. Brancheau, "Key Issues in Information Systems Management: An International Perspective," *Journal of Management Information Systems*, vol. 13, pp. 91-115, 1997.

[14] P. Gottschalk, R. T. Watson, and B. H. Christensen, "Global Comparisons of Key Issues in IS Management: Extending Key Issues Selection Procedure and Survey Approach," presented at Proceedings of the 33rd Hawaii International Conference on Systems Sciences, Hawaii, 2000.

[15] J. Luftman and R. Kempaiah, "Key Issues for IT Executives 2007," *MIS Quarterly Executive*, vol. 7, pp. 99-112, 2007.

[16] J. Luftman and T. Ben-Zvi, "Key Issues for IT Executives 2009: Difficult Economy's Impact on IT," *MIS Quarterly Executive*, vol. 9, pp. 49-59, 2009.

[17] J. Luftman and T. Ben-Zvi, "Key Issues for IT Executives 2010: Judicious IT Investments Continue Post Recession," *MIS Quarterly Executive*, vol. 9, pp. 263-273, 2010.

[18] CSIComputerSecurityInstitute, "CSI Computer Crime and Security Survey 2010/2011," Computer Security Institute, New York 2011.

[19] Deloitte, "Raising the Bar 2011 TMT Global Security Study – Key Findings," Deloitte Touche Tohmatsu TMT Security & Resilience, Netherlands 2011.

[20] K. Prince, "Protecting Your Organization from Insider Threat," vol. 2012, 2009.

[21] ITRC, "2011 Data Breach Statistics," Indentity Theft Resource Centre, San Diego 2012.

[22] DatalossDB, "2011 Year End Report," 2012.

[23] D. W. Straub, "Computer Abuse and Security: Update on an Empirical Pilot Study," *Security, Audit, and Control Review*, pp. 21-31, 1986.

[24] C. P. Garrison and M. Ncube, "A Longitudinal Analysis of Data Breaches," *Information Management & Computer Security*, vol. 19, pp. 261-230, 2011.

[25] Verizon, "2012 Data Breach Investigations Report," vol. 2012: Verizon RISK Team, 2012.

[26] ForresterConsulting, "The Value Of Corporate Secrets: How Compliance And Collaboration Affect Enterprise Perceptions of Risk," vol. 2011. Cambridge: Massachussets 2010.

[27] Ponnemon.Institute, "The True Cost of Compliance: Benchmark Study of Multinational Organizations," Michigan 2011.

[28] I. R. Paans and I. S. Herschberg, "Computer Security: The Long Road Ahead," *computers & Security*, vol. 6, pp. 403-416, 1987.

[29] R. L. Lehmann, "Tracking Potential Security Violations," *Security, Audit, and Control Review*, pp. 26-39, 1981.

[30] J. L. Spears and H. Barki, "User Participation in Information Systems Security Risk Management," *MIS Quarterly*, vol. 34, pp. 503-522, 2010.

[31] E. Shultz, "The Human Factor in Security," *Computers & Security* vol. 24, pp. 425-426, 2005.

[32] J. C. McIntosh and J. P. Baron, "Mobile Commerce's Impact on Today's Workforce: Issues, Impacts and Implications," *International Journal of Mobile Communications*, vol. 3, pp. 99-113, 2005.

[33] T. E. Julsrud, "Behavioral Changes at the Mobile Workplace: A Symbolic Interactionistic Approach " *Mobile Communications Computer Supported Cooperative Work*, vol. 31, pp. 93-111, 2005.

[34] E. F. Churchill and A. J. Munro, "Work/place: Mobile Technologies and Arenas of Activity," *ACM SIGGROUP Bulletin*, vol. 22, 2001.

[35] PrivacyRightsClearinghouse, "Chronology of Data Breaches," vol. 2012, 2012.

[36] J. M. Hagen, E. Albrechtsen, and J. Hovden, "Implementation and Effectiveness of Organizational Information Security Measures," *Information Management & Computer Security*, vol. 16, pp. 377 - 397, 2008.

[37] C. Sundt, "Information security and the law," *Information Security Technical Report*, vol. 11, pp. 2-9, 2006.

[38] T. Tsiakis and G. Stephanides, "The Economic Approach of Information Security," *Computers & Security*, vol. 24, pp. 105-108, 2005.

[39] M. Siponen, S. Pahnila, and A. Mahmood, "Employees' Adherence to Information Security Policies: An Empirical Study," *New Approaches for Security, Privacy and Trust in Complex Environments: IFIP International Federation for Information Processing*, vol. 232, pp. 133-144, 2007.

[40] D. W. Straub, "Effective IS Security: An Empirical Study," *Inibrmatioti Systetns Research*, vol. 1, pp. 255-276, 1990.

[41] K. McLean, "Information Security Awareness - Selling the Cause," presented at Proceedings of the IFIP TC11, Eigth International Conference on Information Security: IT Security: The Need for International Cooperation Amsterdam, 1992.

[42] M. T. Siponen, "A Conceptual Foundation for Organizational Information Security A`wareness," *Information Management & Computer Security*, vol. 8, pp. 31-41, 2000.

[43] R. v. Solms, H. v. d. Haar, S. H. v. Solms, and W. J. Caelli, "A Framework for Information Security Evaluation," *Information & Management*, vol. 26, pp. 143-153, 1994.

[44] D. Trček, "An Integral Framework for Information Ssystems Security Management," *Computers & Security*, vol. 22, pp. 337-360,, 2003.

[45] A. K. Ganame, J. Bourgeois, R. Bidou, and F. Spies, "A Global Ssecurity Architecture for Intrusion Detection on Computer Networks," *Computers & Security*, vol. 27, pp. 30-47, 2006.

[46] M. Siponen and A. Vance, "Neutralization: New Insights into the Problem of Employee Information Systems Security Policy Violations," *MIS Quarterly*, vol. 34, pp. 487-502, 2010.

[47] M. Bilger, L. O'Connor, M. Schunter, M. Swimmer, and N. Zunic, "Data-Centric Security: Enabling Business Objectives to Drive Security," IBM Corporation 2006, New York G310-0777-00, 2006.

[48] N.-y. Kim, R. J. Robles, C. Sung-Eon, L. Yang-Seon, and K. Tai-hoon, "SOX Act and IT Security Governance " presented at International Symposium on Ubiquitous Multimedia Computing, Hobart, 2008.

[49] G. Dhillon and S. Moores, "Computer Crimes: Theorizing about the Enemy Within," *Computers & Security*, vol. 20, pp. 715-723, 2001.

[50] Q. Liu and G. Ridley, "IT Control in the Australian Public Sector: A International Comparison," presented at Thirteenth European Conference on Information Systems, Regensburg, Germany, 2005.

[51] J. Pathak, "Internal Audit and E-Commerce Controls," *Internal Auditing*, vol. 18, pp. 30-34, 2003.

[52] B. Aziz, S. Crompton, and M. Wilson, "A Metadata Model for Data Centric Security " *Secure and Trust Computing, Data Management and Applications: Communications in Computer and Information Science*, vol. 186, 2011.

[53] J. Bayuk, "Data-Centric Security," *Computer Fraud & Security*, vol. March, pp. 7-11, 2009.

[54] W. H. DeLone and E. R. McLean, "Information Systems Success: The Quest for the Dependent Variable," *Information Systems Research*, vol. 3, pp. 60-95, 1992.

Securing Policy Negotiation for Socio-Pervasive Business Microinteractions

Secure, Simple and Efficient Implementation of Policy Negotiation

Mitja Vardjan
Research department
SETCCE
Ljubljana, Slovenia
mitja.vardjan@setcce.si

Miroslav Pavleski
Research department
SETCCE
Ljubljana, Slovenia
miroslav.pavleski@setcce.si

Jan Porekar
Research department
SETCCE
Ljubljana, Slovenia
jan.porekar@setcce.si

Abstract— **In this paper, we study security of policy negotiation and policy-based agreements for emerging mobile based dynamic business environments that feature many previously unknown parties sharing services to each other in an ad hoc fashion. Signed agreements among parties are basic enablers of trust in such dynamic environments. Before a micro service or a 3rd party application can be consumed by an employee, a policy like Service Level Agreement (SLA) typically has to be agreed by the service consumer and service provider. Various types of policies have to be accepted also in other socially tailored use-cases, e.g., when an employee joins a community. To enable appropriate degree of trust, consumer privacy protection, as well as authenticity and non-repudiation of the final agreement, the policy negotiation process has to be secured. The security principles introduced in the paper are applicable to any kind of policy negotiation and selection where two entities are involved: the provider and the requester who need to establish trust between them. A simple but secure policy negotiation or selection is described, followed by description of its implementation on Android operating system. The interactions between the parties are minimized in order to boost usability and limit bandwidth usage in socio-pervasive environment.**

Keywords-security; trust; policy agreement; policy negotiation.

I. INTRODUCTION

Formal policies are used to legally define relationship between the parties and to specify conditions under which something is provided by one party and consumed by the other party. A prominent example is Service Level Agreement (SLA) that defines relations between service provider and service consumer. SLA represents the means through which the parties involved in service provision and consumption agree on different aspects of the terms of service in question. The business user acting as service consumer needs to be aware of what the service will be providing and is the starting point for definition of Quality of Service (QoS) metrics that the service will need to respect. This is especially important in dynamic environments placed in business oriented socio-pervasive scenarios (see [10]). Furthermore, the end-user needs to know how the service will treat his private and confidential information. If a fee is applicable, the service provider needs to be sure that the end-user will pay for the service and needs to provide information on how he is going to charge for the service. As SLA defines rights and obligations of both parties, all in relation to the service provided, it contains some private information about the parties, especially about the consumer, his preferences and possibly also some sort of his identity. The provider's identity is usually less critical and most providers disclose their identities voluntarily in advance in order to attract potential consumers to their services.

However, the scope of policy is not limited to collaboration between businesses via service sharing and to SLAs only. Same principles apply when an employee is trying to join a social group or a community that may have internal rules that are not compatible with his company's confidentiality policy. In more general terms, the parties will be referred to as the requester and the provider.

In any case, the requester and provider exchange some of their data during each step of policy negotiation. In each step they need to either limit the data to be sent to non-sensitive data only, which may result in other party terminating the negotiation due to incomplete or wrong data, or trust the other party would not abuse the received sensitive data. Obviously, the former option would severely limit the usefulness of such negotiation. In order to trust the other party, each side should at least be able to verify the other one's identity using digital certificates. They both also have to make sure that the final policy, which is a binding agreement, has not been modified during the process and they would not formally confirm such tainted agreement.

All verifications and data transmissions increase usage of network bandwidth and other resources. This is especially critical in a mobile environment with unreliable connectivity, limited battery power and possible costs of data transfer. Therefore, negotiation process and number of network transfers should be minimal.

Finally, after the policy has been negotiated, the agreement itself should be stored in a secure manner because it may contain sensitive private data and because it is an evidence of agreed terms between the business parties that can be used if one of the parties repudiates its involvement.

II. POLICY NEGOTIATION TYPES

Traditionally, a single policy is generated by the service provider or other authority and the requester (e.g., service consumer, user) can either accept or reject the policy. Even with this classic approach where there is only one option for the policy and the requester can either accept or reject it, the provider gets some information about the requester, usually his identity or point of contact and his interest or fondness of the particular type of service.

A more advanced alternative is to include negotiation between both parties where the requester can reach an agreement more suitable to his needs, possibly by merging or combining various policies using dedicated algebra [11] which is out of scope of this paper. The negotiation can be done either by choosing a policy among two or more policies offered by the provider (Figure 1) or by negotiating individual parts of the overall policy with the provider. The latter alternative [12] is most complex and while it allows for complete customization of the final policy it is also harder for the provider to generate and maintain. Furthermore, whether the negotiation succeeds or fails, such a complex negotiation can be used to extract additional private data from the requester during the negotiation process [1]. This paper focuses on the former alternative, i.e., the simple degenerated policy negotiation or policy selection process (Figure 1).

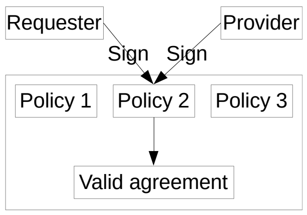


Figure 1: Policy Selection

In social and pervasive environments number of service advertisements and interactions may increase significantly. In such cases the SLA should be negotiated in a semi-automatic manner, bothering the user as little as possible and making some decisions automatically. The ability to automate policy negotiation depends on machine readability of the policy, while the security and trust aspects of policy negotiation described here are applicable for both semi-automatic and manual negotiations.

III. SECURITY PROPERTIES OF POLICY NEGOTIATION AND POLICY AGREEMENT

In order for the policy negotiation and its resulting agreement to be secure and to consequently enable trust between parties involved in service sharing or group collaboration the following security properties need to be provided:

- **Authentic evidence of non-repudiation of involvement in the negotiation process**: both parties involved in the negotiation process (the requester and provider) should not be able to deny that they were involved in the process of negotiation.
- **Confidentiality of negotiation process:** only parties involved in negotiation should be able to read and process messages passed in each step of negotiation.
- **Integrity of resulting policy or SLA agreement**: at the end of the negotiation process both parties should have identical documents that represent the agreement they have achieved with negotiation.

After the agreement is reached it will reside at each party's device for some time – at least until it is sent to the company's back end system to be archived. This may not happen instantly since mobile devices with limited connectivity or bandwidth are used in negotiation process. During this post-negotiation time an additional security property is therefore required:

- **The integrity of agreement storage:** the agreement document should not be modified before it is archived in company's back-end systems**.**

IV. SECURING NEGOTIATION

In this section, we describe how the required security properties of negotiation described in former section were satisfied. Security measures and technologies for satisfying the requirement of maintaining integrity of agreement storage are described in Section V. For each step of the negotiation process we describe the security measures, technologies, standards and implementations used to realize the required security properties. The steps or aspects of negotiation are the following:

- Provider's initial offering
- Negotiation phase
- Communication channel

A. *Provider's Initial Offer*

Regardless of the policy negotiation type, the requester has to be given some assurance that the provider's offer is real and the provider will not just collect the requests, possibly associate them with requester's data (e.g., his identity) and then not even provide the advertised service or advertised community membership.

The requester can put more trust into provider's offer if it is digitally signed by the provider using a verifiable certificate, especially if the provider is a known entity whose reputation can be affected by the requester. The trust in certificate authorities is important, but out of scope of this paper. X.509 [5] certificates are used in the described prototype.

Policy options can be prepared in advance, or generated on request. The latter alternative might not make sense because the requester does not provide any private data at the first step and the provider can tailor the policy only by vague data that can sometimes be gathered from the remote connection such as requester's IP address.

Technically, the policy options are encoded in an XML resource. The XML is in canonical form [7] and digitally signed with the provider's digital identity using XML-DSig [6]. Figure 7 shows the structure of the initial offer by example.

B. *Negotiation phase*

The whole process of policy negotiation is shown in Figure 3. At every step the parties verify the policy content has not changed from the previous step. This assures consistency of the policy during the whole negotiation process and prevents policy modifications by the other party.

At any time both parties can verify that the author of received message at each step of negotiation is the same as before. The author's identity can also be checked. A prerequisite is digital signing of policy by both parties at any step of negotiation and the usage of digital certificates.

If at any step a fraud is suspected by either party, it can safely terminate the process of negotiation. For example, if the other party (or somebody else in case of a man-in-the-middle attack) changes the terms during negotiation, or if signature verification fails, or if identity of the other party is not verifiable, then the negotiation is terminated.

The number of steps is minimized and shown in Figure 3. Figure 7 shows the initial policy options in collapsed view and the provider's signature. After getting the initial policy options, the requester locally (without interactions with provider) performs verifications (digital identity check and cryptographic signature validity check), chooses a suitable policy and signs it. The requester's signature is highlighted in Figure 8. Verification of provider identity can be skipped if Transport Layer Security (TLS) or similar protocol is used to verify it. Requester then sends the provider his choice and signature. XML-DSig [6] provides a convenient way to add requester's signature into XML-based policy. The provider locally verifies policy consistency and requester's signature. Before final policy is sent back to the requester, the provider appends another signature to the XML document. This time, the signed reference is not the policy, but the requester's signature of the policy. These two signatures are shown in Figure 9. The last signature is a confirmation and proof from the provider's side that not only the requester has agreed to the policy but also that the policy and requester's acceptance of the policy were successfully received by the provider. When the requester receives this final policy with provider's second signature, the requester archives this proof. At this point, the provider can not dispute the policy validity on grounds that the requester did not sign it or did not send it back.

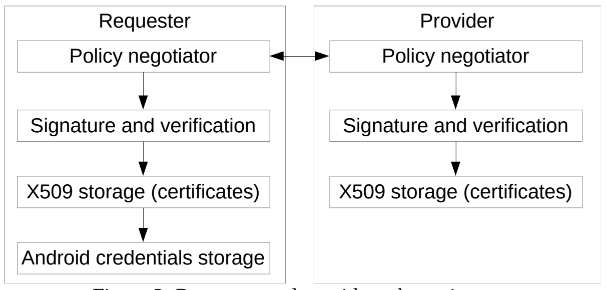

Requester
Policy negotiator
Signature and verification
X509 storage (certificates)
Android credentials storage
Provider
Policy negotiator
Signature and verification
X509 storage (certificates)

Figure 2: Requester and provider schematics

The prototype implementation is based on schematics in Figure 2. High-level negotiator components control the process and communicate between each other, while low-level components are used for signature management and storage of sensitive data.

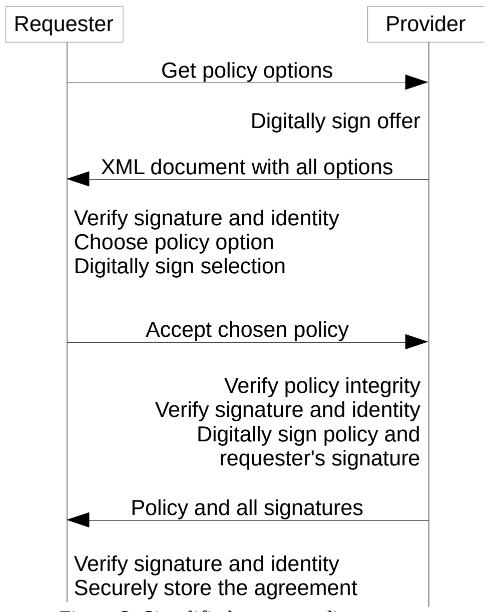

Requester
Provider
Get policy options
Digitally sign offer
XML document with all options
Verify signature and identity
Choose policy option
Digitally sign selection
Accept chosen policy
Verify policy integrity
Verify signature and identity
Digitally sign policy and
requester's signature
Policy and all signatures
Verify signature and identity
Securely store the agreement

Figure 3: Simplified sequence diagram

C. *Communication channel*

Typically, TCP/IP protocols are used and communication is secured using Transport Layer Security (TLS). Only server (in this case the provider) authentication is to be used because the identity of the requester can be revealed only if necessary at a later stage of the communication as opposite from the start of the communication mandatory by design of the TLS protocol. Technically, the client will reveal its digital identity only if and when it chooses to accept the policy.

Alternatively, instead of using wireless network, the whole negotiation could be done through Near Field Communication (NFC). Dodson and Lam [8] and Dodson et al. [9] describe the concept of micro-interactions through NFC which is applicable to policy negotiation as well. Although the close proximity required by NFC may increase requester's trust in the provider identity, the connection itself is insecure. To amend for this and encrypt the connection, a common encryption protocol like Transport Layer Security (TLS) can be used over NFC.

V. STORING THE FINAL AGREEMENT

The Android operating system facilitates the concept of Secure Credentials Storage which is used by system services to manage sensitive information such as passwords and keys. This sensitive information is stored in system protected files encrypted with a "Credential Storage Password" (Figure 4). The password is entered by the user during the first use of the negotiation and is not required for subsequent accesses to the protected data by same process. The data is still not decrypted and made available for other services in the system. Our prototype uses this system to store Policy Agreements in the phone before they are sent to company back-end storage systems.

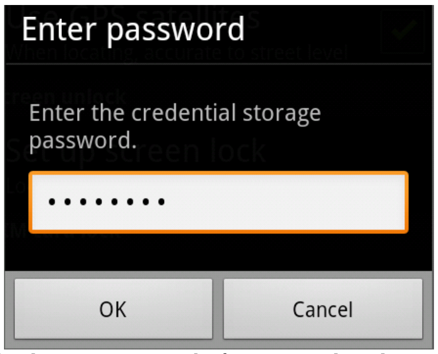


Figure 4: Android Secure Storage. The first time credential storage is used user has to provide the password to unlock the storage. The window does not reappear for further requests.

Although our prototype can access the agreement at any time, it is problematic to export the agreement to the phone's memory card, which is the only area directly and easily accessible by the user, e.g., when the phone is connected to a computer. Any installed app with permission to read from external storage can access any file on the memory card. If future improvements of our prototype implement such export of the agreements, they should appropriately protect the exported data by encrypting and signing the data. Therefore, it may be more convenient to send the agreement using TLS protocol where the data is transparently decrypted at the recipient side.

VI. Installation of Certificates

Android Secure Storage is not designed for third party applications. However, since Android is free open source project, it is possible to use the Secure Storage for a custom purpose. A special application was developed to install certificates and private keys into Android's secure storage (Figure 5). Each certificate has to be unlocked by the user before it is put into Android Secure Storage in decrypted form (Figure 6). After the initial installation, certificates are conveniently – without entering password other than that for Android Secure Storage – picked by the user when he chooses the identity to present himself with during policy negotiation.

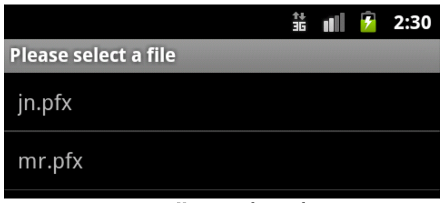


Figure 5: Installation of certificates

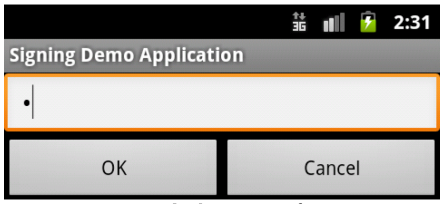


Figure 6: Unlocking a certificate

Conclusion and Future Work

A secure and trustworthy policy negotiation was presented. The negotiation itself is simplified and policy is actually selected in a single step and not seriously negotiated. Although limited in negotiation possibilities, this fast and efficient approach still addresses the needs of most providers in the real world and offers great amount of security for both requester and provider sides by means of digital certificates. Digital signatures associated with those certificates and policy integrity shall be verified in each step of negotiation, regardless of any secure network connection. The final result is an XML-based document that contains the policy, digital identities of the signing parties, evidence of the negotiation process and provides non-repudiation of the negotiation process. A working Android prototype was described as an example of secure policy negotiation in mobile pervasive environment.

In future, the prototype is planned to support also semi-automatic policy selection based on multiple policies or micro-agreements that are in place at the time of negotiation. With this upgrade the prototype is planned to be integrated into a service platform.

Acknowledgment

Authors would like to thank the SOCIETIES project [4] consortium and EC for sponsoring the project.

References

[1] J. Porekar, K. Dolinar, A. Jerman-Blažič, and T. Klobučar, Pervasive Systems: Enhancing Trust Negotiation with Privacy Support. Boston, MA: Springer US, 2007, ch. 2, pp. 23–38. [Online]. Available: http://dx.doi.org/10.1007/978-0-387-71058-7_2 [retrieved: June, 2012].

[2] K. E. Seamons, M. Winslett, and T. Yu, "Limiting the Disclosure of Access Control Policies during Automated Trust Negotiation," Proc. symposium on network and distributed systems security, NDSS, pp. 1-11, 2001.

[3] W. Chen, L. Clarke, J. Kurose, and D. Towsley, "Optimizing cost-sensitive trust-negotiation protocols", Proceedings of 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Vol. 2, pp. 1431-1442, 2005.

[4] Self Orchestrating CommunIty ambiEnT IntelligEnce Spaces (SOCIETIES), EU FP7 project, Information and Communication Technologies, Grant Agreement Number 257493.

[5] X.509 standard recommendation, http://www.itu.int/rec/T-REC-X.509/en [retrieved April, 2012].

[6] XML-DSig, XML Signature Syntax and Processing, 2nd Edition http://www.w3.org/TR/xmldsig-core/, [retrieved April, 2012].

[7] Canonical XML 1.1, W3C recommendation, http://www.w3.org/TR/xml-c14n11/, [retrieved April, 2012].

[8] B. Dodson and M. S. Lam, "Micro-Interactions with NFC-Enabled Mobile Phones",

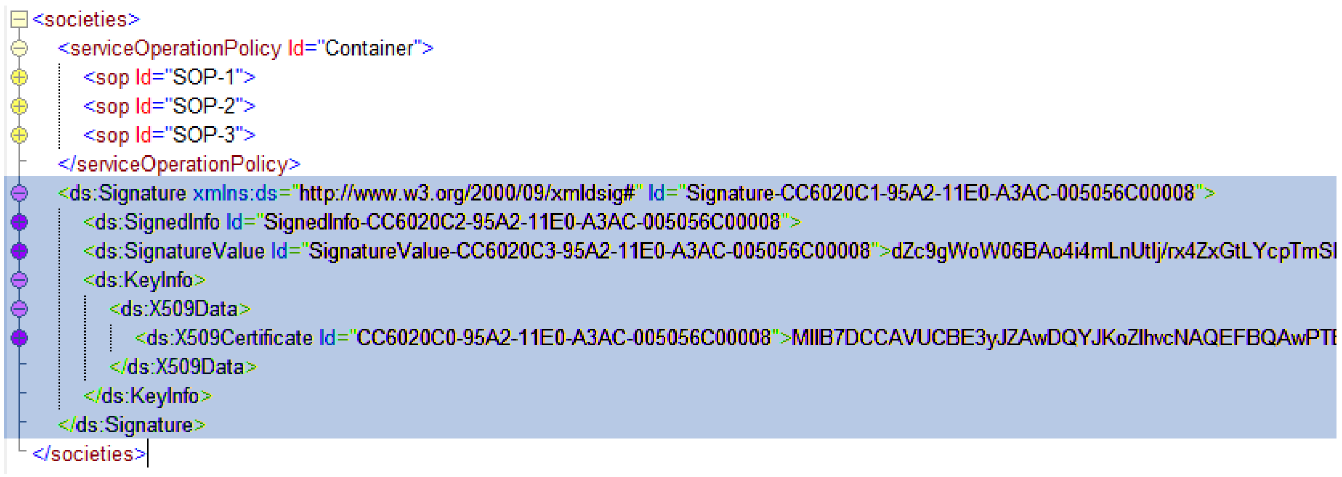
http://mobisocial.stanford.edu/papers/mobicase11.pdf, [retrieved April, 2012].

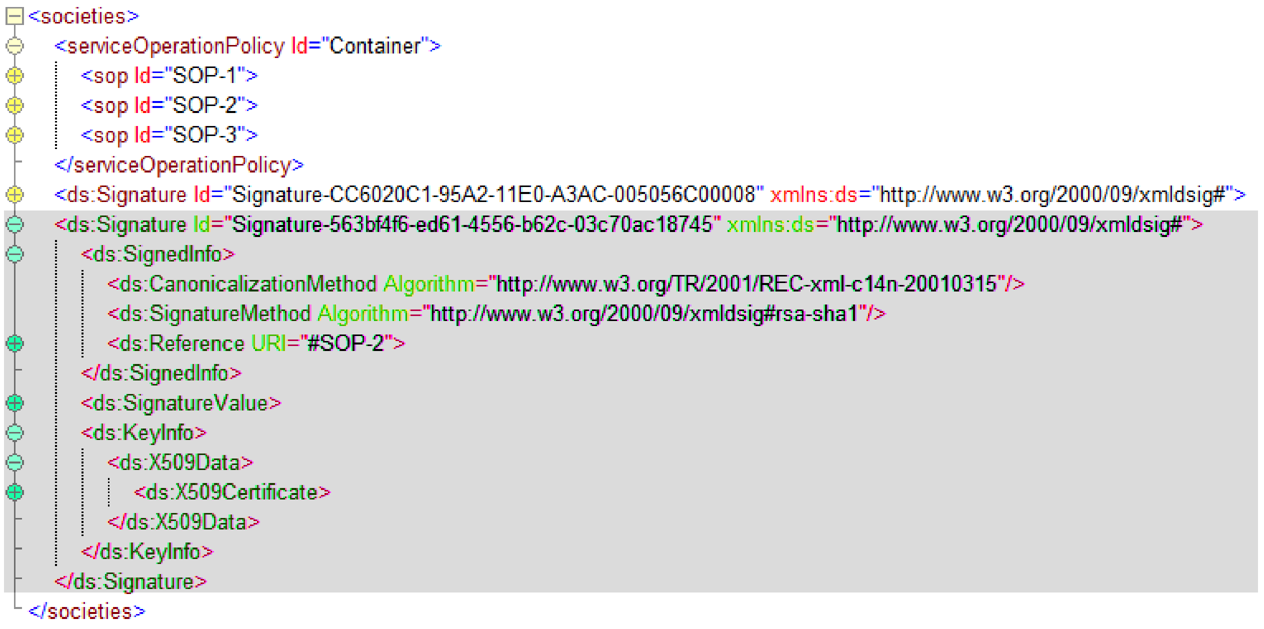
[9] B. Dodson, H. Bojinov, and M. S. Lam, "Touch and Run with Near Field Communication (NFC)", http://mobisocial.stanford.edu/papers/nfc.pdf, [retrieved April, 2012].

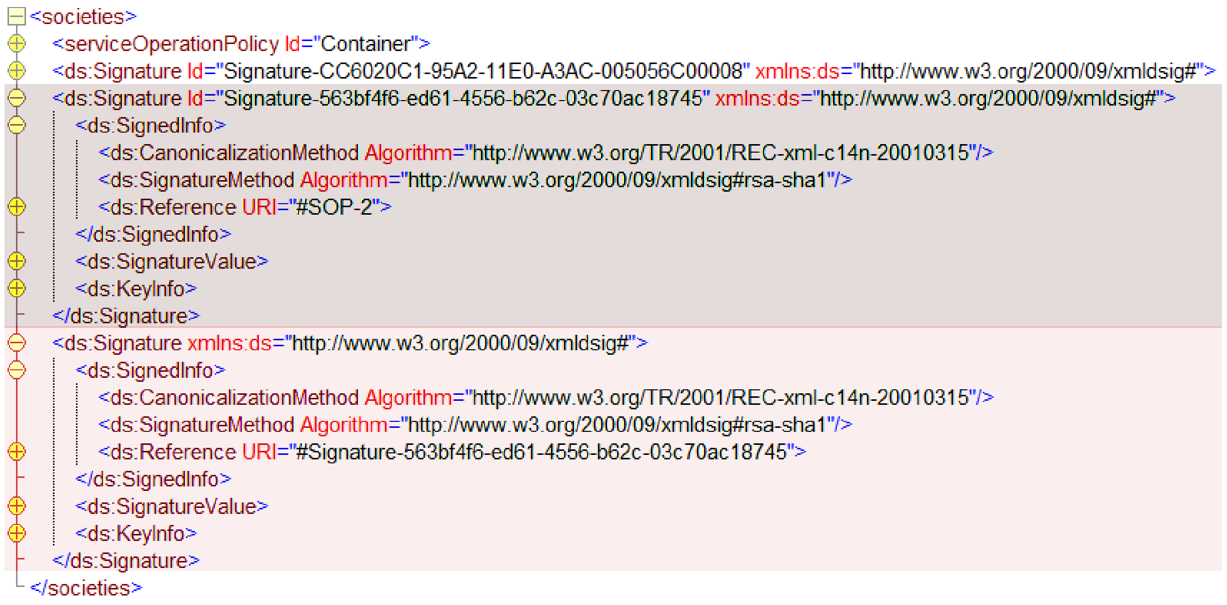
[10] S. Gallacher, E. Papadopoulou, N. K. Taylor, I. Roussaki, N. Kalatzis, N. Liampotis, F. R. Blackmun, M. H. Williams, and D. Zhang, "Personalisation in a system combining pervasiveness and social networking," in 2011 Proceedings of 20th International Conference on Computer Communications and Networks (ICCCN). IEEE, Jul. 2011, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/ICCCN.2011.6005900 [retrieved April, 2012].

[11] P. Bonatti, S. De Capitani di Vimercati, and P. Samarati, "An Algebra for Composing Access Control Policies," in ACM Transactions on Information and System Security (TISSEC), vol. 5, no. 1, pp. 1-35, February, 2002.

[12] C. Sierra, P. Faratin, and N. R. Jennings, "A service-oriented negotiation model between autonomous agents," in Lecture Notes in Computer Science, Volume 1237/1997, pp. 17-35, 1997.


```
<societies>
  <serviceOperationPolicy Id="Container">
    <sop Id="SOP-1">
    <sop Id="SOP-2">
    <sop Id="SOP-3">
  </serviceOperationPolicy>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#" Id="Signature-CC6020C1-95A2-11E0-A3AC-005056C00008">
    <ds:SignedInfo Id="SignedInfo-CC6020C2-95A2-11E0-A3AC-005056C00008">
    <ds:SignatureValue Id="SignatureValue-CC6020C3-95A2-11E0-A3AC-005056C00008">dZc9gWoW06BAo4i4mLnUtlj/rx4ZxGtLYcpTmSI
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate Id="CC6020C0-95A2-11E0-A3AC-005056C00008">MIIB7DCCAVUCBE3yJZAwDQYJKoZIhvcNAQEFBQAwPTE
      </ds:X509Data>
    </ds:KeyInfo>
  </ds:Signature>
</societies>
```
Figure 7: Signature (highlighted) of the initial policy offer. XML nodes for policy options are collapsed.


```
<societies>
  <serviceOperationPolicy Id="Container">
    <sop Id="SOP-1">
    <sop Id="SOP-2">
    <sop Id="SOP-3">
  </serviceOperationPolicy>
  <ds:Signature Id="Signature-CC6020C1-95A2-11E0-A3AC-005056C00008" xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:Signature Id="Signature-563bf4f6-ed61-4556-b62c-03c70ac18745" xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <ds:Reference URI="#SOP-2">
    </ds:SignedInfo>
    <ds:SignatureValue>
    <ds:KeyInfo>
      <ds:X509Data>
        <ds:X509Certificate>
      </ds:X509Data>
    </ds:KeyInfo>
  </ds:Signature>
</societies>
```
Figure 8: Policy during negotiation after the requester has chosen option "SOP-2" and signed it



```
<societies>
  <serviceOperationPolicy Id="Container">
  <ds:Signature Id="Signature-CC6020C1-95A2-11E0-A3AC-005056C00008" xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
  <ds:Signature Id="Signature-563bf4f6-ed61-4556-b62c-03c70ac18745" xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <ds:Reference URI="#SOP-2">
    </ds:SignedInfo>
    <ds:SignatureValue>
    <ds:KeyInfo>
  </ds:Signature>
  <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
    <ds:SignedInfo>
      <ds:CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"/>
      <ds:SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"/>
      <ds:Reference URI="#Signature-563bf4f6-ed61-4556-b62c-03c70ac18745">
    </ds:SignedInfo>
    <ds:SignatureValue>
    <ds:KeyInfo>
  </ds:Signature>
</societies>
```

Figure 9: Final policy with all three signatures. The first signature is made by provider and is shown collapsed. The second signature is made by the requester. They both reference the policy. The last signature is the provider's one and references the requester’s signature.

ComSen: A Detection System for Identifying Compromised Nodes in Wireless Sensor Networks

Yi-Tao Wang
Department of Computer Science
University of California, Los Angeles
Email: yitao@cs.ucla.edu

Rajive Bagrodia
Department of Computer Science
University of California, Los Angeles
Email: rajive@cs.ucla.edu

***Abstract*—Wireless sensor networks (WSNs) are vulnerable to adversaries as they are frequently deployed in open and unattended environments. Adversaries can extract vital information, such as security keys, from compromised nodes and use them to launch insider attacks, so detecting when a node is compromised is important to securing WSNs. In this paper, we present ComSen, an accurate and lightweight intrusion detection system for identifying compromised nodes in wireless sensor networks. Through quantitative results, we demonstrate the benefits of ComSen: (1) it is not vulnerable to slander attacks, (2) it provides detection rates of 99% and false positive ratios of less than 2% in environments with loss rates of 30%, which cannot be achieved by existing systems, (3) it runs on most WSNs because it uses common application features (sensor readings, receive power, send rate, and receive rate) and can adjust its detection behavior if the sensor application doesn't have periodic transmissions or lacks inter-node communication, and (4) it has low memory, computation, and communication overheads that allows it to scale to networks of over thousands of nodes.**

***Keywords*-wireless sensor network; ComSen; intrusion detection system;**

I. Introduction

The inexpensive and autonomous nature of sensor nodes, called motes, have allowed wireless sensor networks (WSNs) to expand to many areas, including habitat monitoring, healthcare applications, home automation, and traffic control. However, as WSNs expand to more security-critical applications like battlefield surveillance, securing them against adversaries becomes an important concern. Without security in hostile environments, it is impossible to trust the reports from WSNs.

However, motes have limited resources, in terms of computational power, memory, and battery life, and are frequently deployed in open environments, so they are vulnerable to node compromise (where an adversary gains control of a node in the network). Without detection by the network, the compromised node is considered an authorized participant in the network and can launch insider attacks, which are attacks that leverage their higher access and authority.

Thus, security-critical WSNs must deploy measures to guard against node compromise. In general, security strategies can be classified as prevention, detection, or recovery. Since motes are designed to be small and cheap, their resource constraints limit the effectiveness of prevention mechanisms in WSNs [2], [15], [19]. Adversaries can physically compromise nodes using more powerful machines, such as laptops, so prevention will only delay attackers.

Detection provides a defense against compromises that bypass any deployed preventive measures. If the compromise is detected quickly, then appropriate measures can be taken to limit the damage of compromised nodes. Although there are several approaches that focus on detecting compromised nodes, they all suffer from various shortcomings. Most approaches are targeted at specific scenarios and perform poorly otherwise. For example, most systems do not account for lossy environments, common in WSNs [1], [19], and a 20% loss rate will decrease the detection rate by more than 50% and increase the false positive ratio to over 90% [28].

In this paper, we present ComSen, an intrusion detection system for identifying compromised nodes in WSNs that satisfies all of the following criteria:

- **Accuracy:** ComSen accurately detects compromised nodes in WSNs in a timely manner. Specifically, this requirement involves (1) a high detection rate, (2) a low false positive rate, and (3) a low detection time. A high detection rate means that most compromises are reported. Few false positives means that the majority of reported compromises are valid (i.e., they correspond to actual compromises) so actions against any reported compromised node can be taken with confidence. Lastly, a low detection time limits the malicious actions that can be performed by compromised nodes before they are detected.
- **Flexibility:** ComSen is not tailored for a specific application or deployment, which would limit its applicability. It makes few as assumptions about the underlying network as possible and can be used in the majority of deployments to detect compromises.
- **Robustness:** Compromised nodes may attempt to undermine the detection system through malicious behavior, such as slander attacks, where they send false information that implicates legitimate nodes as compromised. ComSen must guard against such malicious actions. Even with perfect knowledge of ComSen, an adversary will never be able to use ComSen to control the rest of the network.
- **Scalability:** Since motes have limited resources, applications with high overheads will interfere with other applications and decrease the lifespan of the mote [1]. ComSen provides low overhead so that its impact on any other applications deployed in the network is minimized.

Our primary goal in designing and implementing ComSen is to improve the overall security of WSNs by providing a system for accurately identifying compromised nodes that has, thus far, been absent. ComSen provides a lightweight and distributed architecture that can be deployed on resource limited devices such as the nodes of a sensor network with minimal impact on other deployed applications and the lifespan of the network. It provides low overhead while maintaining its accuracy for WSNs by adapting its detection mechanism to leverage the characteristics of the underlying network. Through several experiments, we show that ComSen provides accurate detection of compromised nodes than other comparable systems and can scale up to thousands of nodes.

The rest of the paper is structured as follows: Section II goes over related work in detecting compromised nodes for WSNs. Section III discusses our system and threat models. ComSen's design is presented in Section IV, implementation in Section V, and evaluation in Section VI. Lastly, Section VII is the conclusion and future work.

II. Related Work

The majority of existing detection approaches for WSNs focus on specific attacks, such as replication [5], [21], wormhole [12], [14], and sybil [6], [13]. Although they may indirectly detect compromised nodes, adversaries can avoid detection by avoiding the target attack.

The traditional method of detecting compromised nodes is to use attestation. Attestation is the checking of all or random portions of a node's memory for changes that would exist if the node was running altered code. The advantage of this approach is that it is capable to detecting compromised nodes that are not misbehaving. Several software-based attestation techniques have been proposed for WSNs [22], [23], [24]. Unfortunately, secure software-based attestation has yet to be realized in WSNs and existing attestation techniques have been circumvented through various approaches, most of which have no hardware requirements beyond a laptop and serial cable [2].

Other approaches focus on monitoring communications for misbehavior. Misbehavior is decided using anomaly-based or rule-based approaches. Anomaly-based approaches establish a baseline behavior for neighbors and consider behavior that deviate from the baseline as anomalous. For example, Onat and Miri proposed an intrusion detection system (IDS) that monitors two features: (1) the packet arrival rate and (2) the receive power [20]. Nodes will continue to monitor these two features from neighbors and any new value that deviates a certain amount from the established baseline is considered anomalous. Malicious behavior that causes a change in either feature, such as a replay attack, would be detectable. Rule-based approaches detect misbehavior as soon as a condition, established before deployment of the network, is met. For example, COOL is an IDS that uses the relationship between incoming and outgoing messages to detect compromised nodes [30]. COOL's approach is based on the observation that the majority of outgoing messages should be forwards of incoming messages. Any node that is sending T more than it is receiving, where T is some threshold, is considered to be compromised.

Our work differs from existing approaches in that is offers improved flexibility, robustness, and scalability. ComSen provides accurate detection of compromised nodes in lossy environments and regardless of other applications deployed on the mote. It is secure against mass slander attacks, where compromised nodes collude to hinder the detection process. Furthermore, it has a low overhead that allows it to be deployed in WSNs of over a thousand nodes without affecting other deployed applications or significantly reducing the lifespan of the network.

III. System and Threat Model

We have designed ComSen based on the following common characteristics of WSNs and compromised nodes:

- The network has densely deployed sensor nodes such that sensor nodes have overlapping sensing ranges and a given event is detected by multiple nodes. Since the ranges overlap, a sensor node can monitor the activities of its neighbors.
- Sensor nodes are motes with limited computation, communication, and energy resources. For example, the Mica2 motes have a 4 MHz 8 bit Atmel microprocessor, and are equipped with an instruction memory of 128KB and a RAM of 4KB [17]. There are no mobile nodes because of mobility's high energy cost.
- There is a routing protocol used to forward messages between the base station and the nodes.
- The base station is a higher order device, such as a computer, that is in a secure location.
- Sensor nodes have unique identifiers so that the base station knows which reported compromise corresponds to which node.
- All messages have timestamps and nonces.
- Adversaries can compromise sensor nodes using physical capturing or through the radio communication channel. Once a sensor node is compromised, all information of the node, including any security keys, becomes available to the adversary.
- Although compromised nodes can perform any number of attacks to degrade the network's security and performance, we focus on compromised nodes that perform malicious acts (i.e., launch insider attacks such as falsifying data).

As we will show in Section IV, ComSen leverages several of the characteristics above to provide accurate identification of compromised nodes and low overhead.

IV. ComSen's Architecture

In designing ComSen, we had to choose whether to use a distributed, centralized, or hybrid approach. A purely distributed IDS is challenging because the resource constraints of sensor nodes prevent the use of complex algorithms that would provide more accurate results in a timely manner. For example, modular arithmetic, used in traditional security protocols like RSA, is expensive on the 8-bit processors of motes. Complex calculations could be distributed across multiple sensor nodes, but the node performing a critical calculation can be compromised, and the result falsified. In contrast, a centralized approach does not have these problems because the base station has more resources and

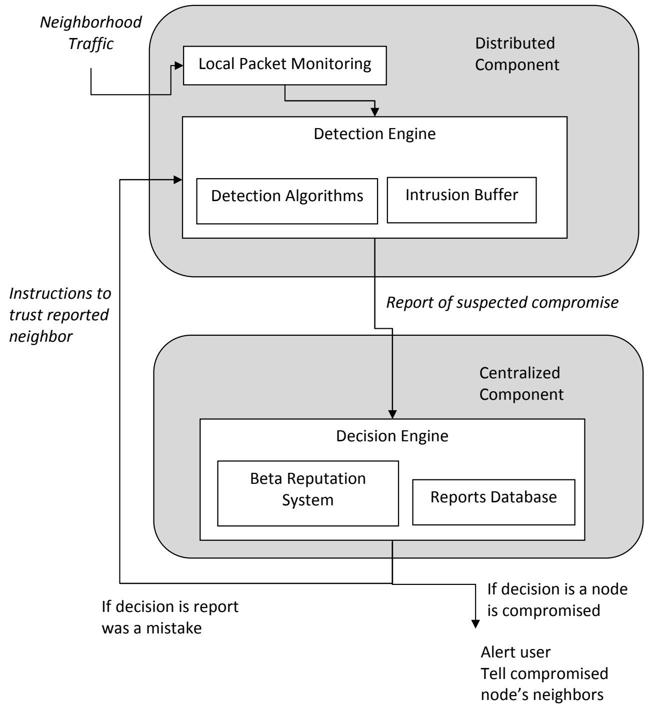


Fig. 1. Overview of ComSen

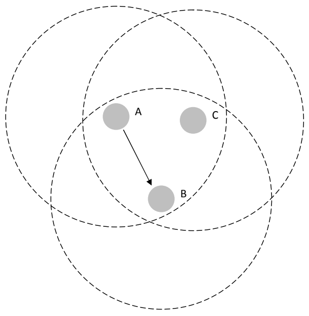


Fig. 2. Neighbor monitoring

is secure. However, the data received by the base station from compromised nodes can be false, so there must be some level of redundancy added to the network so that false data from a single node is detectable.

Thus, ComSen uses a hybrid approach, illustrated in Figure 1, that consists of two components: a distributed system running on every node in a WSN and a centralized system running on the base station.

- **Distributed Component:** A copy of this component runs on every sensor node, alongside the sensor applications, routing protocols, etc. Each copy is responsible for detecting *possible* compromises in neighboring nodes and reporting that to the base station. The detection relies on neighbor monitoring, where each node records and analyzes the behavior of its neighbors. This can be done without incurring any communication overhead because WSNs broadcast by nature. As Figure 2 illustrates, when one sensor node communicates with another, sent packets are overhead by neighboring nodes in radio range of the sender. Section IV-A discusses this component in detail.
- **Centralized Component:** The base station is a higher order machine, so ComSen uses it to perform complex analysis to decide if reports of possible compromises are accurate or mistakes. The base station aggregates data from the entire network to make a decision that would be impossible for sensor nodes given their local view and limited resources. The centralized component is discussed in Section IV-B.

There is an initial setup period after the network is deployed. During this period, nodes establish their neighbor lists, routes to the base station, etc. This period will not introduce any vulnerabilities as long as the network is secure for an amount of time after its initial deployment (i.e., adversaries are not able to immediately compromise a node in the network as soon as the network is deployed). However, if this requirement cannot be met, then the information must be pre-programmed onto each node before the network is deployed.

A. Distributed Component

Each sensor node, with a distributed component running on it, will record data from its neighbors and establish baselines based on the records. The data is from system features that reflect the behavior of nodes, so the baselines reflect the normal behavior of nodes. Every newly recorded behavior will be analyzed for *anomalies*, behavior that deviates from their established baselines; every instance of which is considered a misbehavior. If a neighbor continues to misbehave, it is considered compromised and a report about it is sent to the base station.

TO account for transient errors, such as collisions, and other non-malicious misbehavior, ComSen provides some flexibility in the amount of misbehavior that is tolerated before a neighbor is considered compromised. There is no collaboration between nodes to decide if a neighbor is misbehaving. This independent decision process means that compromised nodes cannot influence legitimate neighbors' views.

The system features that are monitored are discussed in Section IV-A1. Section IV-A2 discusses the algorithms that the distributed component uses to detect misbehavior.

1) Monitored Features: The initial step in designing any detection based security system is to select the system features that will be monitored. To provide support for most WSNs, ComSen only monitors some common features in WSNs:

- **Sensor Reading:** Nodes in WSNs are rarely deployed in scenarios where there is no correlation between the sensor readings of neighboring nodes (i.e., the readings of one node are not independent of that of its neighbors) [1]. By monitoring the sensor readings, we can detect attacks that attempt to distort gathered information.
- **Receive Power:** In static networks, the receive power should remain constant. Fluctuations may be caused by changes in the communication hardware or position of the corresponding node.
- **Send Rate:** Most applications take sensor readings and transmit them periodically. Any routed packets would also be periodic. Thus the rate of packets sent by nodes should follow a consistent pattern. Most attacks, such as selective forwarding, sybil, and replay, cause deviations in

this metric. Furthermore, sudden periods of inactivity may be caused by the process of adversaries re-programming nodes.

- **Receive Rate:** The ratio between incoming and outgoing packets should be constant because outgoing packets can only be packets being routed or packets generated by the node. A neighboring node that does not change its send rate when its receive rate changes may be compromised. It should be noted that, regardless of whether its data is encrypted, a packet's header (with the source and destination information) is often viewable by all nodes.

These choices allow ComSen to be widely applicable, because these features will be accessible in the majority of WSNs. However, these features may not be appropriate under two scenarios: (1) the packet data can only be decrypted by the base station and (2) the application rarely communicates information to the base station.

The first scenario arises when the confidentiality of information being transmitted in the WSN is important (i.e., not simple temperature readings). Since compromise cannot be detected and prevented in zero time, it's possible that some sent information may be eavesdropped by compromised nodes. Thus, packet data may be encrypted so that only the base station can decrypt it. Under such conditions, we can compensate for not monitoring the sensor readings by increasing the number of monitored neighbors to achieve comparable performance from ComSen at the cost of higher memory overhead. Section VI discusses this in further detail.

The second case is caused by applications with non-periodic communication. For example, a WSN in a demilitarization zone that only sends when movement is detected would have no communication under peaceful circumstances. Baselines cannot be established for most of the features because there's inadequate information being monitored. ComSen compensates by using an *activity mode* where nodes to send sensor readings at a rate that's a hash of its unique identification number. The communication overhead is necessary in order to prevent long periods of silence where nodes can be compromised and studied by adversaries without detection. Activity mode is only used when there is inadequate communication by the application.

2) Algorithms for Detecting Misbehavior: There are two categories of algorithms that detect misbehavior: anomaly-based and rule-based, both of which use records of monitored systems features. In anomaly-based algorithms, a baseline is established from the records and any new records that differ from the baseline, above a threshold, are considered anomalies. In contrast, rule-based check for a specific criteria (e.g., any two packets with the same header signals a replay attack). In ComSen, we decided to focus on anomaly-based algorithms in order to meet ComSen's flexibility criteria; rule-based algorithms target specific scenarios and the rules they use have to constantly updated for every new scenario.

The detection of misbehavior by the distributed component of ComSen can be divided into five algorithms: detection using (1) sensor reading, (2) receive power, (3) send rate, (4) receive rate, and (5) join messages. The first four are anomaly-based algorithms using the four features that are monitored (Section IV-A1). The last algorithm is rule-based.

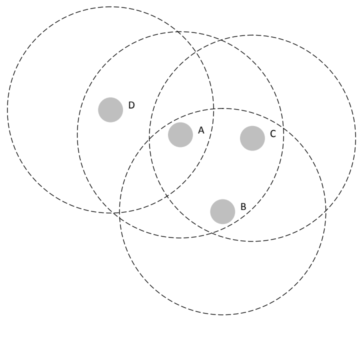


Fig. 3. Node A cannot impersonate any other node without one neighboring detecting a new neighbor

For the rule-based algorithm, if a node detects a new neighbor outside of the setup period, by hearing any messages from it. It will immediately consider the neighbor compromised.

These rules prevent compromised nodes, and even external attackers, from impersonating other nodes without being considered as new neighbors and reported. Figure 3 illustrates this property. Suppose node A was compromised and wants to impersonate another node. It cannot impersonate A or B without node D detecting a new neighbor. It cannot impersonate D without nodes B and C detecting a new neighbor. It cannot impersonate any other node except with B, C, and D detecting new neighbors. Thus, with enough neighbors monitoring each other, any attacks involving impersonation can be detected.

The four anomaly-based algorithms all follow a similar approach. Every node has two buffers for each monitored neighbor: a *packet buffer* and a *misbehavior buffer*. The buffers are shared by all four algorithms and use a sliding window approach where the last *N* packets/reports about the corresponding neighbor are stored. The data stored in the packet buffer is used to calculate that neighbor's baseline. New packets are compared against the baseline and any packets that deviate from the baseline by more than some threshold is considered anomalous. Anomalous packets indict misbehavior and cause the detecting node to generate a misbehavior report. All such reports are added to its misbehavior buffer. When the cumulative weight of reports in the misbehavior buffer passes a threshold, the node will report, to the base station, that the corresponding neighbor is compromised.

An overview of the algorithm that uses receive power is shown in Figure 4 and its corresponding equations are:

$$\begin{aligned} power_{new} - power_{max} > T, if\, power_{new} > power_{max} \\ power_{min} - power_{new} > T, if\, power_{new} < power_{min} \end{aligned} \tag{1}$$

The algorithm calculates the minimum and maximum values of packet receive power from the packet buffer. A new packet is anomalous if its receive power is T below the min or T above the max. Any detected anomalous packets are considered to be misbehaviors. Anomalous packets are added to the packet buffer so that anomalies caused by environmental changes can be accounted in future baseline calculations.

The algorithm that uses the sensor readings is almost identical to the one that uses the receive power. The only difference is

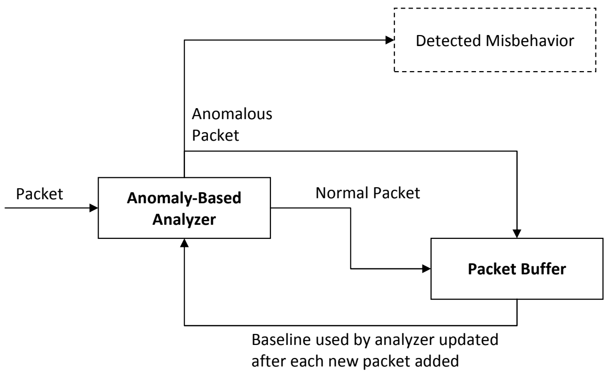


Fig. 4. Overview of the detection algorithms that use receive power and sensor readings

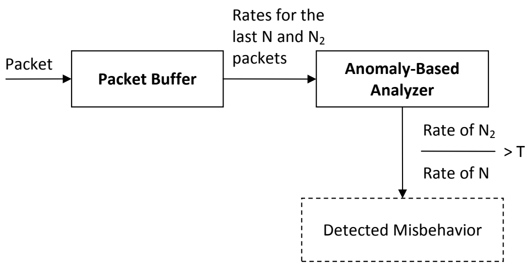

Rates for the last N and N₂ packets
Packet
Packet Buffer
Anomaly-Based Analyzer
Rate of N₂ / Rate of N > T
Detected Misbehavior

Fig. 5. Overview of the detection algorithms that use send rate and receive rate

that the difference between the node's and the neighbor's sensor readings is used instead of the receive power.

Figure 7 shows an overview of the algorithm that uses the send rate. It calculates two rates: the rate at which the last N_2 packets are sent (including the last packet), $rate_{N2}$, and the rate at which the last N packets are sent, $rate_N$ where $N > N_2$. If the ratio of these two rates is above a threshold K the corresponding neighbor is considered to be compromised.

The algorithm that uses the receive rate only differs in two ways. First, instead of counting packets sent by the neighbor, the counts are of packets received by the neighbor (i.e., sent to the neighbor). Second, the rates are replaced with send-receive ratios (i.e., $rate_{N2}$ becomes $rate_{sent_N 2}/rate_{rec_N 2}$ and $rate_N$ becomes $rate_{sent_N}$ $/rate_{rec_N}$).

All reported misbehaviors generated by the four anomaly-based algorithms for a node are stored in a shared misbehavior buffer. Each reported misbehavior is given a weight based on the time that it was detected, t_{stamp}, and the current time, $t_{current}$. When a misbehavior for a neighbor is detected, the total weight of detected misbehaviors for that neighbor is calculated using:

$$\sum_{M}(t_{current} - t_{stamp}) + 0.3\sum_{m}(t_{current} - t_{stamp}) \qquad (2)$$

where M is all detected misbehaviors of the same type and m are all detected misbehaviors of all other types. When this total passes a threshold, T_M, the corresponding neighbor is considered compromised. Thousands of simulations, described in Section VI, were used to determine the weight 0.3, optimal values of thresholds, and other parameters in these equations.

Once a node determines that a neighbor, node A, is compromised, it will send out three reports about the compromise to the base station. The reports are all identical with fields for the reporter, the node being reported, and the type of misbehavior that triggered the report. From that point on, the reporter will continue to record information from node A, but will not perform any more calculations to detect anomalies unless instructed otherwise by the base station.

B. Centralized Component

The centralized component, running on the base station, is responsible for deciding if a reported node is actually compromised, based on data from other nodes, or if the reporter made a mistake. If the reported node is compromised, the base station will alert the user and perform any recovery procedures, such as ignoring all of its messages. However, if the reported node is determined to not be compromised, the base station will tell the reporter(s) to treat it as non-compromised and continue monitoring.

For all new neighbor reports, the user is alerted. If an actual node was added to the network, the user can instruct the network that it is not malicious. For other cases, the base station will process data based on reports from other nodes. In order to make a decision about whether a reported node is compromised, ComSen uses a *beta reputation system* [4].

It has been shown that beta reputation systems are able to accurately detect misbehavior based on numerous reports and lower the high false positive rates in strict detection systems; because the system takes history into account, in order to successfully hide a compromised $nodeA$, on average 72% of $nodeA's$ neighbors have to be compromised (in contrast to the 33-50% for other approaches like majority voting) [11], [16], [26]. Beta reputation systems are an extension that uses probability density functions to combine feedback from multiple sources and determine a reputation rating, or rating of how trustworthy the subject node is. In our case, the reputation rating corresponds to a decision of compromised or not if it is past a threshold value.

In beta reputation systems, the probability, ρ, that a reported event is accurate given two parameters α and β is the beta distribution, $f(\rho|\alpha, \beta)$, which can be expressed using the gamma function Γ as:

$$f(\rho|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)+\Gamma(\beta)}\rho^{\alpha-1}(1-\rho)^{\beta-1}, \quad 0 \le \rho \le 1, \alpha > 0, \beta > 0 \qquad (3)$$

We chose the parameters, α and β, to be the weighted sum of past reports for the reported node and the number of compromised nodes within two-hops of the reported node. These allow the network topology and past reports to impact the final decision about whether a reported node is compromised. Initial baseline values are determined during the setup period.

The base station has knowledge of every node's neighbor, which may have been gathered during the setup period. A report about node A being compromised by node B has a higher probability of being correct the more of node A's neighbors report it, the longer its history of reported, and the more compromised nodes there are near it.

On the other hand, if the probability is too low, then the reporter, node B, may be compromised and launching a slander attack against node A. In which case, the base station will instruct other nodes that node B is compromised, alert the user, and launch recovery procedures.

This approach prevents adversaries from using ComSen to attack the network without being detected. If a compromised node impersonates the base station, nodes closer to the base station on the routing path will detect messages coming from the wrong direction and alert the base station. Thus, an adversary cannot impersonate the base station without being immediately detected.

ComSen is also not vulnerable to slander attacks. Suppose a compromised node, node C, wants to slander one of its neighbors, node D. As previously mentioned in Section IV-A2, impersonating other nodes will be detected by neighbors and nodes cannot influence each other. The base station knows node C's neighbors, so reporting a non-neighbor will also result in detection. The only possible slander attack is for node C to influence the base station by sending reports of compromised neighbors. However, without supporting reports from node D's neighbors, the base station will not consider node D as compromised. Moreover, the slander attack may only cause node C to be considered compromised.

V. Implementation

We implemented ComSen using TinyOS [27]. There were two key issues that may be applicable to other implementations of ComSen.

First, recall that every node has a misbehavior buffer for each of its monitored neighbors. The format and size of that buffer is implementation-dependent (i.e., depends on the needed accuracy and performance of the WSN). Nevertheless, the reports in the buffer must consist of the following fields: time of alert, type of misbehavior, and its source.

Second, there may be a high memory overhead for the buffers need to monitor all neighbors. The overhead grows as a quadratic function of the number of immediate neighbors, which is not scalable for high density networks. To address this, ComSen nodes can select a subset of neighbors to monitor. The selection can be random or come from other protocols. For example, in random pairwise key distribution protocols [7], [9], there are a number of keys generated before deployment and each node is given a random key. After deployment, there is a probability that two neighboring nodes will have compatible keys and be able to communicate. Depending on the density of the network, it's possible to control the average number of neighbors that each node can communicate with it by adjusting the total number of keys. As we will show in later sections, the performance of ComSen is maximized after a certain number of monitored neighbors, so it's unnecessary to monitor all neighbors in dense networks.

VI. ComSen's Performance

In order to analyze the performance of ComSen, we ran a series of experiments using SenSec [29], an evaluation tool which allow us to simulate and analyze various attacks on WSNs running real applications. The experiments provided quantitative analysis of the effectiveness of ComSen with different parameters.

We used the standard metrics of performance for detection systems [8], [10], [15]:

- **Detection rate:** This metric is the percentage of actual compromises that were detected by the system. However, even with a 100% detection rate, the accuracy of the system cannot be determined without considering false positives.
- **False positives:** It is possible for legitimate nodes to be reported as compromised. These reports are known as false positives. The detection rate is not inversely proportional to the false positive rate. For example, a system can have a 100% detection rate and still have a 99% false positive rate. Systems with a high percentage of false positives are inaccurate because the majority of reported compromises are false.
- **Detection time:** Detection mechanisms require time to collect and process collected data before making a decision on whether a node is compromised. Detection time is the interval during which a compromised node remains undetected.

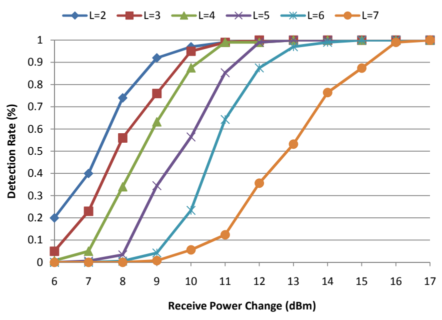
Section VI-A discusses the performance of the distributed component. The overall performance of ComSen (i.e., both its distributed and its centralized component) is discussed in Section VI-B.

A. Distributed Component's Performance

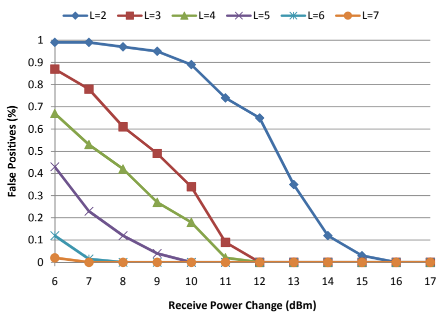
In modeling the compromise of nodes in a network, we used a gradient-based model proposed by Chen et. al. [3]. This model is based on the observation that compromises exhibit spatial locality. For example, if adversaries are compromising nodes while walking from one node to the next, then the chances of a node being compromised is higher if they're closer to a compromised node. Thus, the probability that a node is compromised forms a gradient, with higher probability for nodes closer to compromised nodes and vice versa.

Our network topology consists of 100 simulated motes randomly deployed over a 100m x 100m area. The motes have radios with transmission powers of 5 dBm and are running common sensor applications that taken sensor readings every 1s and routing them to a base station at a random edge of the network. Tree routing and CSMA are used. There is a setup period. At some random time, after the setup period, a random node in the simulated WSN is compromised every 10 simulated minutes and launches a series of attacks against the network. The attacks launched by compromised nodes are all those provided by SenSec, such as replay, sybil, wormhole, pulse-delay, selective forwarding, etc. (i.e., the performance is independent of the attack). A real TinyOS application that takes sensor readings every 0.1 s is run for each node in the simulation. As we vary the parameters in the detection algorithms, we analyze their effect on ComSen's performance. Every experiment consisted of 50 runs, each lasting 1 simulated hour.

Figures 6(a) and 6(b) show the results of the experiments where we evaluated the performance of the detection algorithm



(a) Detection rate



(b) False positives

Fig. 6. Performance of detection algorithms based on receive power change

that uses receive power at various receive power changes and packet buffer lengths (L). The threshold value is 1 dBm. For the first L packets, we use an unchanging 5dBm for the transmission power, while the receive power is simulated based on the physical topology. Then the power level of the transmission power is increased.

From the results, we can see that smaller packet buffers require smaller changes in the receive power to detect a compromise; a buffer of length 2 can achieve a 95% detection rate with the smallest change in receive power. However, the tradeoff is a higher false positive rate; a buffer of length 2 has a 95% false positive rate. These results are explained by our using past data to establish the baseline and a smaller buffer means that the algorithm is more sensitive to small changes whether they're caused by compromises or transient changes in the environment.

We found that the detection times remain constant for each buffer length, regardless of the change in receive power. The graphs were omitted for brevity, but they demonstrate that the buffer length is the deciding factor in the algorithm's sensitivity.

These results also show that, it is possible to achieve a false positive ratio of less than 10% with small packet buffers (i.e., lengths in the single digits). Thus, the memory overhead imposed by ComSen is low.

The experiments on the sensor reading detection algorithm produced similar results, which are omitted for brevity.

For the algorithm that uses send rate, we used the buffer length (L) of 6, and a sublength (L_2) of 2. Figures 7(a) and 7(b) show the performance of this algorithm as we vary the receive power, by some percentage, and the threshold value K. The results from this experiment mirror those of the previous experiment: lower values, for threshold and buffer length, lead to more responsive algorithms that offer better detection rates at the cost of higher false positive rates. For example, if K is 1.02, then a 90% detection rate is achieved with a 30% increase in the send rate, but the false positive rate is 97%. Once again, the detection times are dependent only on K and remain constant as the send rate changes.

The experiments on the receive rate detection algorithm produced similar results, which are omitted for brevity.

These experiments are meant to analyze the possible performance of the detection algorithms deployed on every node in the WSN. The actual parameters should be adjusted according to application security requirements. However, these results show that the algorithms are capable of detecting compromised nodes with detection of over 98% and false alarm of under 5%.

Of course, the performance varies greatly under different environments, because a single node with limited resources cannot achieve high accuracy under all conditions. The role of the detection algorithms is to notify the base station of *possible* compromises. The base station will compensate for the limitations of motes by aggregating reported compromises from multiple sources and decide if a report is correct.

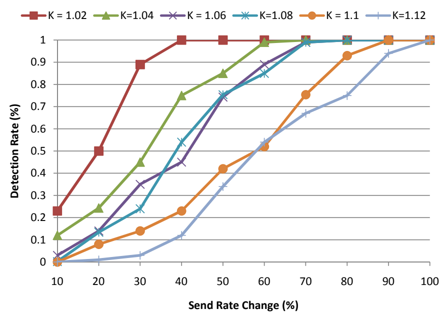
B. Overall Performance

We ran several experiments to provide quantitative analysis of the performance of ComSen. The experimental setup is identical to the one used in Section VI-A, where 100 nodes, running real TinyOS applications, are randomly deployed.

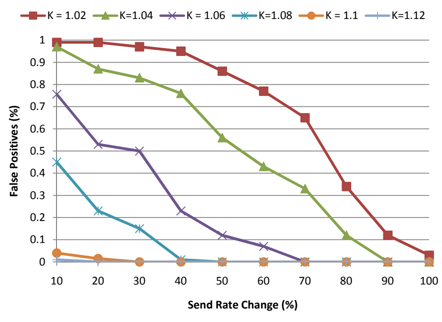
For our first experiment, we wanted to evaluate the effects of increasing the number of neighbors monitoring each other. Furthermore, most detection systems perform poorly (detection rates of less than 50% and false positive ratios of more than 90%) in lossy environments [28]. So, we varied both parameters and analyzed the results.

Figure 8 shows the results of the ComSen's performance evaluation at various loss rates and number of monitoring neighbors. Figures 8(c) and 8(b) shows that the algorithm can compensate for high loss rates with more neighbors monitoring each other. At 30% loss rate, a 99% detection rate and a 2% false positive ratio can be achieved if each node monitors an average of 9 neighbors. Higher loss rates affect the detection rate more than the false positive ratio, because lost reports make real compromises look like transient errors. However, in most cases, the compromise is detected as future malicious behavior is reported, so the detection time is higher for larger loss rates (Figure 8(c)).

We also measured the communication overhead, the number of packets sent related to the operation of ComSen, which is shown in Figure 8(d). As expected, at higher loss rates, more packets are sent due to retransmissions. However, increasing the number of monitoring neighbors does not increase the number of packets sent. In the majority of cases, the number of packets sent does not change significantly and, in some cases, the number of packets sent actually decreases 5% for every neighbor added. Closer inspection shows that when the number of monitored neighbors is increased, more neighbors are sending reports based on the same misbehavior, which increases the communication overhead. However, having more generated

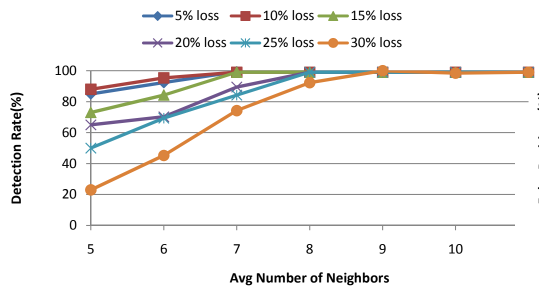


(a) Detection rate

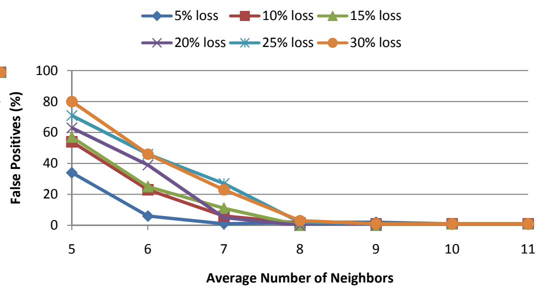


(b) False positives

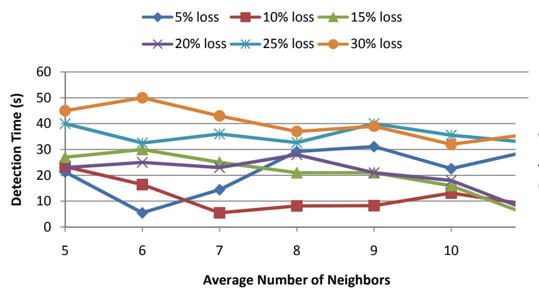
Fig. 7. Performance of detection algorithms based on send rate change



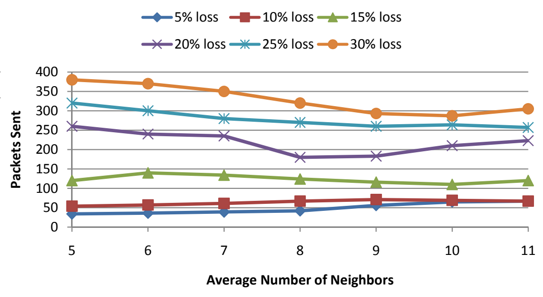
(a) Detection rate



(b) False positives



(c) Detection time



(d) Communication Overhead

Fig. 8. Performance of ComSen at various loss rates and number of monitoring neighbors

reports allows the base station to detect compromised nodes faster (i.e., the detection time is lower) when some of the reports are lost. Thus, the future misbehavior of compromised nodes won't generate reports, which lowers the communication overhead. In conditions where the loss rate is more than 15%, the net result is a decreased or unchanged communication overhead with each additional neighbor.

We measured the energy consumption of ComSen. The energy consumption of the radio accounts for the majority, of the total energy consumption, which is consistent with previous results [1], [18], [25]. Since the total energy consumption is proportional to the communication overhead, the graphs were omitted for brevity.

For our second experiment, we evaluated the effects of network density on the previous results. We increased the overall network density from 100 nodes to up to 10,000 nodes. Our results showed that density has no significant effect on ComSen's performance. We omit the results for brevity, but Figure 8 does not change significantly with network density.

We also ran experiments on the computational and memory overheads, which are omitted for brevity. As expected, they are $<1\%$ for motes, because the distributed component only performs simple computations and requires monitoring a constant number of neighbors.

These results demonstrate that ComSen can provide accurate detection of compromised nodes and is scalable to large networks. Although similar systems offer comparable performance with no loss, with 30% loss rate, their best detection rate drops to 14% and the false positive ratio becomes 99% [28]. However, ComSen can provide a 99% detection rate and a 2% false positive ratio at 30% loss rate. Moreover, its overhead is not significantly increased for denser or larger networks, scaling up to thousands of nodes

VII. Conclusion and Future Work

In WSNs, compromised nodes can undermine the integrity of data by sending false data reports, injecting false data, and disrupting transmissions. Since cryptographic solutions are not sufficient to prevent these attacks, we proposed ComSen, a system for detecting compromised nodes in WSNs.

This paper has presented the design of a novel and reliable detection system that is not vulnerable to slander attacks, where malicious nodes use the detection algorithm to launch attacks. Through a series of experiments, we showed that ComSen can provide detection rates of 99% and false positive ratios of less than 2% in environments with loss rates of 30%. ComSen can run on most WSNs because it uses common application features (sensor readings, receive power, send rate, and receive rate) and can adjust its detection behavior if the sensor application doesn't have periodic transmissions or lacks inter-node communication. It has low memory, computation, and communication overhead that allows it to scale to networks of over thousands of nodes.

Possible directions for future work include creating a response system and adding a challenge system. ComSen provides a means of identifying compromised nodes in the network but not how to respond to such a compromise. The most basic approach would be to isolate the offending node, but that may not be appropriate for all scenarios. Furthermore, ComSen does offer high accuracy in identifying compromised nodes, but once ComSen determines that a node is compromised, it could challenge the node to prove that it isn't compromised, improving the accuracy further.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102 – 114, Aug 2002.

[2] C. Castelluccia, A. Francillon, D. Perito, and C. Soriente. On the difficulty of software-based attestation of embedded devices. In *CCS '09: Proceedings of the 16th ACM conference on Computer and communications security*, pages 400–409, New York, NY, USA, 2009. ACM.

[3] X. Chen, K. Makki, K. Yen, and N. Pissinou. Node compromise modeling and its applications in sensor networks. In *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*, pages 575 –582, July 2007.

[4] B. E. Commerce, A. Jsang, and R. Ismail. The beta reputation system. In *In Proceedings of the 15th Bled Electronic Commerce Conference*, 2002.

[5] M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei. A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '07, pages 80–89, New York, NY, USA, 2007. ACM.

[6] M. Demirbas and Y. Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, WOWMOM '06, pages 564–570, Washington, DC, USA, 2006. IEEE Computer Society.

[7] R. Di Pietro, L. V. Mancini, and A. Mei. Random key-assignment for secure wireless sensor networks. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, SASN '03, pages 62–71, New York, NY, USA, 2003. ACM.

[8] D. Djenouri, L. Khelladi, and A. Badache. A survey of security issues in mobile ad hoc and sensor networks. *Communications Surveys Tutorials, IEEE*, 7(4):2 – 28, 2005.

[9] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili. A pairwise key predistribution scheme for wireless sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8:228–258, May 2005.

[10] X. Du and H.-H. Chen. Security in wireless sensor networks. *Wireless Communications, IEEE*, 15(4):60 –66, Aug 2008.

[11] S. Ganeriwal and M. B. Srivastava. Reputation-based framework for high integrity sensor networks. In *In SASN 04: Proceedings of the 2nd ACM workshop on Security of ad hoc and sensor networks*, pages 66–77. ACM Press, 2004.

[12] Y.-C. Hu, A. Perrig, and D. Johnson. Packet leashes: a defense against wormhole attacks in wireless networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1976 – 1986 vol.3, Mar 2003.

[13] N. James, E. Shi, D. Song, and A. Perrig. The sybil attack in sensor networks: analysis & defenses. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 259–268, New York, NY, USA, 2004. ACM.

[14] I. Khalil, S. Bagchi, and N. B. Shroff. Liteworp: A lightweight countermeasure for the wormhole attack in multihop wireless networks. *Dependable Systems and Networks, International Conference on*, 0:612–621, 2005.

[15] C. Krau, M. Schneider, and C. Eckert. On handling insider attacks in wireless sensor networks. *Information Security Technical Report*, 13(3):165 – 172, 2008.

[16] F. Li and J. Wu. Mobility reduces uncertainty in manets. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 1946 –1954, May 2007.

[17] Mica2. http://www.xbow.com/, June 2012.

[18] M. J. Miller and N. H. Vaidya. A mac protocol to reduce sensor network energy consumption using a wakeup radio. *IEEE Transactions on Mobile Computing*, 4:228–242, 2005.

[19] K. Okeya and T. Iwata. Side channel attacks on message authentication codes. In R. Molva, G. Tsudik, and D. Westhoff, editors, *Security and Privacy in Ad-hoc and Sensor Networks*, volume 3813 of *Lecture Notes in Computer Science*, pages 205–217. Springer Berlin / Heidelberg, 2005.

[20] I. Onat and A. Miri. An intrusion detection system for wireless sensor networks. In *Wireless And Mobile Computing, Networking And Communications, 2005. (WiMob'2005)*, volume 3, pages 253 – 259 Vol. 3, Aug 2005.

[21] B. Parno, A. Perrig, and V. Gligor. Distributed detection of node replication attacks in sensor networks. In *Security and Privacy, 2005 IEEE Symposium on*, pages 49 – 63, May 2005.

[22] A. Seshadri, M. Luk, and A. Perrig. Sake: Software attestation for key establishment in sensor networks. In *DCOSS '08: Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems*, pages 372–385, Berlin, Heidelberg, 2008. Springer-Verlag.

[23] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla. Scuba: Secure code update by attestation in sensor networks. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 85–94, New York, NY, USA, 2006. ACM.

[24] A. Seshadri, A. Perrig, L. V. Doorn, and P. Khosla. Swatt: Software-based attestation for embedded devices. In *In Proceedings of the IEEE Symposium on Security and Privacy*, 2004.

[25] V. Shnayder, M. Hempstead, B.-r. Chen, G. W. Allen, and M. Welsh. Simulating the power consumption of large-scale sensor network applications. In *Proceedings of the 2nd international conference on Embedded networked sensor systems*, SenSys '04, pages 188–200, New York, NY, USA, 2004. ACM.

[26] Y. L. Sun, Z. Han, W. Yu, and K. J. R. Liu. A trust evaluation framework in distributed networks: Vulnerability analysis and defense against attacks. In *IEEE INFOCOM*, pages 230–236, 2006.

[27] TinyOS. http://www.tinyos.net, June 2012.

[28] Y.-T. Wang. *Accurate and Scalable Security Evaluation Framework for Wireless Sensor Networks*. PhD thesis, University of California, Los Angeles, Los Angeles, CA, USA, Nov. 2011.

[29] Y.-T. Wang and R. Bagrodia. Sensec: A scalable and accurate framework for wireless sensor network security evaluation. In *SN '11: The Fourth International ICDCS Workshop on Sensor Networks*, pages 230–239, 2011.

[30] Y. Zhang, J. Yang, W. Li, L. Wang, and L. Jin. An authentication scheme for locating compromised sensor nodes in wsns. *J. Netw. Comput. Appl.*, 33:50–62, Jan 2010.

Case Study of a Black Hole Attack on 6LoWPAN-RPL

Karishma Chugh
Engineering and Information Science
Middlesex University
London, UK. NW4 4BT
karishma.chugh@ymail.com

Aboubaker Lasebae
Engineering and Information Science
Middlesex University
London, UK. NW4 4BT
a.lasebae@mdx.ac.uk

Jonathan Loo
Engineering and Information Science
Middlesex University
London, UK. NW4 4BT
j.loo@mdx.ac.uk

***Abstract*—This paper throws light on shortcomings of the Contiki Operating system and ContikiRPL routing protocol, which may lead to an easy injection of malicious activity similar to black hole attack in wireless sensor network. Contiki and ContikiRPL are components for microcontroller devices belonging to the 6LoWPAN group and Internet of Things (IoT). Directed acyclic graph Identification Object (DIO) packets are a part of routing metrics and form an integral part of ContikiRPL. Increased number of DIO messages reflect instability in the network routing topology and their decreasing frequency reflects stable network. In unstable networks, re-formation of path for data packets is initialised by RPL. In this case study it was found that malicious nodes, which continue to send self-generated data packets cause an increase in the number of DIO messages exchanged between nodes while malicious nodes, which supress self-generated data packets are able to disguise the instability of network by having no effect on the number of DIO messages or packet delay. Scenario with malicious node sending self-generated data packets showed 8% increase in total number of DIO packets exchanged amongst nodes while scenario with malicious node not generating any data packets had less number of DIO messages exchanged thus falsely presenting a stable network topology. It was also found that data packets suffer delay in presence of malicious activity in the network. Data packets generated by malicious nodes were 4.3 times higher delayed as compared to data packets from their counterparts in clear network. Data packets from non-malicious nodes also suffered considerably higher delay. Thus, increased packet delay and increase in exchange of DIO messages can be treated as preliminary indicators of malicious activity but more concrete parameters are required to identify malicious nodes. This case study may be helpful in designing an effective defense system against known attacks on wireless sensor networks.**

Keywords- *IoT; 6LowPAN; Contiki; RPL; ContikiRPL; Wireless sensor network; Black hole attack.*

I. INTRODUCTION

Internet of Things (IoT) is a network of billions of small and big communicating devices. Wireless sensor networks are a subset of IoT. Devices in wireless sensor networks are small sensor nodes, having power and memory constraints and ad-dressed using Internet Protocol version 6 (IPv6) [2]. Sensor nodes communicate with each other as per specifications provided by IEEE 802.15.4 [2]. Protocols corresponding to physical and data link layer are specified in IEEE 802.15.4. Specialised task group formed by Internet Engineering Task Force (IETF) has defined header compression and framing technique to facilitate communication between sensor nodes using IPv6 over a network of low power and low rate devices. This group is called 6LoWPAN [1][2]. Specialised operating systems and routing protocols have been designed and implemented to facilitate communication between sensor nodes as per 6LoWPAN specification. Contiki [7] operating system is one such open source operating system. Contiki provides a multi-threaded, event based multi-tasking environment [7][8]. Routing protocols are important building block for communication in any network. Routing for low Power and Lossy networks (RPL) protocol has been designed to support cost effective routing over low power and lossy networks (LLN) [9]. ContikiRPL is one of the many implementations of RPL. Black hole attack [10] in a network would imply that one or more malicious nodes would partially or fully drop data packets being routed through it causing disruptions in the normal data flow in the network. Malicious node advertises itself as the best route towards the control node (called sink node) just like other sensor nodes. Some nodes (sender nodes) select the malicious node as their parent node (next in line node in the routing topology) and start forwarding their data packets; these data packets are then dropped [3]. Securing IoT, especially sensor network is essential. This required detailed understanding of the functioning as well as shortcomings of various building blocks of the network such as operating system, device properties and routing mechanism.

This work would strengthen the knowledge of various forms of attacks, their effect on wireless sensor network, parameters to facilitate identification of attack and attacking nodes, and ultimately help introduce a strong defence system. Section II explains the simulated environment of the study and the parameters, which are observed. Section III tabulates the findings from the logs obtained as a result of the simulation. Section IV elaborates the observations and helps draw a conclusion by connecting them to the known behaviour of the system. Section V concludes the results obtained from the case study. Finally, Section VI tries to give directions towards elaborating the work for more detailed analysis of attacks on sensor networks.

II. Methodology

Certain features of Contiki and ContikiRPL are exploited to simulate and monitor malicious behaviour in this work. Contiki deals with each type of data packet differently. Each node processes data packets, which are generated by other node but routed through it in a different manner than processing self-generated data packets. In order to simulate malicious activity, modifications are made in contiki OS such that data packets from neighbouring nodes are completely dropped by the malicious node. Malicious node continues to take part in the route formation by sending consistent DIO packets. This ensures that nodes are live and continue to advertise themselves to neighbours. Malicious sensor nodes may or may not continue to send data packets generated by itself [4]. DIO messages are an integral part of RPL and play a critical role in formation of a topology. They contain metrics, which is used by nodes to form a route. Number and frequency of DIO messages decrease as the route stabilises [4][5][6]. A light weight simulator called Cooja is used to monitor the network under various scenarios. All signal messages are collected in the Log Listener plug-in of Cooja and used for analysis. Parameters under observation are packet delay, packet delivery fraction and rate of a specific control message called the DIO message. There are three types of scenarios created to record and compare the above stated parameters. First scenario called Clear Network is free from any malicious activity and consists of 1 sink node and 10 sender nodes. Sender nodes were randomly placed covering a large distance. Figure 1 shows sink node (ID: 1) and its position with respect to other sender nodes. Highlighted area denotes the radio coverage of sink node. Cooja offers different types of radio coverage; standard radio coverage with default values is used for this case study. Nodes are randomly placed covering all sides of the sink node. Some nodes fall in direct range of sink node while others fall out of it and data packets from nodes outside direct radio coverage reach the sink node via other neighbouring nodes. The second scenario has one of the sender nodes randomly selected out of the 10 sender nodes to behave in a malicious way. Node 5 from Figure 1 is replaced by a new node (ID: 12) with malicious activity. All data packets from neighbouring nodes destined to sink node were dropped by this malicious node, which continues to send data packets generated by itself towards the sink. Third scenario had malicious node, which took part in route formation but did not sent any self-generated data packets across the network. Malicious node in scenario 2 (ID: 12) was replaced by modified malicious node (ID: 13) in scenario 3. Relative location of all nodes remains same across scenarios. Malicious nodes in second and third scenario are active nodes as they exchanges DIO messages and takes part in route formation. Scenario 2 represents selective forwarding attack. Selective forwarding is a special case of black hole attack where some data packets are dropped while others are forwarded successfully. Scenario 2 forwards self-generated data packets and drops data packets from other nodes. Scenario 3 represents complete black hole attack where none of the data packets are forwarded. Simulation in each scenario is allowed to execute for 25,000 seconds during which nodes are allowed to exchange data and control information. Clear network scenario serves as a benchmark and would help understanding deviation in values of selected parameters obtained from other scenarios. Effect of malicious activity on delay of data packets reaching the sink node is analysed. Increase in packet delay can serve as an indicator of presence of attacking nodes. Also, packet delay of data packets originating from Node 12 (malicious node) in scenario 2 is compared to delay of data packets from its counterpart node (ID:5) in clear network scenario. This would help indicate effect of malicious behaviour on delay suffered by data packets from malicious node itself thus helpful in identification of malicious node in an attacked network. Scenarios 2 and 3 are compared to scenario 1 in terms of frequency of DIO messages. This helps to identify if malicious activity have an effect on exchange of DIO messages. Count and frequency of DIO messages indicates route stability. Packet delivery fraction (PDF) is another parameter, which is monitored to check if values in PDF deviate significantly and if it can be used to identify possibly attacked networks.

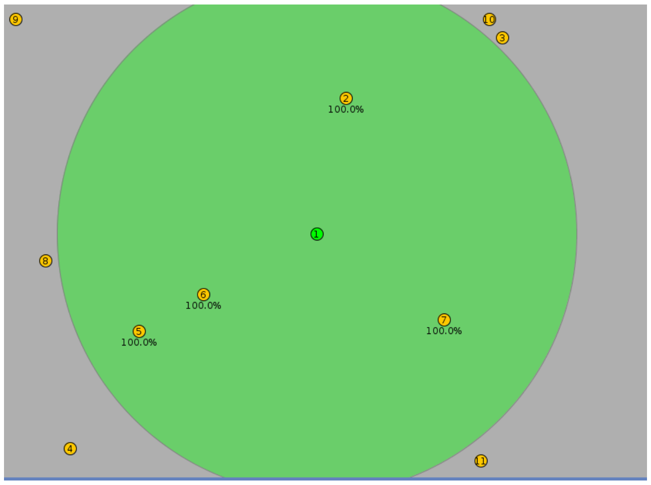


Figure 1. Placement of Sensor nodes w.r.t. Sink Node

III. Observation

Number of DIO messages sent by each sender node and time at which each of these was released, was recorded. Time of DIO message sent helps calculate and analyse the frequency of DIO messages. Number of DIO messages released by nodes across scenarios are summarised in Table 1. Increase in the number of DIO messages exchanged is a

Table I
DIO packets released in Scenarios

Node	Scn-1	Scn-2	Scn-3
2	61	68	64
3	69	72	72
4	69	77	70
5/12/13	62	78	58
6	62	63	64
7	66	65	68
8	64	88	63
9	77	87	71
10	69	65	63
11	73	70	70

Table II
Summary of Total Control messages released in a Scenario

Scenario No.	Scenario 1	Scenario 2	Scenario 3
Total DIO Messages	674	733	663
% Increase w.r.t. 1	Benchmark	+8.75%	-1.63%

direct indicator of instability in the routing topology. DIO messages released per node show that whether each node had knowledge of network instability and whether those nodes were attempting to stabilise the network by sending their own DIO packets or not. Clear network scenario would serve as a benchmark for the other two scenarios. Table 2 shows the results of this analysis. Increasing the granularity of this analysis, table 3 tabulates the number of DIO messages released by each individual node. Node 12 is the malicious node in scenario 2 and node 13 is the malicious node in scenario 3. Nodes 8 and 9 are the affected nodes in both scenarios 2 and 3. Node 4 is affected only in scenario 3. Node 5 is the counterpart of node 12 and 13 and is present in clear network scenario alone. Table 4 presents the extract of the log where node 4 in scenario 3 is trying to find a stable parent for its data packets. Figure 2 shows the neighbouring nodes of node 4 in scenario 3. This extract from the log helps explain that malicious nodes suppressing self-generated data packets have better routing metrics thus would be preferably chosen by other nodes as preferred parent. Delay of packets was monitored to analyse whether increase in packet delay could be treated as an alarm for presence of malicious activity. Delay of packets originating from malicious nodes was analysed separately and delay for data packets from non-malicious nodes was done separate. Table 5 tabulates the delay of packets from node 12 in scenario 2 and compares it to data packets from a healthy node i.e., node 5 from scenario 1. Malicious node from scenario 3 is not considered in calculation of packet delay as malicious node in scenario 3 does not generate any data packets of its own. Table 6 shows delay of data packets from all nodes in all scenarios. Entries with infinity imply that none of the data packets of that particular node reached their destination. Packet delivery fraction is the ratio of number of data packets sent from all nodes to number of data packets received successfully at sink. Table 7 tabulates the total number of data packets received at sink and those sent by nodes across scenarios.

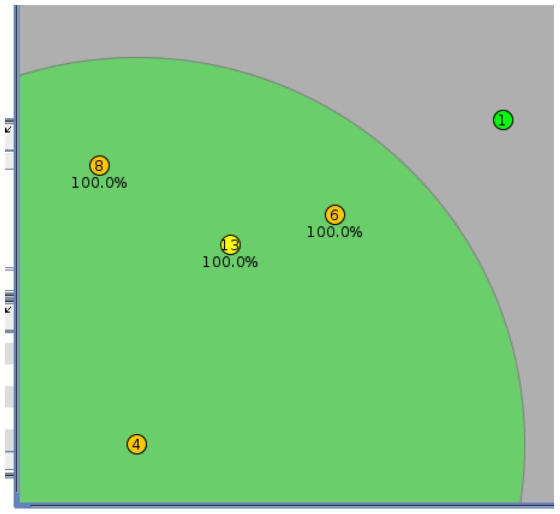


Figure 2. Node 4 and Neighbours

Table III
Summary of DIO Messages Released by Malicious and Affected Nodes alone

	5/12/13	Node 4	Node 8	Node 9
Sc 1	62	69	64	77
Sc 2	78	77	88	87
sc 3	58	70	63	71

Table IV
Extract from Log File: Preferred parent for Node 4

Node Id	Remarks	Time(ms)
ID:4	The preferred parent is Node 13	27624
ID:4	The preferred parent is Node 6	43351
ID:4	The preferred parent is Node 13	48695
ID:4	The preferred parent is Node 13	52562
ID:4	The preferred parent is Node 13	91360
ID:4	The preferred parent is Node 13	157143

Table V
Average Packet Delay for Node 5 and 12 in Scenario 1 & 2 Respectively

Scenario No.	Packets Sent	Packets Recvd	Delay(ms)
Node 5 in Scenario 1	416	416	4081.77
Node 12 in Scenario 2	416	346	17557.91

Table VI
DELAY IN MS FOR PACKETS FROM NON-MALICIOUS NODES

Node No.	Scenario 1(ms)	Scenario 2(ms)	Scenario 3(ms)
2	3481.25	4694.31	4291.95
3	4053.11	57222.43	4443.84
4	4651.91	14927.13	Infinity
6	4045.55	4734.45	3800.53
7	4013.45	4068.08	4795.66
8	3881.56	Infinity	Infinity
9	48298.87	Infinity	Infinity
10	3354.77	3505.75	3917.08
11	9898.37	49145.86	43700.46

Table VII
PACKET DELIVERY FRACTION ACROSS SCENARIOS

Scenario No.	Packets Sent	Packets Recvd	Lost %
Scenario 1	4160	4155	0.12%
Scenario 2	4160	3249	21.9%
Scenario 3	3744	2492	33.4%

IV. ANALYSIS AND RESULT

As per the working of ContikiRPL routing protocol, various control messages are exchanged between sender nodes and sink nodes to form a topology. DIO messages are formed when nodes send and receive control information from each other. Round trip time (RTT) of these control packets helps identify distance from neighbours and hop count of control packets from sink is instrumental in determining nodes own relative position to sink node. Once the topology is deemed stable, the frequency of DIO messages decrease.

Analysing the rate and frequency of DIO messages released by nodes in various scenarios, it is evident that due to malicious activity introduced by node 12 in scenario 2, all the nodes experienced unstable network topology. Scenario 2 experienced an overall increase in the number of DIO messages. Data plotted in Figure 3 shows that all nodes in scenario 2 released higher number of DIO messages when compared to clear network scenario. In scenario 3, total number of DIO messages released during the simulated time was lower than those released in scenario 1 and 2. This indicates that despite malicious node in scenario 3 dropping all data packets from its neighbours, there was no effect on the perception of other nodes, which presumed that the network was stable. This lead to the rate of DIO messages being closely comparable to rate of DIO packets in clear network scenario. This argument is further supported by Figure 4, which shows that it took much longer time for Node 5 in scenario 1 to release the same number of DIO messages than its counterpart malicious node in scenario 2; and even longer for malicious node in scenario 3. Considering the DIO messages by one of the nodes, node 8 selected malicious node as its parent in scenario 2 as well as in scenario 3. First 50 DIO messages were released in a very short span of time in scenario 2 as compared to the same first 50 DIO messages in scenario 1 and 3. Figure 5 illustrates this observation. Another important aspect noticed was the selection of preferred parent by node 4. Parent is selected on the basis of metrics extracted from DIO messages. Malicious node in scenario 3 was successful in advertising itself as preferred parent. Table 4 elaborates on the above stated fact. Table 4 shows that node 4 did consider non malicious node with ID: 6 as its preferred parent but soon changed back to malicious node 13. This implies node 13 in scenario 3 offered better routing metrics as compared to its competitive node 6 in the same scenario. Also, node 13 showed better route metrics than nodes 5 and 12 of scenario 1 and 2 respectively. Figure 2 shows nodes in direct range of node 4 available to be selected as preferred parent. This further confirms the argument that malicious node of scenario 3 was successful in performing undetected harmful activities.

In terms of delay, data packets from malicious node in scenario 2 (Node 12) suffered much higher delay than its corressponding node in scenario 1. The data is tabulated in table 5. Delay in packets of node 12 is around 4.3 times higher than its counterpart node 5 of scenario 1. Increased packet delay was not restricted to data packets from malicious nodes alone. Non malicious packets placed far or near to the malicious node also experienced higher delay in scenario 2. As shown in table 6, all nodes in scenario 2 suffered a much higher delay compared to other scenarios. Reason behind packet delay could be buffer queues. Packets might have been in the queue waiting to be processed while sink node was generating and processing DIO packets, and attempting to stabilise the network topology, longer waiting times result in an expired packet based on time to live (TTL). Since network was deemed stable in scenario 3, packet delay for most of the nodes was also comparable to clear network. In order to analyse Packet Delivery Fraction (PDF = packets received / packets sent), data packets released generated from all nodes were considered. Sink node does not release any data packets. Also, malicious node in scenario 3 had supressed forwarding of any self-generated data packets. Total numbers of packets received at the sink node are counted in each scenario. Loss of packets can be due to longer waiting times at buffer queue, or being dropped by malicious node or even might have been successful if the

Table VIII
PDF ACROSS SCENARIOS FOR NON-AFFECTED NODES

Scenario No.	Packets Sent	Packets Recvd	Lost %
Scenario 1	4160	4155	0.12%
Scenario 2	3328	3249	2.37%
Scenario 3	2496	2492	0.16%

simulation was allowed to run for longer. Table 7 has data which is self-explanatory; scenario 3 had higher number of packets dropped by malicious node, thus higher loss%. All nodes in a network are affected by the malicious activity. Since the simulation supports idealistic conditions; some nodes selected non malicious nodes as their next hop. Such sender nodes are initialised as non-affected nodes in this paper for better understanding. Most of the data packets from non-affected sender nodes are expected to reach the sink node. It was also observed that while scenario 2 had loss of packets from affected nodes as well as large number of packets from non-affected nodes were also lost; scenario 3 had only affected nodes contributing to the low PDF. This includes all nodes except nodes 8 and 9. For scenario 3, node 4 is also being excluded since it selected malicious node as its preferred parent, and thus had its packets dropped. Nodes not in direct range or path of malicious node in scenario 2 suffered packet loss compared to respective nodes of scenario 3, which had PDF% comparable to the clear scenario (scenario 1). Data is tabulated in table 8. These nodes suffered loss despite having idealistic conditions. The reason may be longer waiting times in the buffer queue of sink when sink was engaged in exchanging control messages to stabilize the routing topology, just like packet delay. Scenario 3 despite having malicious behaviour was seen stable by all other nodes, as malicious node (ID: 13) was able to support ideal routing metrics (similar to clear network). Thus malicious behaviour in scenario 2 disturbed the routing topology, while the same did not occur in scenario 3. As intended, packets from the nodes directly affected by the malicious node did not reach the sink node but other nodes had low packet loss, almost similar to loss% in scenario 1. Scenario 2 had high loss% .

Thus, it is clear that increased rate of DIO messages, increased packet delay along with falling packet delivery fraction are indicators of malicious activity but simple provisions such as supressing self-generated data packets can help a malicious node disguise its behaviour.

V. Conclusion

This work concludes that 6LoWPAN network with RPL protocol is prone to black hole attack, which can be effectively disguised and may lead to an attacked network behave very similar to a healthy network. Increased delays in most packets being delivered at sink, an overall decreased packet delivery fraction and also an increased frequency of DIO messages being exchanged amongst peers can serve as primitive indicators but do not form an exhaustive list of parameters sufficient to identify attack. These indications may be treated as signature of an attack, especially black hole attack. Packet delay and frequency of DIO messages may behave near normal if the malicious node reduces its own packet sending behaviour to NULL. Such a case would make it difficult to detect malicious behaviour. This case study considers a single malicious node in a small network. The extent of damage that could be caused in case of increase in the number of attacking nodes would be exponential. Such a network may continue to exchange control information only while most or all data packets get dropped.

VI. Future Work

The project was undertaken in a simulator with highly idealistic conditions. Elaborating the work on a real test bed would reveal better data for analysis. It has been learnt that malicious activity can be easily disguised so, analysis of additional parameters would help increase the understanding about the signature behaviour of black hole attack.

References

[1] Z. Shelby and C. Bormann, *6LoWPAN: The Wireless Embedded Internet*, UK: John Wiley & Sons Ltd. , 2010. pp. 2-18. Retrieved: Mar,2012.

[2] E. Callaway; et.al, Home networking with IEEE 802.15.4: a developing standard for low-rate wireless personal area networks, *Communications Magazine, IEEE*, Vol 40,No. 8, Aug 2002. pp. 70-77. Retrieved: Mar,2012.

[3] S. Misra; I. Woungang and S.C. Misra; *Guide to Wireless Ad Hoc Networks*, Netherlands: Springer, 2009. Retrieved: May,2012.

[4] Ward Van Heddeghem, *Cross-Layer link Estimation for Contiki Based Wireless sensor networks*; PhD Thesis, Vrije University, Brussels,2009. Retrieved: May,2012.

[5] T. Clausen and U. Herberg, Some considerations on Routing in Particular and Lossy Environments, *Proceedings of the 1st IAB interconnecting Smart Objects with the Internet Worksop*, Pgargue, Czech Republic, March 2011. Retrieved: May,2012.

[6] T. Winter (Ed.), P. Thubert (Ed.), and RPL Author Team, RPL: IPv6 Routing Protocol for Low power and Lossy Networks, *Internet Draft draft-ietf-roll-rpl-17*, work in progress. http://tools.ietf.org/html/draft-ietf-roll-rpl-17 Retrieved: June,2012.

[7] Contiki Operating system Official website : http://www.contiki-os.org Retrieved: Feb,2012.

[8] A. Dunkels, B. Gronvall, and T. Voigt,Contiki a lightweight and flexible operating system for tiny networked sensors, *In Proceedings of the First IEEE Workshop on Embedded Networked Sensors*,Tampa, Florida, USA, Nov. 2004. Retrieved: May,2012.

[9] T. Winter (Ed.) et.all, RPL: IPv6 Routing Protocol for Low power and Lossy Networks,*Internet Draft draft-ietf-roll-rpl-19*, work in progress. http://tools.ietf.org/html/draft-ietf-roll-rpl-19. Retrieved: May,2012.

[10] Al-Shurman, Mohammad, Yoo, Seong-Moo, Park, and Seungjin, Black hole attack in mobile Ad Hoc networks, *Proceedings of the 42nd annual Southeast regional conference, ACM-SE 42*,2004,isbn 1-58113-870-9,Huntsville, Alabama, pp 96–97. Retrieved: May,2012.

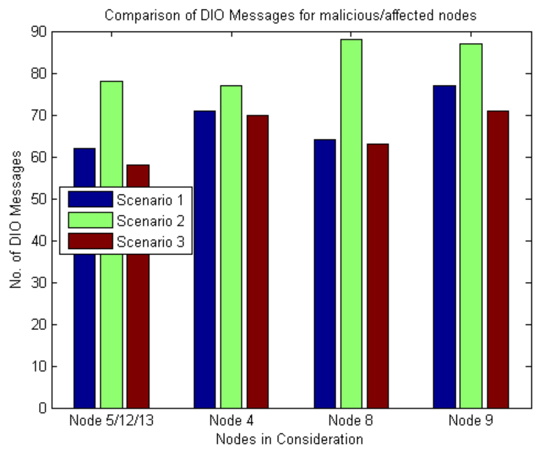


Figure 3. Comparison of DIO messages for Malicious/Affected nodes

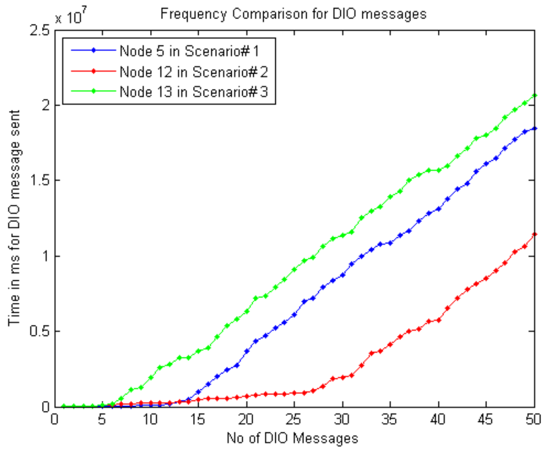


Figure 4. Frequency Comparison of DIO messages sent by Malicious nodes and Counterparts

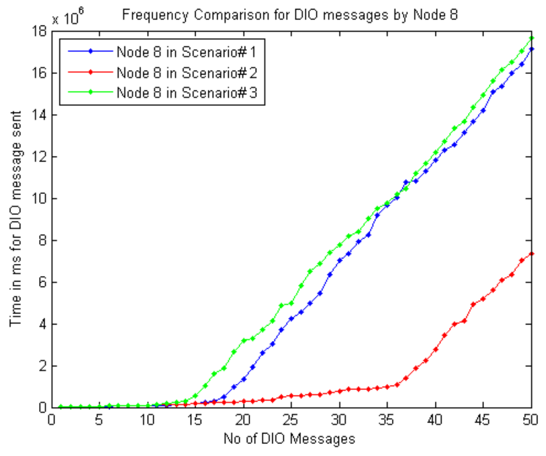


Figure 5. Frequency Comparison for DIO messages w.r.t Node 8

Construction of Optimal 4-bit S-boxes by Quasigroups of Order 4

Hristina Mihajloska
Faculty of Computer Science and Engineering
Ss Cyril and Methodius University
Skopje, Macedonia
email: hristina.mihajloska@finki.ukim.mk

Danilo Gligoroski
Department of Telematics
Norwegian University of Science and Technology
Trondheim, Norway
email: danilog@item.ntnu.no

***Abstract*—We present a new method for constructing cryptographically strong 4×4-bit S-boxes with the help of quasigroups of order 4. So far, cryptographers were constructing 4×4-bit S-boxes used in cryptographic primitives suitable for lightweight cryptography, only by exhaustive search of permutations of order 16. Our construction of 4×4-bit Quasigroup-S-boxes (Q-S-boxes) uses quasigroup string transformations. This methodology enables someone to work basically with several different strong S-boxes iteratively reusing only one hardware circuit and just changing a few parameters (called leaders in our method).**

Keywords-lightweight cryptography; quasigroups; quasigroup string transformations; S-boxes.

I. Introduction

In this paper, we focus on the Symmetric Lightweight Cryptography for cryptographic components that can be efficiently implemented into block ciphers. Although the Advanced Encryption Standard (AES) [1] block cipher is the most used cryptographic component, it was mainly designed to be efficient in software. For many constrained environments, using AES as a block cipher is either too expensive or there is no need for such a high level of security that it offers. Thus, it is not a surprise that in the last several years we see a dynamic development in the area of Lightweight Cryptography especially in the lightweight block ciphers such as PRESENT [2][3].

The main point of security in symmetric cryptography in almost all modern block ciphers is the substitution boxes also known as S-boxes. S-boxes work with a small unit of data, so they have to be distinguished with highly non-linear properties if they want to confuse the input data into the cipher.

PRESENT is an ultra-lightweight block cipher proposed by Bogdanov et al. [2]. It has been designed for extremely resource-constrained environments such as RFID tags. PRESENT is an SP-Network block cipher which consists of 31 rounds and operates on 64-bit block sizes. It supports two lengths of key, 80 or 128 bits, where 80-bit key is recommended to be used. Each of the 31 rounds is applied on three stages. The first stage is AddRoundKey, the second is SBoxLayer and the third stage is the bit permutation pLayer. The most interesting for us, is the second stage where the starring role belongs to the S-boxes.

The non-linear layer (SBoxLayer) uses a single 4-bit input and 4-bit output (4×4-bit) S-box. Also the choice of 4×4-bit S-box is a direct consequence of authors' pursuit for hardware efficiency, where implementation of such an S-box typically being much more compact and requires less resources than that of an 8×8-bit S-box. A 4-bit S-box requires less than a quarter of the hardware area (expressed in GEs - gate equivalences) of an 8-bit S-box. From cryptographic point of view, 4-bit S-boxes have to be selected very carefully because they are weaker than 8-bit S-boxes.

PRESENT S-boxes are derived as a result of an exhaustive search of all 16! bijective 4-bit S-boxes. All S-boxes found in this way that fulfilled additional criteria for optimality have been analyzed in relation to linear equivalence. So, there are only 16 different non-equivalent classes [4]. All the S-box members in these classes are optimal S-boxes with respect to linear and differential properties. Also the authors notified that these S-boxes are also optimal with respect to the algebraic degree or resistance against algebraic attacks. A slightly more general classification of all 4-bit S-boxes was given by Saarinen in [5].

Instead of an exhaustive search of all 16! bijections of 16 elements as it was done for the design of PRESENT, in this work we offer a compact, fast and elegant methodology for construction of cryptographically strong S-boxes by using quasigroups of order 4. Our goal is to give cryptographers an iterative tool for designing cryptographically strong S-boxes (in this paper, we denote them as Q-S-boxes since their construction is done by quasigroups) for future designs in the symmetric lightweight cryptography. Our methodology enables someone to work basically with several different strong S-boxes iteratively reusing only one hardware circuit and just changing a few parameters.

The structure of the paper is the following. In Section II, we give a brief mathematical description of the quasigroups and quasigroup string transformations. In Section III, we present the linear and differential characteristics of S-boxes, and conditions of one S-box to be optimal. We give the representation of quasigroups as vector valued Boolean functions in the Section IV. In Section V, we show a method for construction of cryptographic 4×4-bit Q-S-boxes, and, in Section VI, we give a conclusion and future work.

II. Preliminaries - Quasigroups and Quasigroup String Transformations

In this section, we give a brief mathematical introduction in the area of quasigroups and quasigroup string transformations. A more detailed explanation about quasigroups and their applications can be found in [6][7][8][9][10].

Let $(Q,*)$ be a finite binary groupoid, i.e., an algebra with one binary operation $*$ on the non-empty set Q and $a, b \in Q$.

Definition 1: A finite binary groupoid $(Q,*)$ is called a quasigroup if for all ordered pairs $(a,b) \in Q^2$ there exist unique solutions $x, y \in Q$ to the equations $x * a = b$ and $a * y = b$.

This implies the cancelation laws for quasigroup i.e., $x * a = x' * a \Longrightarrow x = x'$ and $a * y = a * y' \Longrightarrow y = y'$.

Any quasigroup is possible to be presented as a multiplication table known as Cayley table. Quasigroups are closely related to Latin squares. Removing the topmost row and the leftmost column of the Cayley table of a quasigroup, results in a Latin square. A Latin square is an arrangement of n symbols in a $n \times n$ matrix such that no row and no column contains any of the symbols twice.

The order of a quasigroup $(Q,*)$ is the cardinality $|Q|$ of the non-empty set Q. The set of all quasigroups of order n is denoted by $\mathbb{Q}_n$.

In what follows, we will work with finite quasigroups of order 4 only. That means that our design of S-boxes uses $|Q|^2 = 4^2$, 2-bit words of internal memory for storing the quasigroup. We will need 4 bytes (4B) of internal memory for storing the quasigroup, which is acceptable amount if we want to implement it in some lightweight designs.

Example 1: Let $Q = \{0,1,2,3\}$. A quasigroup $(Q,*)$ of order 4 has the following Cayley table:

$*$	0	1	2	3
0	0	1	3	2
1	1	0	2	3
2	2	3	0	1
3	3	2	1	0

For our method for construction of optimal S-boxes described in Section V we will use the notion of quasigroup string transformation "e-transformation" as defined in [11].

Let Q be a set of elements ($|Q| \geq 2$) and let we denote by $Q^r = \{a_0, a_1, \ldots, a_{r-1} | a_i \in Q, r \geq 2\}$ the set of all finite strings with elements of Q.

Assuming that $(Q,*)$ is a given quasigroup, for a fixed element $l \in Q$, called leader, the transformation $e_l : Q^r \rightarrow Q^r$ is as follow:

$$e_l(a_0, a_1, \ldots, a_{r-1}) = (b_0, b_1, \ldots, b_{r-1}) \Leftrightarrow$$

$$\left\{ \begin{array}{lcl} b_0 & = & l * a_0 \\ b_i & = & b_{i-1} * a_i, \quad 1 \leq i \leq r-1 \end{array} \right. \tag{1}$$

This e-transformation is called elementary quasigroup string transformation [12]. It transforms bijectively a given string with length r to other resulting string with the same length r.

Graphical representation of the transformation is shown in Figure 1.

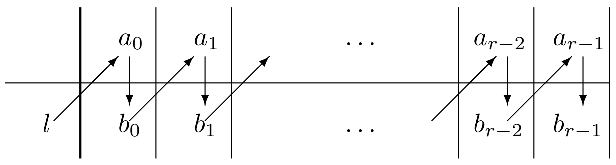


Figure 1. Graphical representation of e-transformation.

If we have some initial sequence of leaders $l_0, l_1, \ldots, l_{k-1}$, then we can make a composition of transformations by applying consecutive e-transformations.

Composite transformation obtained as a composition of e-transformations only, is defined by

$$E(l_0, l_1, \ldots, l_{k-1}) := e_{l_0}(e_{l_1} \ldots (e_{l_{k-1}}(a_0, a_1, \ldots, a_{r-1}))). \tag{2}$$

III. S-boxes and Their Properties

S-boxes have a fundamental role for the security of almost all modern block ciphers because they are usually the only non-linear part in the block ciphers. They have to be selected very carefully to make the cipher resistant against various kinds of attacks.

There is no formal definition for S-boxes. In general, they are defined as a lookup tables or vector valued Boolean functions or Boolean maps.

A Boolean function of n variables is a function $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2$, where $\mathbb{F}_2$ is a Galois field with two elements. A Boolean map (or vector valued Boolean function) is a map $f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^q$.

For two vectors $u, v \in \mathbb{F}_2^n$, where $u = (u_0, u_1, \ldots, u_{n-1})$ and $v = (v_0, v_1, \ldots, v_{n-1})$ the *canonical dot product* can be written as

$$u \cdot v = \sum_{i=0}^{n-1} u_i v_i. \tag{3}$$

Given an S-box mapping n bits to q bits, we present it as a Boolean map $S : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^q$.

Linearity of an S-box represents a measure for the resistance against linear cryptanalysis. Therefore, the smaller the linearity of an S-box is, the more secure the S-box is against linear cryptanalysis. According to all of the mathematical results given in [13] about linearity of Boolean functions and Boolean maps, we can define *linearity of an S-box*, S as:

$$Lin(S) = max\{\frac{1}{2^{2n}} S^W(u,v)^2 \mid u \in \mathbb{F}_2^n, v \in \mathbb{F}_2^q, (u,v) \neq 0\} \tag{4}$$

where u is the part of input, and v is the part of output values of the S-box.

The Walsh spectrum S^W for this S-box is calculated by:

$$S^W(u,v) = \sum_{x \in \mathbb{F}_2^n} (-1)^{u \cdot x + v \cdot S(x)} \tag{5}$$

Theoretically, it is proven that $Lin(S) \geq \frac{1}{2^n}$ [14].

One of the most important properties for S-boxes is so-called *differential potential* of an S-box. It is used in measuring the resistance of the cryptographic primitives that use that S-box against differential cryptanalysis. The differential potential of an S-box S is defined in [13] as:

$$Diff(S) = max\{\frac{1}{2^n}\Delta_S(u,v) \mid u \in \mathbb{F}_2^n, v \in \mathbb{F}_2^q, \quad \text{and} \quad (u,v) \neq 0\} \tag{6}$$

where

$$\Delta_S(u,v) = |\{x \in \mathbb{F}_2^n \mid S(x \oplus u) = S(x) \oplus v\}| \tag{7}$$

Clearly, it holds for any S-box that Diff(S) $\geq \frac{1}{2^q}$.

A. Optimal 4-bit S-boxes in PRESENT

As we mentioned in the introduction, the used S-boxes in the block cipher PRESENT have been obtained by an exhaustive search of all 16! permutations by checking some optimality criteria for their linearity and their differential potentials. Namely, all generated S-boxes were first presented as a Boolean map $S : \mathbb{F}_2^4 \rightarrow \mathbb{F}_2^4$. Then, using the above formulations about $Lin(S)$ and $Diff(S)$, the optimal set of PRESENT S-boxes was formed by S-boxes that have $Lin(S) = \frac{1}{4}$ and $Diff(S) = \frac{1}{4}$ [4].

More formally, as it is given in [4], the definition of *an optimal S-box* is the following:

Definition 2: Let S be an 4×4-bit S-box with 2^4 input values. If S fulfills the following conditions we call S an optimal S-box:

1) S is a bijection;
2) $Lin(S) = \frac{1}{4}$;
3) $Diff(S) = \frac{1}{4}$.

IV. Quasigroups as Vector Valued Boolean Functions

We will use the representation of finite quasigroups $(Q,*)$, of order n, where $n \geq 2$ and $n = 2^d$ as vector valued Boolean functions. That means that the quasigroup can be presented as a Boolean map: $f : \mathbb{F}_2^{2d} \rightarrow \mathbb{F}_2^d$. For each elements $x, y, z \in Q$ the operation $x * y = z$ is represented by

$$f(x_0, x_1, \ldots, x_{d-1}, y_0, y_1, \ldots, y_{d-1}) = (f_0(x_0, \ldots, x_{d-1}, y_0, \ldots, y_{d-1}), \ldots, f_{d-1}(x_0, \ldots, x_{d-1}, y_0, \ldots, y_{d-1})) \tag{8}$$

where $(x_0, x_1, \ldots, x_{d-1})$ and $(y_0, y_1, \ldots, y_{d-1})$ are the binary representations of x and y respectively, and $f_i : \mathbb{F}_2^{2d} \rightarrow \mathbb{F}_2$, $0 \leq i \leq d-1$ are the corresponding components of f (binary representation of z).

Every Boolean function $f : \mathbb{F}_2^m \rightarrow \mathbb{F}_2$, can be uniquely written in its Algebraic Normal Form (ANF), as a polynomial in m variables over the field $\mathbb{F}_2$ that has degree ≤ 1 in each single variable:

$$f(x_0, x_1, \ldots, x_{m-1}) = \sum_{I \subseteq \{0,\ldots,m-1\}} a_I x^I, \tag{9}$$

where the monomial x^I is the product

$$x^I = \prod_{i \in I} x_i, \tag{10}$$

and $a_I \in \{0,1\}$.

The ANF has the advantage that we can immediately read off the algebraic degree. Algebraic degree of a Boolean function is a degree of a polynomial obtained with its ANF presentation. Algebraic degree of a Boolean map is a maximal algebraic degree of its component functions. So, the ANFs of the Boolean functions f_i give us information about their algebraic degree and much better about algebraic degree or complexity of the quasigroup $(Q,*)$.

Example 2: Let us take the quasigroup given in Example 1. This quasigroup can be presented as a vector valued Boolean function $f : \mathbb{F}_2^4 \rightarrow \mathbb{F}_2^2$ by:

$$f(x_0, x_1, y_0, y_1) = (x_0 + y_0, x_1 + y_0 + x_0 * y_0 + y_1)$$

We see that the algebraic degree of this quasigroup is 2.

According to their algebraic degree quasigroups can be divided in two classes, class of linear quasigroups and class of non-linear quasigroups. The class of linear quasigroups has a maximal algebraic degree 1, and all other quasigroups (which maximal algebraic degree is bigger than 1) belong to the class of non-linear.

Considering the class of quasigroups of order 4, it can be checked that there are 144 linear and 432 non-linear quasigroups, i.e., there are three times more non-linear quasigroups of order 4 [15].

V. Construction of Optimal 4-bit Q-S-boxes

Our goal is to generate 4×4-bit cryptographically strong S-boxes by using quasigroups of order 4. Quasigroups of order 4 themselves are 4×2-bit S-boxes. It is theoretically proven that any inversion mapping for even dimension n in $GF(2^n)$ must has algebraic degree smaller than $n-1$ [16]. It should be noted that criterion for good S-box is to have highest possible algebraic degree. From this perspective, we can conclude that we would search for 4×4-bit S-boxes that have algebraic degree 3 for all output bits.

We will use quasigroup string transformations that transform a given string with length 2 to a resulting string with the same length 2, i.e., that maps 4 bits bijectively to 4 bits (Figure 2).

As it is a case with any iterative application of non-linear Boolean transformations, by consecutive application

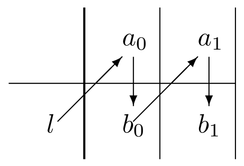


Figure 2. One e-transformation that bijectively transforms 4 bits into 4 bits by a quasigroup of order 4. Here, l, a_0, a_1, b_0 and $b_1 \in \{0, 1, 2, 3\}$.

of e-transformation we will raise the algebraic degree of the produced final bijections. More concretely, as it is shown in Figure 3, we will use one non-linear quasigroup of order 4 and at least 4 e-transformations to reach the desired degree of 3 for all the bits in final output block. Note that every row in Figure 3 starting with a leader l_i is a bijective e-transformation of the pairs of bits from previous row. In such a way, we have a composition of non-linear Boolean bijections producing the final bijection.

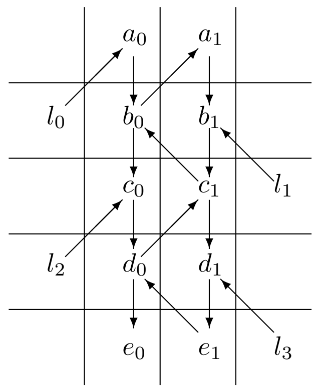


Figure 3. Four e-transformations that bijectively transforms 4 bits into 4 bits by a quasigroup of order 4.

Having one condition satisfied (the algebraic degree is maximal), we have to check further the other conditions from Section III in order to obtain optimal S-boxes, (i.e., optimal Q-S-boxes). The whole algorithm for our methodology is given in Table I.

This algorithm is for generating one Q-S-box from one chosen quasigroup of order 4 from the class of non-linear quasigroups and one combination of input leaders for the e-transformation. We already mentioned that the minimum number of rounds (iterations) for this methodology is 4. Using the described methodology we can generate Q-S-boxes in different ways depending on the number of rounds and the number of leaders that we can choose. In our investigation we choose to work with 2, 4 and 8 different leaders and 4 and 8 rounds, respectively. We found all the Q-S-boxes that fulfill the predetermined criteria to be optimal.

Experiments that are made with 2 leaders and 4 rounds as in the Algorithm 1 showed that there exist optimal Q-S-boxes. There are exactly 6,912 different Q-S-boxes (2^4 possibilities for the leaders $*$ 432 different non-linear quasigroups of order 4) that can be generated in this way, but 1,152 of them belong to the class of optimal. In Table II we give the distribution of differential and linear properties among the 6,912 examined Q-S-boxes.

Table I
CONSTRUCTION OF ONE Q-S-BOX

Algorithm 1. An iterative method for construction of Q-S-boxes	
Step 1	Take one quasigroup of order 4 from the class of non-linear;
Step 2	Input the number of rounds;
Step 3	Input the leaders. Usually, their number is the same as the number of rounds;
Step 4	Generate all possible input blocks of 4 bits in the lexicographic ordering (they are 2^4);
Step 5	Take input blocks one by one, and for each of them:
Step 5.1	Apply e-transformation with leader l on the input block;
Step 5.2	Reverse the result from above and apply e-transformation with other leader l again;
Step 5.3	Continue this routine as many times as there is a number of rounds;
Step 5.4	Save the 4-bit result from the last round;
Step 6	At the end concatenate all saved results which generate permutation of order 16 or 4×4-bit Q-S-box;
Step 7	Investigate predetermined criteria;
Step 7.1	If the Q-S-box satisfies criteria, put it in the set of optimal S-boxes;
Step 7.2	If not, go to Step 3;
Step 8	Analyze the optimal set of newly obtained Q-S-boxes;

Table II
DISTRIBUTION OF THE 6,912 Q-S-BOXES IN RELATION TO DC AND LC WHERE THE ITERATIVE METHOD WITH 2 LEADERS IS USED

LC $\rightarrow$	Lin(S)=1/4		Lin(S)=9/16		Lin(S)=1	
DC $\downarrow$	n	%	n	%	n	%
Diff(S)=1/4	1,152	16.7	0	0.00	0	0.00
Diff(S)=3/8	0	0.00	768	11.1	384	5.6
Diff(S)=1/2	0	0.00	2,304	33.3	768	11.1
Diff(S)=5/8	0	0.00	0	0.00	0	0.00
Diff(S)=3/4	0	0.00	0	0.00	0	0.00
Diff(S)=1	0	0.00	0	0.00	1,536	22.2

From the Table II can be seen that in total 1,152 Q-S-boxes have $Diff(S)$ =1/4 and $Lin(S)$ =1/4. They are 16.7% of all Q-S-boxes, that have a differential bound 1/4 and linear bound 1/4 and belong to the class of optimal S-boxes. All of these Q-S-boxes have maximal algebraic degree of all output bits 3, but some of the output bits may still have one non-linear monomial of degree 2, and therefore, this output bit depends only linearly on 2 input bits. This can be crucial when determining the number of secure rounds; final rounds can be peeled off using such properties. So, the number of Q-S-boxes that satisfy all of the output bits to have algebraic degree 3 is 128. One representative of them is given in Table III.

Table III
ONE OF THE 128 Q-S-BOXES GIVEN IN ITS HEXADECIMAL NOTATION

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	1	2	E	F	9	3	4	8	0	A	B	7	D	6	5

Table IV
DISTRIBUTION OF THE Q-S-BOXES IN RELATION TO DC AND LC WHERE THE ITERATIVE METHOD WITH 4 LEADERS IS USED

LC $\rightarrow$	Lin(S)=1/4		Lin(S)=9/16		Lin(S)=1	
DC $\downarrow$	n	%	n	%	n	%
Diff(S)=1/4	9,216	8.33	0	0.00	0	0.00
Diff(S)=3/8	3,072	2.78	12,288	11.11	6,144	5.56
Diff(S)=1/2	3,072	2.78	36,864	33.33	15,360	13.89
Diff(S)=5/8	0	0.00	0	0.00	0	0.00
Diff(S)=3/4	0	0.00	0	0.00	0	0.00
Diff(S)=1	0	0.00	0	0.00	24,576	22.22

We made experiments with 4 leaders and with the same number of rounds (one leader in each round). We produced 110,592 different Q-S-boxes (2^8 possibilities for the leaders $*$ 432 non-linear quasigroups of order 4), from which 9,216 fulfilled the criteria for optimality. In Table IV, we give the distribution of differential and linear properties among the 110,592 examined Q-S-boxes. There 8.3$\%$ of all Q-S-boxes have a differential bound 1/4 and linear bound 1/4 and belong to the class of optimal S-boxes. All of these Q-S-boxes have maximal algebraic degree of all output bits 3, but some of them still have one output bit of degree 2. The number of Q-S-boxes that satisfy, all of the output bits to have algebraic degree 3 in this case is 1,024. One representative of them is given in Table V.

Table V
ONE OF THE 1,024 Q-S-BOXES GIVEN IN ITS HEXADECIMAL NOTATION

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	D	9	F	C	B	5	7	6	3	8	E	2	0	1	4	A

Table VI
DISTRIBUTION OF THE Q-S-BOXES IN RELATION TO DC AND LC WHERE THE ITERATIVE METHOD WITH 8 LEADERS IS USED

LC $\rightarrow$	Lin(S)=1/4		Lin(S)=9/16		Lin(S)=1	
DC $\downarrow$	n	%	n	%	n	%
Diff(S)=1/4	756,480	2.67	280,320	0.99	0	0.00
Diff(S)=3/8	1,084,416	3.83	9,273,666	32.75	121,278	0.43
Diff(S)=1/2	63,744	0.23	8,394,186	29.65	2,590,518	9.15
Diff(S)=5/8	0	0.00	468,480	1.65	254,208	0.90
Diff(S)=3/4	0	0.00	224,244	0.79	87,564	0.31
Diff(S)=1	0	0.00	0	0.00	4,712,448	16.65

We made also experiments with 8 leaders and 8 rounds. In this case the number of generated Q-S-boxes significantly increased. We produced 28,311,552 different Q-S-boxes, from which 756,480 fulfilled the criteria for optimality. Distribution of these examined Q-S-boxes in relation to Differential Cryptanalysis (rows) and Linear Cryptanalysis (columns) is given in the Table VI.

The number of Q-S-boxes that satisfy, all of the output bits to have algebraic degree 3 is 331,264. One representative of them is given in Table VII.

Table VII
ONE OF THE 331,264 Q-S-BOXES GIVEN IN ITS HEXADECIMAL NOTATION

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	5	E	6	D	7	4	2	A	8	C	0	9	1	B	F	3

Apparently, by increasing the number of leaders and rounds, the number of optimal Q-S-boxes also increases. With this methodology we can generate all of the optimal S-boxes, which are already found for PRESENT. The concrete values for the leaders, the number of used leaders, and which non-linear quasigroup of order 4 to be used, in order to produce a PRESENT S-box, can be found by using some of the modern symbolic algebra systems such as Magma [17] or SAGE [18].

At the end of this section, we want to note that since we use non-linear quasigroups of order 4, the iterative procedure in Algorithm 1 has much bigger probability to produce S-boxes with optimal criteria than a random search through the set of all 16! permutations of order 16.

VI. CONCLUSION AND FUTURE WORK

In this paper, we gave a simple iterative method for producing cryptographically optimal 4×4-bit S-boxes by quasigroups of order 4, using the concept of quasigroup string transformations. We have given also the summary of our extensive experimental results. With this method and right choice of input parameters, we can generate the same optimal S-boxes like one in the lightweight block cipher PRESENT.

As a future work we emphasize the generality of our approach, and its extensibility to permutations of higher order. Thus a natural extension of our work would be to produce cryptographically strong 6×4-bit, 8×8-bit and other types of S-boxes using again iteratively just quasigroups of order 4. First of all, we should obtain how many rounds and leaders are necessary to produce Q-S-boxes with the same quality like known one, and then to see which of them belong to the class of optimal ones regarding to linear and differential characteristics of S-boxes.

REFERENCES

[1] J. Daemen and V. Rijmen. AES Proposal: Rijndael, AES Algorithm Submission, September 3, 1999. [retrieved: July, 2012]. [Online]. Available: http://csrc.nist.gov/groups/ST/toolkit/index.html

[2] A. Bogdanov, L. R. Knudsen, G. Le, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An Ultra-Lightweight Block Cipher," in *The Proceedings of CHES 2007*. Springer-Verlag, 2007, pp. 450–466.

[3] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, “A Survey of Lightweight-Cryptography Implementations,” in *IEEE Design and Test*, vol. 24, no. 6. IEEE Computer Society Press, November, 2007, pp. 522–533.

[4] G. Leander and A. Poschmann, “On the Classification of 4 Bit S-Boxes,” in *Proceedings of the 1st International Workshop on Arithmetic of Finite Fields*. Springer-Verlag, 2007, pp. 159–176.

[5] M.-J. O. Saarinen, “Cryptographic Analysis of all 4x4-bit S-boxes,” in *Proceedings of the 18th International Conference on Selected Areas in Cryptography*, ser. SAC’11. Springer-Verlag, 2012, pp. 118–133.

[6] V. D. Belousov, *Osnovi teorii kvazigrupp i lupp*. Nauka, Moskva, 1967.

[7] J. Denes and A. D. Keedwell, *Latin squares: New developments in the theory and applications*. Elsevier science publisher, 1991.

[8] ——, *Latin squares and their applications*. Academic Press Inc, December 1974.

[9] S. Markovski, D. Gligoroski, and L. Kocarev, “Unbiased Random Sequences from Quasigroup String Transformations,” in *Lecture Notes in Computer Science*, ser. FSE, vol. 3557. Springer-Verlag, 2005, pp. 163–180.

[10] J. D. H. Smith, *An Introduction to Quasigroups and Their Representations*. Chapman and Hall/CRC, 2007.

[11] S. Markovski, “Quasigroup String Processing and Applications in Cryptography,” in *The Proceedings of the 1st MII Conference*, 2003, pp. 278–290.

[12] A. Mileva, “Cryptographic Primitives with Quasigroup Transformations,” Ph.D. dissertation, University Ss. Cyril and Methodius, Skopje, Macedonia, 2010.

[13] K. Pommering, *Fourier Analysis of Boolean Maps*. A Tutorial, Mainz, 2005.

[14] F. Chabaud and S. Vaudenay, “Links Between Differential and Linear Cryptanalysis,” in *Advances in Cryptology - EUROCRYPT’94, Workshop on the Theory and Application of Cryptographic Techniques*, vol. 950. Springer-Verlag, 1995, pp. 356–365.

[15] D. Gligoroski, V. Dimitrova, and S. Markovski, “Quasigroups as Boolean Functions, Their Equation Systems and Gröbner Bases,” in *Grobner Bases, Coding, and Cryptography*. Springer-Verlag, 2009, pp. 415–420.

[16] K. Nyberg, “Differentially Uniform Mappings for Cryptography,” in *Advances in Cryptology - EUROCRYPT’93, Workshop on the Theory and Application of Cryptographic Techniques*, vol. 765. Springer-Verlag, 1994, pp. 55–64.

[17] W. Bosma, J. Cannon, and C. Playoust, “The Magma algebra system. I. The user language,” in *Journal of Symbolic Computation*, vol. 24, 1997, pp. 235–265.

[18] W. Stein, *Sage Mathematics Software (Version 5.0.1)*, The Sage Development Team, [retrieved: July, 2012][Online]. Available: http://www.sagemath.org.

Analyzing the Impact of Security Protocols on Wireless LAN with Multimedia Applications

Thaier Hayajneh, Samer Khasawneh, Bassam Jamil, Awni Itradat
Department of Computer Engineering
Hashemite University
Zarqa, Jordan
{Thaier, Samerkh, Bassam, Itradat}@hu.edu.jo

Abstract—The availability and reasonable cost of broadband Internet made it an attractive and favorable option to billions of users worldwide. Being a fast service also encourages its users to use multimedia applications. The performance of such applications in wireless LAN may be highly affected by security protocols. This paper examines the effect of different security protocols on the performance of wireless LAN with multimedia applications. Experiments were performed on a wireless test-bed and the results were analyzed for throughput, delay and jitter for four security settings: disabled security, WEP, WPA1, and WAP2. The experiments were performed under two different scenarios and using multimedia traffic streams. The results revealed a significant degradation in performance when security protocols were enabled in wireless LAN. Specifically, delay and jitter, were significantly increased, both of which are key metrics for multimedia applications. The increase is clearer when a larger number of hosts exist in the network. We finally propose an outline for a solution to obtain strong security in wireless LAN without significant performance degradation. The solution proposes that the security processing at the hosts be conducted by the powerful host processor rather than by the radio card processor. As for the wireless access point, adding ASIC or FPGA processor is suggested for performing the heavy security processing.

Keywords-WLAN; WEP; WPA1; WPA2; delay; jitter; multimedia traffic.

I. INTRODUCTION

In the last decade, wireless local area networks (WLAN) technology has become more convenient and thus has spread extensively worldwide. The security of this technology, however, is a constant concern to all its users, especially those who use it for online banking, social networking, and monetary transfer. In that regard, determining the relationship between the strength of the used security protocol and the performance of WLAN is of utmost importance. This relationship becomes even more important in applications that require high QoS to operate properly such as video conferencing and live video streaming.

Researchers have extensively investigated the impact of security protocols on the performance of WLAN. The majority focused on the network's throughput while less attention was given to delay. The results were conflicting on the impact of security on the performance of WLAN, with several [1, 2, 3, 6, 7] discussing its tangible negative impact and few concluding its negligible impact on performance of WLAN [5].

The main security protocols in WLAN are wired equivalent privacy (WEP) [8], WiFi protected access (WPA1) [10], and WiFi protected access II (WPA2) [10]. WEP is the simplest and uses computationally light cipher. However, it has been shown to be insecure and should no longer be used. WPA1 is stronger than WEP; but, has few security vulnerabilities and was replaced by WPA2 [11]. WPA2 is known to be secure since it relies on strong cipher as AES. Hence, applying WPA2 is expected to be heavy and requires considerable processing leading to increased delay. Further details will be discussed on each protocol in Section 3.

In this paper, we will examine the impact of security protocols on the performance of WLAN through conducting experiments over a test-bed. The performance of the network was examined under four conditions: disabled security, WEP, WPA1, and WPA2. Given the contemporary trend of using multimedia applications among current Internet users, a special attention was given in this paper to the impact of security protocols on the performance of WLAN in such applications. Since the multimedia applications are most sensitive to delay and jitter, these two performance metrics were the focus in this paper.

Moreover, we have proposed a new solution that will allow us to use a strong security protocol in WLAN while significantly minimizing the degradation in WLAN performance. The solution proposes that the security processing be conducted by the powerful host processors instead of the radio card processors. As for the wireless access point, adding ASIC or FPGA processor is suggested for performing heavy security processing. The need for such a proposition arose from the fact that disabled security WLAN by far outperforms other security settings in all performance aspects. Our work is different from pervious studies in considering different security protocols including WPA2 and different multimedia traffic (video streaming traffic); focusing on delay and jitter; and finally proposing a novel solution to achieve strong security without performance degradation.

The consequent parts of this paper are organized as follows: Section 2 overviews previous related work. Section 3 provides a brief description of the security protocols and their pitfalls in WLAN. Methodology and the hardware/software used in the experiments are discussed in Section 4, and results are described in Section 5. Section 6 illustrates the proposed solution, and the derived conclusion and future work are summarized in Section 7.

II. RELATED WORK

The impact of security protocols on WLAN performance has been studied in the literature from different perspectives. The majority of the researchers considered the impact on throughput and only few considered delay and jitter. Most of the networks were tested with file transfer traffic.

Barka and Boulmalf [1] studied the impact of security protocols on the throughput of WLAN. They considered both TCP and UDP traffic under two scenarios. Only the impact of WEP and WPA1 was considered but not WPA2. The study concluded that adding security will cause a decrease in the average network throughput and increase in the percentage of dropped packets for both TCP and UDP traffic. The authors assume fixed traffic intensity with fixed packet delay. Moreover, Barka and Boulmalf found that WPA1 will have the largest impact on the network performance due to the bigger key sizes and longer processing time.

Kolahi et al. [2] evaluated the impact of different security protocols on network throughput and round trip time (RTT) for both TCP and UDP traffic. They also considered different operating systems (Windows server 2003, XP and Vista). Similar to Barka and Boulmalf [1], the authors did not consider WAP2. The results showed that using security protocols reduces the throughput and increases RTT. The increase in RTT is more noticeable when larger encryption keys are used.

Several experiments have been carried out to explore the impact of security protocols on the performance of voice and data traffic in WLAN [3]. The performance metrics that were tested are: throughput, delay, and jitter. However, delay and jitter were only tested for WEP protocol with 64-bit key size; such a protocol is not secure and could be broken in few seconds [4]. Boulmalf et al. [3] showed that no noticeable throughput degradation is incurred through adding security protocols. However, considerable increase in packets delay and jitter was noticed especially for voice traffic.

Gin and Hunt [5] evaluated the impact of 802.11i security on network throughput performance. Several experimental scenarios were carried out in which the traffic volume and the number of traffic initiators (users) are changed. In their first set of experiments, where only two users coexisted in the network, minor throughput degradation is incurred even if the security level is maximized. In their second set of the experiments, where multiple users are transmitting through the network concurrently, slight throughput degradation resulted when adding security even with larger key length. With their results, Gin and Hunt [5] disagreed with many research papers that confirmed throughput deterioration when implementing increasing security.

In Begh and Mir work [6], IPTraffic was used to generate different rates of TCP and UDP traffic to quantify the impact of adding security on network throughput, transmission delay and packet loss. The results are obtained using five different security setups: no security, WEP-64, WEP-128, WPA-TKIP and WPA-AES. For 1 Mbps and 5 Mbps traffic rates, the results showed no major degradation in network throughput (except for WPA-AES), no significant increase in transmission time, and almost negligible packet loss ratio. However, when the traffic rate is increased to 12 Mbps, noticeable detraction in performance metrics was noticed. Begh and Mir [6] only performed their experiments with a single wireless station and they did not consider WPA2. Furthermore, they did not consider the end-to-end delay, but instead considered the transmission time which may not reflect the impact of using security protocols in WLAN.

In Baghaei and Hunt study [7] the impact of WEP protocol with various settings and key sizes on the performance of WLAN was analyzed. They used a network with three wireless clients and a wired server. Baghaei and Hunt [7] concluded that the stronger the security mechanism implemented the poorer the performance, although the degradation is certainly not linear. Moreover, they found that the response time is increased with stronger security mechanism and also when the network is congested.

As discussed in this section, none of the previous work has studied the impact of WPA1 and WPA2 on multimedia applications where delay and jitter are the key metrics that must be carefully tested. Furthermore, none of the previous work proposed a solution for the performance degradation that is caused by using strong security protocol as WPA2. Therefore, in this paper a special attention will be given to the impact of security protocols on the performance of WLAN in multimedia applications. Moreover, we will propose a new solution that will allow using a strong security protocol in WLAN while significantly minimizing the degradation in WLAN performance.

III. WLAN SECURITY PROTOCOLS

Similar to all wireless technologies, security in WLAN is considered one of its main weaknesses. The wireless medium is shared among the users and open access for any malicious attacker.

In this section, we provide a brief description of the most commonly used security protocols in WLAN.

A. Wired Equivalent Privacy

WEP protocol is the first security protocol for WLAN [8]. It was designed as a part of the original 802.11 standard. Its intended purpose was to provide security to WLAN equivalent to the security existing in the wired network. WEP uses RC4 stream cipher for confidentiality and CRC-32 for integrity. In this paper, we used 128-bit WEP protocol with 104-bit key and 24-bit initialization vector. WEP is known to be insecure since 2001. Tews and Beck [4] surveyed the most common successful attacks against WEP. Although WEP is widely used, it is agreed now that WEP with all its variations and modifications is considered insecure and should not be used. With the available tools such as Aircrack [9] one can break WEP security within minutes.

B. WiFi Protected Access

WPA1 [10] was designed to overcome the limitations and insecurity of WEP protocol. WPA1 implements most of the IEEE 802.11i standard. It uses Temporal Key Integrity Protocol (TKIP) [10] that uses a per-packet key. Hence, unlike WEP, a new 128-bit key is dynamically generated for each packet. Consequently, this prevents most of the type of attacks that compromised WEP. WPA1 replaced the insecure CRC used in WEP with a stronger message integrity check. Despite the fact that WPA1 addressed most of the problems that exists in WEP, it continues to show some limitations such as relying on stream cipher and cryptographically weak integrity (Michael algorithm). In Moen et al. [11] researchers discovered weaknesses in the temporal key hash of WPA1. Tews and Beck [4] presented the details of a potential attack to break WPA1.

C. *WiFi Protected Access II*

WPA2 [10] also referred to as IEEE 802.11i replaces WPA1 and implements the mandatory elements of IEEE 802.11i standard. It uses a new AES-based encryption mode CCMP that is highly secure. This resolved the security issue with TKIP in WPA. WPA2 provides Robust Security Network including two new protocols, the 4-way Handshake and the Group Key Handshake. Junaid et al. [12] and Khan et al. [13] showed that the initial counter value used in the CCMP can be predicted and that WPA2 is subject to dictionary attacks.

He and Mitchell [14] concluded that in CCMP, management frames and control frames are neither encrypted nor authenticated by the Link Layer encryption algorithm, and hence, being vulnerable to many threats discussed in their paper. In addition, they expected CCMP to have some impacts on performance because it requires some hardware upgrades.

IV. METHEDOLOGY AND HARDWARE IMPLEMENTATION

In this section, we describe the test bed used to conduct the experiment and the software tools used to collect and analyze the results. We also present the experimental scenarios conducted for studying the impact of security protocols on several network performance metrics such as throughput, delay, and jitter. Clock synchronization as well as a brief description for the performance metrics, especially the end-to-end delay, will be discussed in this section. Finally, traffic streams used in this experiment will be illustrated.

A. *Test Bed Description*

Two different network setup scenarios were used in this experiment. In the first scenarios, we only used two wireless host (laptops) and one access point as shown in Figure1.

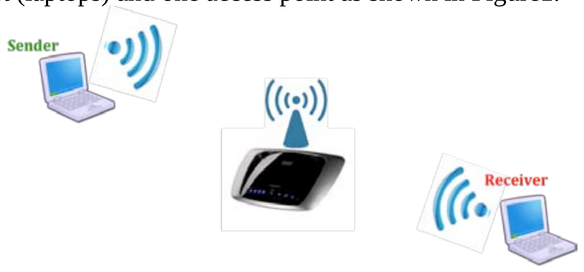


Figure 1. First Scenario

The specification of the devices is as follows:

1) *Linksys E2000 Cisco Advanced Wireless-N Router with the following specifications:*

- Supports up to 11 channels.
- Supports 802.11n, 802.11a, 802.11g, 802.11b, 802.3, 802.3u, 802.3ab Standards
- Security features : WEP, WPA, WPA2
- Security key bits: up to 128-bit encryption

2) *Acer notebooks with the following specifications:*

- Model: Acer Intel core i5, 2.24 GHz processor.
- NIC model: Realtek RTL Gigabit Family.
- Operating system: Ubuntu 11.04

The nodes were adjacent (2-3 meters) and other WiFi networks in the area were eliminated to avoid extraneous interference. All the experiments were performed in the same location and within a short period of time. Hence, the impact of multi-fading or coexistence of other WLANs will be the same for all the security protocols and scenarios.

The second scenario is shown in Figure 2 where four laptops existed in the network. All laptops were transmitting at the same time but communication was monitored between two nodes only. The other nodes were added to cause greater interference and congestion in comparison to the first scenario. This is expected to cause more processing, queuing, and backoff delay (possibly caused by collisions). In both scenarios, the streaming bit rate was about 500 kbps. Moreover, beside the video streaming, a large file was transferred between the sender and the receiver.

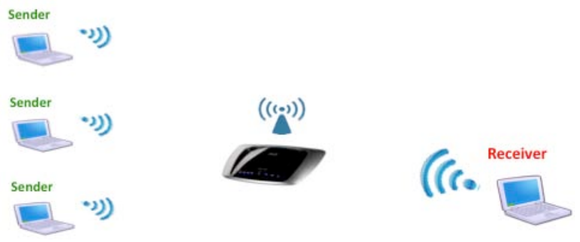


Figure 2. Second Scenario

B. *Clock Synchronization*

Accurate synchronization for the Laptops' clocks is crucial to avoid erroneous results on delay and jitter, such as a smaller packet arrival time compared to the packet sending time which may lead to unexplained negative delay. Several techniques can be used to ensure accurate synchronization. One technique is the manual adjustment of the devices' clocks based on a certain reference time. However, this method is prone to human error and therefore may be inaccurate and leading to wrong delay.

Another technique is issuing "net command" between the networked devices. The command guarantees instantaneous and accurate adjustment. However, shortly afterwards, the devices usually skew from the initial adjustment.

A third technique and the one applied in this experiment is the AtomTime Pro software [15], which frequently connects to global servers to adjust the time of the devices used in the experiment. As for the communication with the time server, an out-of-band connection was used to guarantee no interference with our traffic.

C. *Software*

Wireshark TM packet analyzer was used to capture and analyze network packets. In order to extract the required data from the large output files, we used AWK scripting language. This tool is used to extract certain packets information being read from Wireshark output files.

Some scripts were written using C# program for packets' matching. Throughput, delay, and jitter were calculated using the list of matched packets obtained from output files.

D. Performance Metrics

1) End-to-End Delay

Delay is one of the most important metrics for multimedia applications. Unlike file transfer, multimedia applications could tolerate packet loss but not a high delay. Delay can be classified into four main types or causes:

a) Transmission delay:

This is the time needed to transmit all the bits of a packet. It is measured by dividing the frame size in bits by the wireless link transmission speed in bits per second. In our case, this time is expected to be in the order of tens of milliseconds.

b) Queuing delay:

This is the time when the packet waits its turn to be transmitted in the queue of a router or host. This delay could be neglected if only few hosts are transmitting in a lightly loaded network. We expect this delay to take place only with the second scenario.

c) Propagation delay:

This is the time needed for one bit to travel from the sending to the receiving host. This delay is measured by dividing the distance between the sender and the receiver by the signal speed (usually equals the speed of light). Usually, this time is small and could be neglected.

d) Processing delay:

This is the time taken to process the packets by the router and hosts. This time is expected to be more significant if security is enabled in the wireless router given that encryption and authentication verification require significant processing and may cause longer delay.

2) Packet Jitter

The packet jitter measures the variation in the end-to-end delay from packet to packet. Multimedia application could tolerate some delay but large jitter could cause a greater damage to the consistency of their operation.

3) Throughput

Throughput is the average number of bits that is sent over the network per second. Usually, it is the most significant metric if the main purpose of the network is to transfer large files.

E. Traffic Streams

In our scenarios, we used video steaming traffic between wireless hosts. Multimedia traffic uses real-time streaming protocol (RTSP) to control the transmission of a media stream. It also relies on real-time protocol (RTP), which runs on the top of UDP or possibly TCP [16]. These multimedia protocols add sequencing and timestamp information to the transferred packets. Hence, their operationability will be significantly affected by the packets delay and jitter. Moreover, a heavily-loaded network will have more collision and backoff time incurring longer delays.

V. RESULTS

In this section, results from both scenarios are discussed in terms of delay, jitter, and throughput. For statistical validation, all the experiments were repeated 10 times and the averages were considered in our results.

A. Delay

Figure 3 shows the total average end-to-end round trip delay in milliseconds for the packets transmitted between two wireless hosts for WEP, WPA1 and WAP2 security protocols. Since our goal was to study the impact of security protocols on the delay, then change in delay is what matters not the value of the delay itself. Hence, we neutralized the results by subtracting the delay with disabled security from the other three cases.

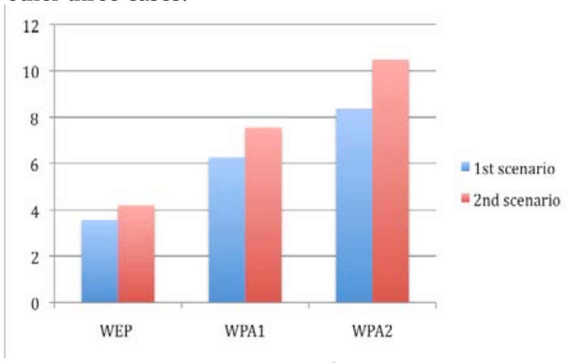


Figure 3. Delay

Figure 3 shows that for both scenarios, delay is increased when security protocols are enabled in WLANs. As expected, WPA2 which is the most complicated security protocol for using the strong AES encryption, showed the highest delay. On the other hand, WEP protocol, which is the lightest protocol for relying on the simple RC4 stream cipher, showed the lowest delay.

For all the security protocols, scenario two has higher delay than scenario one. This is due to the fact that the network traffic in scenario two is much larger than that in scenario one. Larger traffic leads to a larger number of collisions, which in turn results in longer backoff times and longer delay.

Figure 3 also shows that the difference between the delay of WPA1 and WAP2 with scenario two is larger than that with scenario one. This indicates that the processing delay is not the only reason that causes the delay with WPA2. It can also be caused by the overhead in WAP2 protocol handshake messages.

B. Jitter

Figure 4 shows the packet jitter in milliseconds for the packets transmitted between two wireless hosts for WEP, WPA1 and WAP2 security protocols. Since our goal was to study the impact of security protocols on the jitter rather than value of the jitter per se, we neutralized the results by subtracting the jitter with disabled security from the other three cases.

Figure 4 shows that for both scenarios, the jitter is increased when security protocols are enabled in WLANs. As expected, WPA2 showed the highest jitter and WEP showed the lowest.

Similar to Figure 3, for all security protocols, scenario two has higher jitter than scenario one. The same justification also applies here.

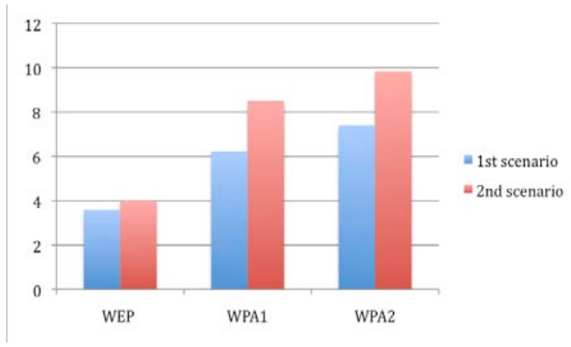


Figure 4. Jitter

Figure 4 also shows that the difference between the jitter of WPA1 and WAP2 with scenario two is even larger compared to scenario one, and for the same reasons that caused the delay to be higher.

C. Throughput

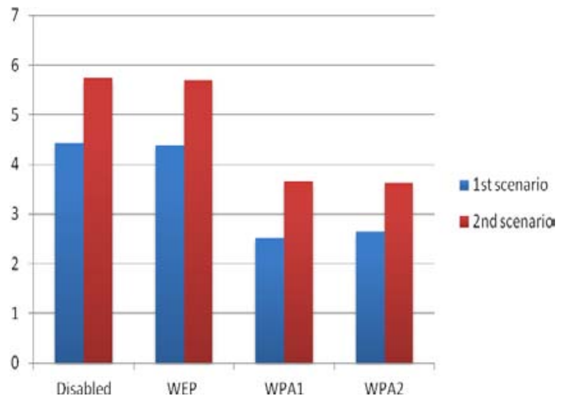


Figure 5. Throughput

Figure 5 shows the throughput in Mbps for the two wireless network scenarios with the four cases: disabled security, WEP, WAP1, and WPA2. The figure shows that WEP has no impact on the throughput. This observation may be explained by the fact that WEP is light and does not cause large overhead. Even for WPA1 and WPA2 protocols, the throughput was slightly affected which may be due to the fact that the multimedia traffic was not heavy on the network. The throughput in the second scenario was higher than the first scenario because of the additional file transfer.

VI. PROPOSED SOLUTION

In this section, we propose an outline for a new approach to achieve security in WLAN while reducing the impact on the network performance. The proposed outline is based on the fact that disabling security in WLAN results in higher throughput and significantly lower delay and jitter. This fact was also agreed upon in other research papers [1, 2, 6, 7].

Potorac and Balan [17] studied the impact of security overheads on 802.11 WLAN throughput. They provided theoretical analysis about the impact of the three security protocols on system's performance. Their findings indicated that theoretically, the impact of WEP, WPA1, and WPA2 security overheads is insignificant for large packets. They concluded that the processing devices and computing resources that are available at the level of the radio station need more processing power for encryption. Thus, according to Potorac and Balan [17] it is possible to observe smaller throughput or larger delay given that the communication processor cannot encrypt or decrypt the data flow at the necessary speed. Lars [18] presented measurements that match the theoretical results presented by Potorac and Balan [17]. Based on our results and the findings we discussed for Potorac and Balan [17] and Lars [18], we propose a solution to overcome the performance degradation caused by using strong security protocols in WLANs. The actual implementation, deep analysis, and performance evaluation of this solution is beyond the scope of this paper and is left for future extension of this work.

The idea of the proposed solution is to move all the security to the end systems and use WLAN with disabled security. Using open access (disabled security) WLAN will give us the following advantages:

- Higher throughput, also shown in [1, 2, 6, 7].
- Lower delay or response time, also shown in [2, 3, 6].
- Less percentage of dropped packets, also shown in [1].

Authentication and encryption will be applied to the data prior to its delivery to the low processor performance wireless card.

At the hosts, the encryption and all other security services, such as authentication and key exchange, will be done before the data is passed to the radio card by the device (possibly laptop) processor. These processors are usually powerful and will process the encryption in much less time compared to the radio card processer. The delay and jitter are both expected to be significantly small and possibly negligible.

At the access point, the increasing complexity of security protocols signifies the need to improve the performance of network processing hardware for real-time cryptographic processing. The cryptographic algorithm throughput and delay can be significantly improved by implementing the algorithm in specialized processors using ASIC solution or FPGA implementation. While ASIC designs are superior in performance, FPGA implementation has the advantages of low cost, reprogramability and short time to market. Numerous ASIC and FPGA designs have been proposed in the literature such as those in Wang et al. [19] and Chang et al. [20]. Wang et al. [19] presented an ASIC implementation with on-the-fly key expansion and reconfigurable core architecture. The design provides a throughput of up to 3.75 Gb/s at 102 MHz. Chang et al. [20] discussed an FPGA implementation of 32-bit AES algorithm. The design has a low area of 156 slices and a throughput of 876 Mbps.

In terms of hardware implementation of the algorithm, our preliminary analysis indicates that the synthesized HDL design would result in a gate count of 50K gates. An ASIC implementation using 90-nm technology will have an area of 1mm^2 and a power dissipation of 150 mW with a throughput of about 40 Gbps. An FPGA implementation, on the other hand, would require about 10K slices with a throughput of about 15 Gbps.

Figure 6 illustrates the proposed solution. The laptops will use the Fast Security Add-on (FSA), which will be an add-on software to the wireless card driver. FSA will be responsible for all the security algorithms and will be processed by the powerful fast laptop processor (could be dual core, i3, i5, or i7). On the other hand, at the access point the ASIC/FPGA processor will be responsible for all the processing that maybe required by the security protocols.

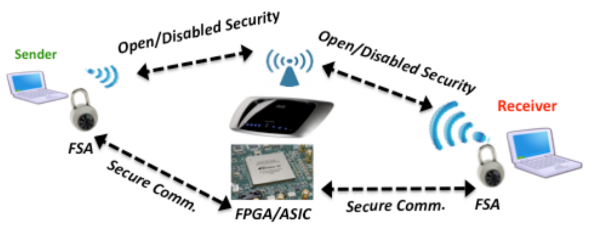


Figure 6. Proposed Solution

The proposed solution is considered successful only if it causes a negative impact on the performance that is less than the impact caused by WPA2. As for the impact on the throughput, Potorac and Balan [17] analytically proved that the overhead in WPA1 and WPA2 are 20 and 16 bytes per packet, respectively. Thus, it is less likely that the proposed solution will have any impact on the throughput. In our future work, we are planning to implement and test the FSA tool on several laptops platforms to analyze its impact on the delay and jitter. The specialized processors using ASIC solution or FPGA implementation will also be tested for delay and jitter.

VII. Conclusion and Future Work

This paper examined the effect of several security protocols on the performance of a WLAN with multimedia applications. Throughput, delay, and jitter for four security settings: disabled security, WEP, WPA1, and WAP2 were analyzed. The results showed a significant degradation in performance occurring when enabling security protocols in a WLAN. Specifically, delay and jitter, which are key metrics for such applications, were significantly increased. The increase was more noticeable when more wireless hosts exist in the network. Therefore, we proposed a solution outline to achieve strong security in WLAN without noticeable performance degradation. The solution proposes that the security processing be conducted by the powerful host processors rather than by the radio card processors. As for the wireless access point, adding ASIC or FPGA processor is suggested for performing heavy security processing. Our study has shown that in theory, the proposed solution is effective. However, future empirical research is needed to practically prove its effectiveness.

Moreover, in order to better comprehend the possible impact of the increase in delay and jitter with WPA2 one should consider objective and subjective video quality metrics. However, theses quality metrics and the details and prototype of the proposed solution are left for future research.

References

[1] E. Barka and M. Boulmalf, “On The Impact of Security on The performance of WLANS,” Journal of Communications (JCM), Vol. 2, pp. 10-17, June 2007.

[2] S. Kolahi, S. Narayan, Y. Sunarto, D. Nguyen, and P. Mani, "The Impact of Wireless LAN Security on Performance of Different Windows Operating Systems," in Proceedings of the IEEE Symposium of Computers and Communications(ISCC), pp. 260-264, July 2008.

[3] M. Boulmalf, E. Barka, and A. Lakas, “Analysis of the effect of security on data and voice traffic in WLAN,” ACM journal of Computer Communications, vol. 30, pp. 2468-2477, 2007.

[4] E. Tews and M. Beck, "Practical attacks against WEP and WPA," In Proceedings of the second ACM conference on Wireless network security (WiSec '09), pp. 79-86, 2009.

[5] A. Gin and R. Hunt, “Performance Analysis of evolving Wireless IEEE 802.11 Security Architectures,” Proceedings of the 8th International Conference on Mobile Technology, Applications, and Systems, Vol 2, pp. 101-106, 2008.

[6] G.R. Begh and A.H. Mir, "Quantification of the Effect of Security on Performance in Wireless LANs," Proceeding of the 3rd International Conference on Emerging Security Information, Systems and Technologies (SECURWARE '09), pp. 57-62, June 2009.

[7] N. Baghaei and R. Hunt, "Security Performance of Loaded IEEE 802.11b Wireless Networks," Journal of Computer Communications, vol. 27, pp. 1746-1756, 2004.

[8] LAN MAN Standards Committee of the IEEE Computer Society, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification, IEEE Std 802.11-1999 edition, IEEE, New York, 1999.

[9] <http://www.aircrack-ng.org> 27.7.2012

[10] ANSI/IEEE standard 802.11i., “Amendment 6 Wireless LAN Medium Access Control (MAC) and Physical Layer (phy) Specifications”, Draft 3. (2003).

[11] V. Moen, H. Raddum, and K. Hole, "Weaknesses in the temporal key hash of WPA," In Proceeding of ACM Mobile Computing and Communications Review (SIGMOBILE 04) , pp. 76-83, 2004.

[12] M. Junaid, Muid Mufti, and M.Umar Ilyas, "Vulnerabilities of IEEE 802.11i Wireless LAN CCMP Protocol," Journal of Transactions on Engineering, Computing and Technology, vol. 11, pp. 228-233, February 2006.

[13] M. Khan, A. Cheema, and A. Hasan, "Improved Nonce Construction Scheme for AES CCMP to Evade Initial Counter Prediction," Proceedings of the 9th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 307-311, 2008.

[14] C. He and J. Mitchell, "Security Analysis and Improvements for IEEE 802.11i," Proceeding of NDSS, 2005.

[15] < http://www.atomtime.com/> 27.7.2012

[16] J. Kurose and K. Ross, Computer Networking: A Top-Down Approach (5th Edition), Addison-Wesle, 2010.

[17] A. Potorac and D. Balan, "The Impact of Security Overheads on 802.11 WLAN Throughput," Journal of Computer Science and Control Systems, vol. 2, pp. 47-52, 2009.

[18] McCarter Harold Lars, “Analyzing Wireless LAN Security Overhead”, research report at Virginia Tech University, available at http://scholar.lib.vt.edu/theses/ available/etd-04202006-080941/unrestricted, 2006.

[19] M. Wang, C. Su, C. Horng, C. Wu, and C. Huang, "Single- and Multi-core Configurable AES Architectures for Flexible Security," in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 18 , pp. 541-552, 2010.

[20] C. Chang, C. Huang, K. Chang, Y. Chen, and C. Hsieh, "High throughput 32-bit AES implementation in FPGA," in Proceeding of IEEE Asia Pacific Conference on Circuits and Systems, 2008.