
GOBIERNO DE ESPAÑA
MINISTERIO DE ENERGÍA, TURISMO Y AGENDA DIGITAL
incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD
Mondragon Unibertsitatea
JNIC 2018
Actas de las IV Jornadas Nacionales de Investigación en Ciberseguridad

Actas de las Cuartas Jornadas Nacionales de Investigación en Ciberseguridad

Donostia-San Sebastián, Gipuzkoa, 13-15 de Junio, 2018

Editores: Urko Zurutuza, Mikel Iturbe, Enaitz Ezpeleta e Iñaki Garitano.

Publicado por: Servicio Editorial de Mondragon Unibertsitatea
ISBN: 978-84-09-02697-5
Diseño de la portada: Roberto Uribeetxeberria

Prefacio

La ciberseguridad está de moda. Son ya muchos los años en los que esta comunidad ha realizado formidables trabajos de investigación, formaciones regladas y no regladas, transferencias de conocimiento, acciones de sensibilización, internacionalización de los propios investigadores y un gran etcétera. Pero es ahora, en este año 2018, cuando los conceptos de ciberseguridad han traspasado todas las fronteras imaginables entre la industria, la academia, la ciudadanía y las instituciones, y parece que los mensajes están calando. Han sido las instituciones las que, al percibir esa gran necesidad (de servicios de ciberseguridad, de protección de empresas e infraestructuras críticas, del grado de vulnerabilidad a todos los niveles), y enfrentarla con un nivel de amenaza creciente, ha decidido impulsar nuevos programas, aumentar presupuestos y ayudas, y acompañar a las organizaciones en busca de un mayor nivel de protección global, y a su vez generar riqueza por medio de nuevas organizaciones que generen valor con la venta de productos y servicios de ciberseguridad.

Tras fraguarse las primeras Jornadas Nacionales de Investigación en Ciberseguridad en 2015 en León, organizadas por el grupo RIASC de la Universidad de León, las mismas han ido creciendo año a año, celebrándose nuevas ediciones en Granada en 2016 (organizada por el grupo NESG de la Universidad de Granada), y en Madrid en 2017 organizadas por la Universidad Rey Juan Carlos, siempre con la co-organización y apoyo incondicional de INCIBE. Durante este tiempo se ha generando una estructura que vela por su continuidad y máxima calidad (a través del comité ejecutivo que rota anualmente), y una reglamentación que guía las JNIC en sus principios y estilo propios.

En la edición de 2018, la sede se ha situado en Donostia-San Sebastián, y ha sido el grupo de Análisis de Datos y Ciberseguridad de Mondragon Unibertsitatea el que ha tenido el placer de co-organizarla junto con INCIBE tras ser elegida en la edición anterior. Y probablemente, es en Euskadi donde más patente ha sido el auge de la ciberseguridad. A lo largo de este año 2018, El Gobierno Vasco ha creado el Centro Vasco de Ciberseguridad (o Basque CiberSecurity Centre - BCSC), con el objetivo principal de generar cultura de ciberseguridad en Euskadi dinamizando la actividad económica relacionada con la aplicación de la ciberseguridad y fortalecer el sector profesional; La Diputación Foral de Gipuzkoa está finalizando el proceso de creación de ZIUR, o Industrial Cyber Security Center-Gipuzkoa, con el fin de ayudar a proteger un tejido industrial de primer orden como es el guipuzcoano; prácticamente las 4 universidades de Euskadi cuentan con actividad de investigación en ciberseguridad, y se pueden contar 4 centros tecnológicos con actividad creciente. Durante el año 2018 se han podido contabilizar más de 5 eventos importantes de ciberseguridad (IndusSec, CyberSec, NextSecure, JNIC, Euskalhack...).

Esta edición de JNIC se celebra en el emblemático Palacio de Miramar, en plena bahía de la Concha, en parte dentro del paraguas de los Cursos de Verano de EHU-UPV. Contamos con Dr. Guillermo Suarez-Tangil (Kings College London), Dra. Ana Isabel González-Tablas (Universidad Carlos III de Madrid) y Juan Díez González (INCIBE) como presidentes de los programas de Investigación, Formación e Innovación educativa, y de Transferencia tecnológica respectivamente.

En total se recibieron 73 trabajos, de los que se aceptaron 45, con un 62 % de ratio aceptación. De ellos, los trabajos de investigación se han expuesto divididos en 9 temáticas diferentes, que son aquellas que definitivamente trabaja el ecosistema investigador nacional: Detección de ataques y cibercrimen, seguridad en dispositivos móviles, seguridad del software, criptología y criptomonedas, seguridad IoT, seguridad en redes industriales, conjuntos de datos para investigación en ciberseguridad, privacidad y autenticación, y gestión de la seguridad. Temáticas en las que los investigadores muestran estar en primer nivel de propuestas y resultados. Hay una sesión especial dedicada a la formación e innovación educativa en ciberseguridad, y se presentan los finalistas de los premios al mejor trabajo de estudiante. Además, a lo largo de esta edición finaliza el primer certamen de Retos de JNIC, como parte del track de transferencia. En el mismo, 7 grupos de investigación presentan las mejores soluciones a los retos planteados en las JNIC 2017 de Madrid, que reciben sus premios. Después las empresas y organizaciones presentan los retos de esta edición. Además, contamos con dos conferencias plenarias de primer nivel. Por un lado, Dr. Álvaro Cárdenas, de University of Texas at Dallas ofrece una conferencia repartida en dos apartados: Seguridad del IoT, y seguridad de sistemas de control industriales. Por el otro lado, Dr. Arturo Ribagorda recibe un caluroso homenaje debido a su larga trayectoria en ciberseguridad, y ofrece la segunda conferencia plenaria.

El equipo de organización de las JNIC 2018 quiere finalmente agradecer a los patrocinadores sin los cuales no habría sido posible ofrecer 3 días de intensas actividades: la Corporación Mondragon como patrocinador Platino, a los centros BCSC y ZIUR como patrocinadores Oro y a las empresas Inycom, Exclusive Networks, y LKS como patrocinadores Bronce. Así mismo, La Red de Excelencia Nacional de Investigación en Ciberseguridad RENIC mantiene su compromiso como patrocinador científico. A todos, nuestra más sincera gratitud.

Eskerrik asko denoi, eta jardunaldi oparoak izan daitezela espero dugu!

El Comité Organizador

Organización JNIC2018

Comité Ejecutivo

- Marta Beltrán Pardo (*Universidad Rey Juan Carlos*)
- Juan Caballero (*IMDEA Software Institute*)
- Andrés Caro (*Universidad de Extremadura*)
- Luis Javier García Villalba (*Universidad Complutense de Madrid*)
- Juan Díez González (*INCIBE*)
- Juan Tapiador (*Universidad Carlos III de Madrid*)
- Gregorio Martínez (*Universidad de Murcia*)
- Urko Zurutuza Ortega (*Mondragon Unibertsitatea*)

Comité Organizador

General Chair:

Urko Zurutuza (*Mondragon Unibertsitatea*)

Program Chairs:

De investigación:

Guillermo Suarez-Tangil (*King's College London*)

De formación e innovación educativa:

Ana Isabel González-Tablas (*Universidad Carlos III de Madrid*)

De transferencia tecnológica:

Juan Díez (*INCIBE*)

Financial Chair:

Sonia Anduaga (*Mondragon Unibertsitatea*)

Arrangement Chair:

Enaitz Ezpeleta (*Mondragon Unibertsitatea*)

Publication Chair:

Mikel Iturbe (*Mondragon Unibertsitatea*)

Publicity Chair:

Iñaki Garitano (*Mondragon Unibertsitatea*)

Comités de programa:

De investigación:

Hector Alaiz Moreton
Universidad de Leon

Cristina Alcaraz
Universidad de Málaga

Jorge Blasco Alís
Royal Holloway, University of London

Guillermo Calvo Flores
INCIBE

David Camacho
Universidad Autónoma de Madrid

José Camacho
Universidad de Granada

Enaitz Ezpeleta
Mondragon Unibertsitatea

Iñaki Garitano
Mondragon Unibertsitatea

Hugo Gascón
TU Braunschweig

Manuel Gil Pérez
Universidad de Murcia

Felix Gomez Marmol
Universidad de Murcia

Alessandra Gorla
IMDEA Software Institute

Luis Hernández
Consejo Superior de Investigaciones Científicas

Mikel Iturbe
Mondragon Unibertsitatea

Ilias Leontiadis
Telefonica I+D

Javier López
Universidad de Málaga

Jorge López Hernández-Ardieta
Indra

Alejandro Martín
Universidad Autónoma de Madrid

Gregorio Martínez
Universidad de Murcia

Srdjan Matic
IMDEA Software Institute

Hector D. Menéndez
University College London

Alfonso Muñoz
i4s-BBVA

Sergio Pastrana
Cambridge University

Ricardo J. Rodríguez
Centro Universitario de la Defensa de Zaragoza, Academia General Militar

Jose Such
King's College London

Juan Tapiador
Universidad Carlos III de Madrid

De formación e innovación educativa:

Isaac Agudo Ruíz
Universidad de Málaga

José Francisco Barea López
Universidad Politécnica de Madrid

César Cáceres Taladriz
Universidad Rey Juan Carlos

Miguel Carriegos Vieira
Universidad de León

Noemí De Castro García
Universidad de León

Miguel Fernández Arrieta
Mondragon Unibertsitatea

Félix Jesús García Clemente
Universidad de Murcia

José Antonio Gómez Hernández
Universidad de Granada

Jorge López Hernández-Ardieta
MINSAIT

Andrés Marín López
Universidad Carlos III de Madrid

José Carlos Sancho Núñez
Universidad de Extremadura

Adriana Suárez Corona
Universidad de León

De transferencia tecnológica:

Marcos Arjona
ElevenPaths

Juan Caballero
IMDEA Software Institute

Víctor Villagrá
Universidad Politécnica de Madrid

ÍNDICE DE CONTRIBUCIONES

Track de Investigación I: Detección de Ataques y Cibercrimen I

Track de Investigación II: Detección de Ataques y Cibercrimen II

Track de Formación y Talento

Track de Investigación III: Seguridad en Dispositivos Móviles

Track de Investigación IV: Seguridad del Software

Track de Investigación V: Criptología, Criptomonedas

Track de Investigación VI: Seguridad IoT

Track de Investigación VII: Seguridad en Redes Industriales

Track de Investigación VIII: Conjuntos de datos

Track de Investigación IX: Privacidad y Autenticación

Track de Investigación X: Gestión de la Seguridad

Discovering and Plotting Hidden Networks created with USB Devices

Francisco Ramírez Pablo González Carmen Torrano José María Alonso

CDO, Telefónica
Madrid, Spain
{pablo, carmen.torrano, chema}@11paths.com
franciscojose.ramirezvicente@telefonica.com

Abstract- **USB is still a dangerous vector of attack. Attacks such as Stuxtnet make this evident. Sharing USB devices between machines that are physically or logically isolated can stablish connections between them, building what is called a *Hidden Network*. Being aware of such links is essential for guarantying the security of the network, machines connected and data within them. This paper presents a tool able to detect these links automatically, both remotely and locally. These links are explicitly plotted in graphs, making them visible and allowing taking security measures against threats. Furthermore, this tool can be used for forensic purposes, for example in cases of data exfiltration, since it can plot the trazability of a given USB device within a network, besides showing which USB devices were connected to a certain computer.**

Index Terms- **Hidden networks, USB devices, forensics, network security, network connections.**

Tipo de contribución: *Investigación original (límite 8 páginas)*

I. INTRODUCTION

Several security incidents reveal that USB should be paid attention since it could be used for malicious intentions. Even for air-gapped networks, USB represents an attack vector. Then, having a computers network not connected through Ethernet cable or Wi-Fi to other networks is not enough and any type of external connection for computers may constitute a threat.

In network analysis, usually attention is paid to connections at link level, such as Ethernet, Wi-Fi connections, etc. However, USB connections are often ignored.

USB connections also represent a mean to spread attacks, infect with malware or steal important data. In fact, several famous attacks have been performed through the connection of USB devices. Next, a review of the most important incidents is presented.

Firstly, Stuxnet [1] is mentioned, given its importance and repercussion. This malware is designed to target supervisory control and data acquisition (SCADA) systems that monitor and control industrial processes. In fact, in 2010 it was able to infect a Nuclear Power Plant in Iran. The worm took control of a thousand of machines involved in nuclear material production and gave them instructions that leave them inoperable. Stuxnet could access the network via an infected USB device. Then it scanned the network and propagated though it, reprogramming the software that controls certain machines involved in the nuclear process. The consequence was that thousands of those machines were left out of service.

Other popular malware is Brutal Kangaroo. This is one of the tools developed by CIA and included in Vault 7. This malware has been specially designed to infect air-gapped computers, i.e., isolated computers that are not connected to Internet. The term air-gapped makes reference to the space between the system and Internet. The first step of the operation of this malware is infecting a computer connected to Internet inside the target network and installing the malware. From there, the infection to air-gapped computers is done by means of a USB device infected with some malware.

There are other threats related to USB, such the so called "USB killer" [2], that damages the computer it is connected to by accumulating part of the electric energy of the computer and lately sending that energy brusquely against it. Rubber Ducky is a keyboard included in a USB device that types in a computer as soon as it is connected. As a keyboard, it can type malicious code or launch programs located either in the victim device or in the USB itself.

All these facts denote that attention should be paid to the connection of USB devices, since they represent a potential threat. These USB connections can create what is called a *Hidden Network*, i.e., networks created through the use of USB devices that allow communication between physically or logically isolated computers. This is the case of Brutal Kangaroo, where all computers infected are part of a hidden network where elements within it can communicate and exchange data. This takes major importance considering the value of data in a computer and also circulating between the nodes of a network, in both the corporative and personal spheres.

In order to help in the automation of the discovery of hidden networks, in this paper we present a tool that makes explicit those hidden links created by the connection of USB devices. Our tool can be used to collect this information remotely and locally. Furthermore, since the connection with the computers in the network can be done by using different protocols, we have developed software that covers different scenarios. Our solution collects the information of the computers within the domain in a file and it even plots the hidden USB links in a graph shape. It makes possible to be aware of such links, what is very important for protecting and

securing the network and devices connected to it. The discovery of such links can be useful for example in cases of data exfiltration, to help in the forensic analysis by revealing which USB devices where connected to a computer and tracing the USB within the network. This capacity is not exclusive for USB memories but also for devices connected through USB, such an external disk.

By mapping a network, a greater level of understanding regarding the network and the threats inside it can be achieved. This is important from the security point of view in order to adequately protect and secure all elements of the architecture, having more insights of the borders inside the network to mitigate intrusions, detecting attacks, or preventively carry out the implementation of safety measures.

The rest of this paper is structured as follows: Section II gives insights about hidden links, showing an example that presents their potential risks. Section III describes the registry and the particular branch where information about USB devices connected to a computer can be collected. Section IV explains the structure of our solution, that works in different modes: remotely and locally. Section V shows the implementation details of our software for discovering hidden networks, concretely of our tool and Powershell scripts. Section VI presents details about graph plotting. Section VII covers the experimental stage where we have tested our software in different environments. Section VIII treats possible limitations of the tool. Mitigation measures against hidden networks are explained in section IX and finally, the conclusions drawn as result of this study and future work are summarized in section X.

II. NETWORK ISOLATION AND THE CONNECTION OF USB DEVICES

In order to bring clarity in the understanding of hazards in Hidden Networks created from USB devices, a simple example is presented hereafter. In this example it is assumed that an organization has a network formed by three VLANs. The first VLAN contains a computer called *A* and another one called *B*. The second VLAN contains computers *C*, *D* and *E*. And the last VLAN contains the computer *F*. Every computer in a VLAN has connectivity with other computers in the same VLAN. However, computers in different VLANs cannot communicate. The network outline of the network architecture defined in this example is represented in Fig. 1a. In that figure it can be seen that computers are isolated through different VLANs.

Assuming employees of this organization exchange data by means of USB devices, there is a high probability that this information passes from one VLAN's computer to another. This scenario is represented in Fig. 1b, where users of computers *F* and *E* are assumed to exchange information through a USB device and a hidden network is created between both computers. This information can be represented with two nodes, *E* and *F*, and an edge is projected between the computer where the USB device was introduced first and the second computer.

This would represent a serious threat to security since VLANs purpose is creating independent logic networks within a network, providing fragmentation and isolation in a physical network. This link would allow a communication between those computers that did not exist before. This implies that a new communication channel is created through the connection of the USB device in different moments to both computers, that are supposed not be able to communicate.

Therefore, adding the USB device is itself a source of threats, since a hidden network can be created inside the organization.

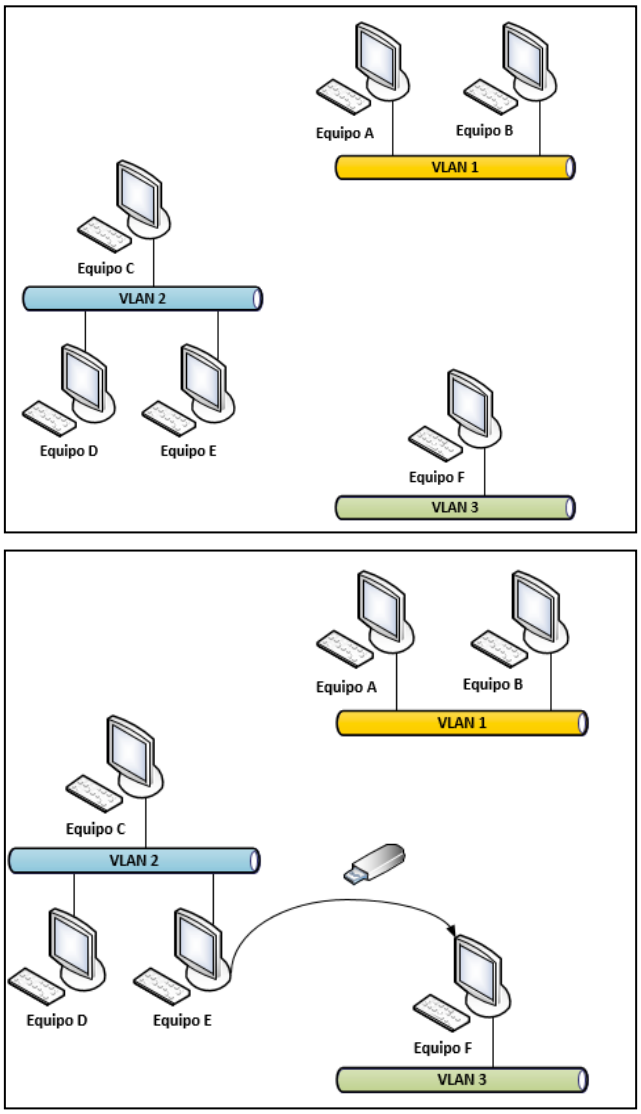


Fig. 1a: Network outline with computers connected to different VLANs. Fig. 1b: Hidden link created in the previous network outline.

III. REGISTRY OF USB DEVICE CONNECTIONS

When a user connects a USB device in a Windows system, a series of entries are created in the Windows registry.

According to the Microsoft Computer Dictionary [3], the registry is a hierarchical database where Windows systems store information necessary to configure the system for one or more users, applications and hardware devices. It contains information like applications installed on the computer, ports used or hardware existing on the system.

Since the registry controls the peripheral devices, when a USB device is connected, certain information is stored there. In particular, the USBStor key created in the Windows system registry saves information regarding all different devices inserted in the computer. Each USB connected to the computer generates a new item in the registry. Specifically,

the branch of the registry where this information is stored is HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USBSTOR [4].

The following information can be collected from this branch regarding USB devices connected to one computer:

- Device name.
- Class.
- ClassGUID.
- HardwareID.
- Service provided by the device, e.g. a hard disk.
- Driver.

An example of these fields of an USB device connected and stored in the USBStor key is shown in Fig. 2.

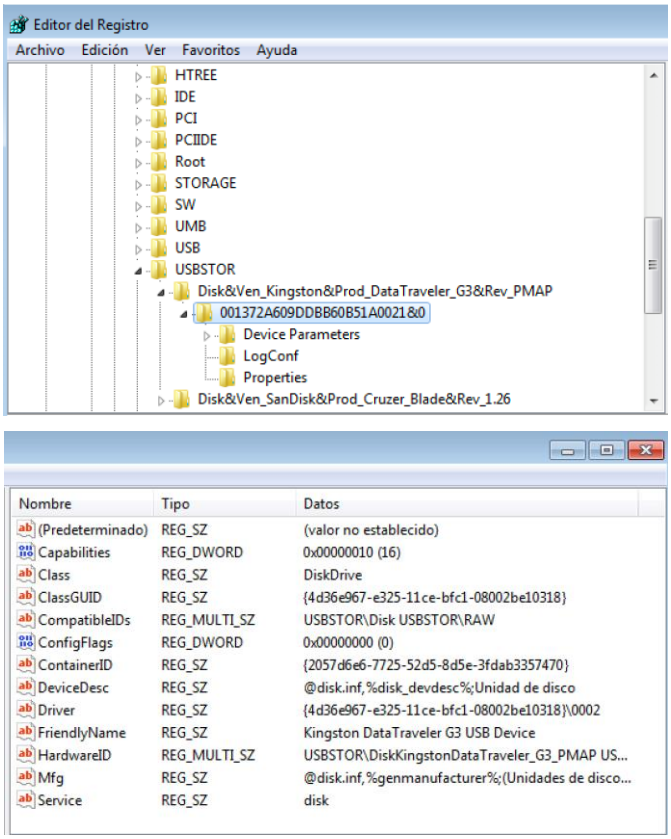


Fig. 2: Registry visualization of the USB devices inserted.

From this information stored in the registry it is possible to know who is sharing a USB device and with whom, what is very useful to discover hidden links of a hidden network. As mentioned, this provides useful information in forensic investigation.

IV. SOLUTION DESCRIPTION

In this paper, we present a tool that is able to automatically discover hidden networks, what could be applied in forensic analysis. The tool can be used in two different modes: remote or local.

- Remote. In this case the software collects the information of the registry regarding connections of USB devices to computers in a given domain. It is possible to specify the set of computers of which the information will be collected. This option is designed to be used by system administrators and it is also useful in security audits.
- Local. The software collects the information of the registry of the local computer where it is run.

In the local case, it is not necessary to connect to any computer and only the information from the registry of the computer running the tool is collected. Contrarily, in the first case, the first step of the hidden network discovery process is connecting to the computers of the network that are selected to be analysed. The idea is that a central node with Active Directory runs the software and collects the information in the registry of all computers selected of the domain. The software then receives and stores information collected from the nodes. We have chosen the use of Active Directory for convenience and simplicity, since it facilitates the administration of computers in the domain. If Active Directory is not used, the administrator credentials of each computer would be needed.

There are different methods and protocols that can be used for stablishing the connection with nodes, depending on the architecture and technologies of the network. For this purpose, we have selected different technologies:

- WS-Management Protocol [5] and WMI [6]. Using the Windows Remote Management (Win RM) implementation [7]. We include WMI also in this section.
- SMB-PSExec [8-9].

Next, we give a brief description of these technologies.

A. *WS-Management Protocol and WMI*

This protocol is an open standard of the Distributed Management Task Force (DMTF) for accessing and exchanging management information with computers that implement this protocol. It is based on the Simple Object Access Protocol (SOAP) [10]. Windows Remote Management (Win RM) is the Windows implementation of this protocol. It allows to remotely run management scripts. One of the advantages of this protocol is that hardware and operating systems from different vendors can interoperate.

Related to Win RM Windows also provides an infrastructure for management of data and operations called Windows Management Instrumentation (WMI). It provides scripting languages to manage computers both locally and remotely. Additionally, it can supply management data to other components, such as Win RM.

B. *SMB-PSExec*

SMB stands for Sever Message Block (SMB). This is an application layer protocol with request-response nature used for sharing files, printers, serial ports and communication abstractions between nodes connected to the same network.

PsExec was born as an alternative to telnet, easier to configure and avoiding the need to install software on the remote computers to be accessed. It allows executing processes on other systems and launching interactive command-prompts or enabling tools on remote systems.

SMB-PsExec implements remote process execution, being possible to run programs on remote computers. It can send a program to a remote computer, run it and read the result. It uses a configuration file for setting certain parameters.

Our solution covers all these scenarios. Since networks may use different technologies, we have implemented this variety of protocols so that our solution can be used in scenarios with different network architectures or configurations.

In particular, we provide a tool programmed in Python for covering the WMI and local cases. Additionally, we have programmed several scripts in Powershell for the SMB-PsExec and Win RM cases.

When it is necessary to connect remotely, all our solutions provide a file in CSV format with the information collected from the selected computers regarding the USB devices that have been connected to those machines. Additionally, the tool also provides JSON format. Furthermore, our tool is able to plot this information in a graph shape. Computers are plotted as and edges represent the hidden links between connected machines. In this way, the subtle connections are made explicit, what is very helpful for improving the security of organizations.

In the next section, we give more details about the implementation.

V. IMPLEMENTATION

As mentioned, since the connection mode and protocols to connect with the computers in the network may vary, we have implemented different software to cover all cases. Next, we explain the details about how our software works. On the one hand the WMI and local cases are covered by our tool developed in Python. On the other hand, we explain the architecture of the Win RM and SMB-PsExec cases, implemented with scripts that use Powershell 3.0 commands. Powershell is an object-oriented command line from Microsoft, which has a simple and powerful interaction with any structure inside a Microsoft operative system.

A. *WMI and local*

We have developed a tool programmed in Python for covering these two cases. The architecture of this implementation can be seen in Fig. 3.

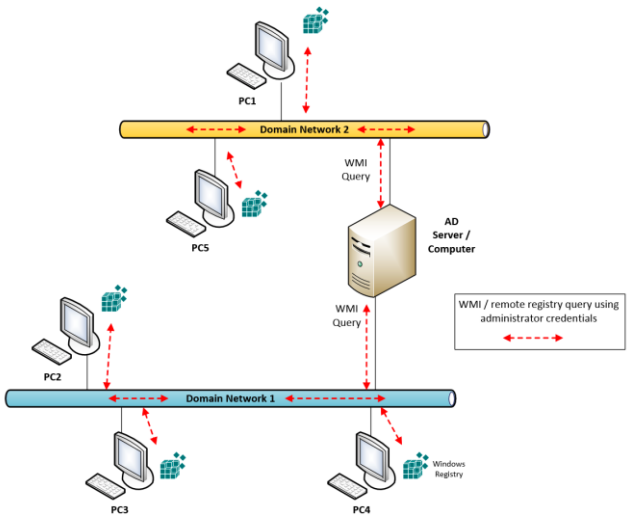


Fig. 3: Tool execution diagram for WMI and local cases.

The tool has a friendly interface that allows the user to interact with it a simple and easy way. The user interface can be seen in Fig. 4. This interface makes possible to specify the path of the project and a file .hn is created, that includes the project name and paths of the resulting file in a CSV or JSON format.

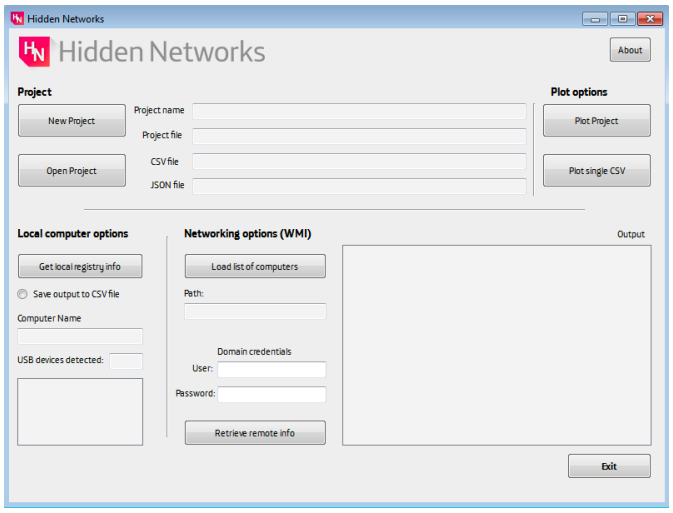


Fig. 4: User interface of the tool for the WMI and local cases

An example of CSV file is shown next:

```
computer_name,computer_ip,usbdevice_name,usbdevice_id
PC001,10.1.1.16,USB DISK 2.0 USB Device,{8bbfc3d9-29d6-58c5-be2f-dc9da53a401c}
PC001,10.1.1.16,Kingston DataTraveler G3 USB Device,{2057d6e6-7725-52d5-8d5e-3fdab3357470}
PC001,10.1.1.16,SanDisk Cruzer Blade USB Device,{1df90487-d45c-5a58-8509-dff4fae7bca6}
SRV0001,192.168.1.14,Kingston DataTraveler G3 USB Device,{2057d6e6-7725-52d5-8d5e-3fdab3357470}
SRV0001,192.168.1.14,SanDisk Cruzer Blade USB Device,{1df90487-d45c-5a58-8509-dff4fae7bca6}
SRV0001,192.168.1.14,TOSHIBA TransMemory USB Device,{53bcd3ca-866c-562f-b50b-c4f9081fa2e9}
PC002,10.1.1.15,USB DISK 2.0 USB Device,{8bbfc3d9-29d6-58c5-be2f-dc9da53a401c}
PC002,10.1.1.15,Kingston DataTraveler G3 USB Device,{2057d6e6-7725-52d5-8d5e-3fdab3357470}
PC002,10.1.1.15,SanDisk Cruzer Blade USB Device,{1df90487-d45c-5a58-8509-dff4fae7bca6}
PC002,10.1.1.15,TOSHIBA TransMemory USB Device,{53bcd3ca-866c-562f-b50b-c4f9081fa2e9}
PC005,192.168.1.30,TOSHIBA TransMemory USB Device,{53bcd3ca-866c-562f-b50b-c4f9081fa2e9}
```

About the values of the CSV file, fields “usbdevice_name” and “usbdevice_id” are collected from the registry as explained in Sec. III. Fields “computer_name” and “computer_ip”, that correspond to the computer name and its IP address, are obtained using the “socket” library in Python.

When a CSV file is opened, the tool automatically converts it to JSON format.

In the WMI mode, the tool connects with those remote computers selected, that are specified in a plain text file. An

example of such file is shown next, where the IP address or FQDN of the selected devices are written down.

```
192.168.1.14
192.168.1.29
PC004
PC005.testdomain.com
SRV001
SRV002.testdomain.com
192.168.23.12
```

After introducing credentials for the domain administrator, the information from key in the registry is read and stored. Differently, in the local mode the key is not downloaded but it is directly read from the registry, what supplies more depth in the exploration of the registry.

Results are shown in the white rectangle on the bottom right of Fig. 4. As can be seen in the graphical interface, the local analysis is located on the left and the network case appears on the right side.

If the user has information about USB connections, instead of creating a new project and collecting the information, the tool can also be used to plot such information.

For representing hidden networks, the tool draws a graph per USB device. The user can choose to either visualize all graphs or choose a specific USB.

Furthermore, our hidden networks tool offers the possibility to plot directed graphs. It indicates the order, from the oldest date to the newest one, when the USB was connected. The date the USB was inserted can be found in the file C:\Windows\inf\setupapi.dev.log [4]. An example is depicted in Fig. 5, where broader line represents an arrow.

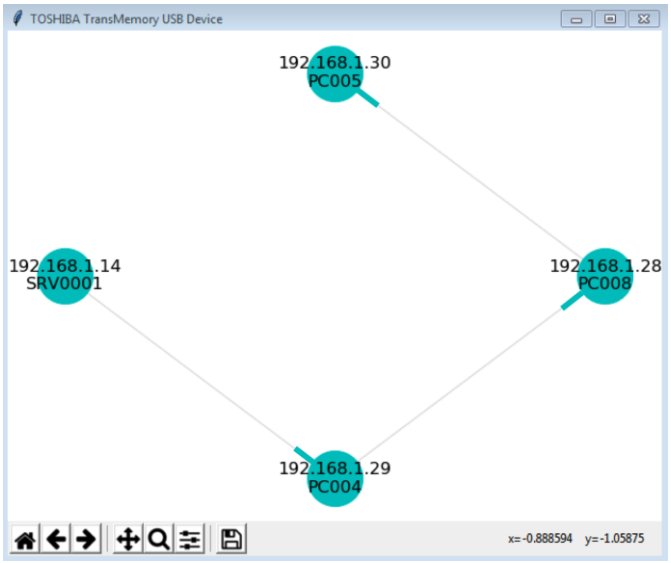


Fig. 5: Example of directed graph.

From the security point of view, it is important to mention that although our tool access the cited branch of the registry, it does not access to other files or data stored in the computer.

For interested readers, the code of our tool can be downloaded here:
https://github.com/ElevenPaths/HiddenNetworks-Python

B. WS-Management Protocol

This solution and the SMB-PSExec one are based on the following process:

- Connection to computers. Sending the script that collects information from the registry regarding USB connections of the computers connected.
- Receiving and storing information from the nodes.

Every computer selected in the network executes the scripts received and returns the output data to the central node, that collects the information reported by the different nodes of the network. Fig. 6 represents the architecture of the script execution in an Active Directory.

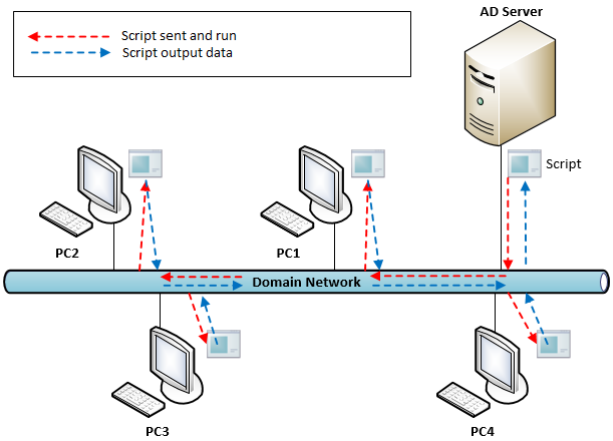


Fig. 6: Tool execution diagram for WS-Management Protocol and SMB-PsExec, using Powershell scripts.

The script WinRM version requires the activation of the Windows Remote Management (WinRM) service in each of the network computers to be audited.

Domain administrator credentials are required when executing the script to approve execution on remote computers in the local network.

The script implementation is composed of two steps:

- A *Launch* program that connects to remote computers.
- A *Recollect* program that collect information from USB devices. It is passed as parameter of the previous program to be executed by every computer selected to do it.

The execution of the "Launch" program is based on the PowerShell command "Invoke-Command". It allows to connect with a computer in the network passing the FQDN, computer name or IP address as parameters, besides executing the PoweShell script.

The outcome of the program is a CSV file called "USBDATA.csv" containing the following fields: Name of the computer, IP (on IPv4 format), USB name, ID (unique identifier). An example of this file is shown next:

PC001,192.168.1.16,Kingston DataTraveler G3 USB Device,{2057d6e6-7725-52d5-8d5e-3fdab3357470}

PC001,192.168.1.16,SanDisk Cruzer Blade USB Device,{1df90487-d45c-5a58-8509-dff4fae7bca6}

PC002,192.168.1.15,Kingston DataTraveler G3 USB Device,{2057d6e6-7725-52d5-8d5e-3fdab3357470}

PC002,192.168.1.15,SanDisk Cruzer Blade USB Device,{1df90487-d45c-5a58-8509-dff4fae7bca6}

This script *Recollect* is responsible for gathering all information referring to the USB devices connected to the computer and it runs locally in the computers to be audited.

C. SMB-PsExec

In order to run the script through SMB, it is necessary to have PSTools previously installed, specifically to execute the PSExec command in the computers to be checked.

The operating philosophy will be practically the same of the WinRM version. It will be connected from the server to the remote computer and the script should be run from the server with domain administrator account, then the USB data collection script will be executed. The computers selected to run the scripts are specified in a file called *servers.txt*. This could be done by specifying the FQDN name or IP address of the computer.

Since the connection protocol is different, the "Launch" script has a few modifications to fit with this new type of connection. Instead of using the "Invoke-Command" command, a shell from Powershell is opened and the script runs from it. The script is downloaded from the network location, preferably from a web server that would execute download through some HTTP protocol. In this way, subsequent problems with execution policy and permits, that might be found when accessing the local shared resource, are avoided.

Similar to the previous version of WinRM, results are stored in a CSV file. To avoid synchronization problems and allow time enough for the program to run on the remote computer, some delays have been included in the code. It should also be taken into consideration that the duration of the delay may vary depending on the environment where the script is run.

It is important to properly configure the location paths for each of the files before running the script, specifically, the path of the *Recollect* script, path of the *servers.txt* file and the path of the CSV file recollecting the information.

The script *Recollect* is almost the same than in the previous case. The generated *USBData.CSV* file will be exactly the same as the one previously shown.

For the implementation we have focused on Windows systems so far. It is also possible to collect information regarding USB connections in other operating systems. For example, computer systems running Mac OS X or macOS have a file with a PLIST extension, which store this information over the USB devices connected to the computer. The file is named *com.apple.finder.plist*.

VI. GRAPH REPRESENTATION

The information contained in the CSV and JSON files are plotted by the tool in a graph representation. Our tool offers the possibility to visualize the graphs independently for each USB connected, that is, for the sake of clarity the representation of the hidden links associated to each USB device are represented in separate windows.

Figure 7 shows an example of representation of the graph corresponding to the USB called "Kingston Data Traveler G3". In the figure, computers are represented as nodes and edges are represented as links between computers when the same USB have been connected to both of them, thus, edges represent the hidden links. In Fig. 7 it can be seen that the Kingston USB device was connected to four different computers (IP: 192.168.1.14, 192.168.1.16, 192.168.1.28, 192.168.1.29) that are located on the same network.

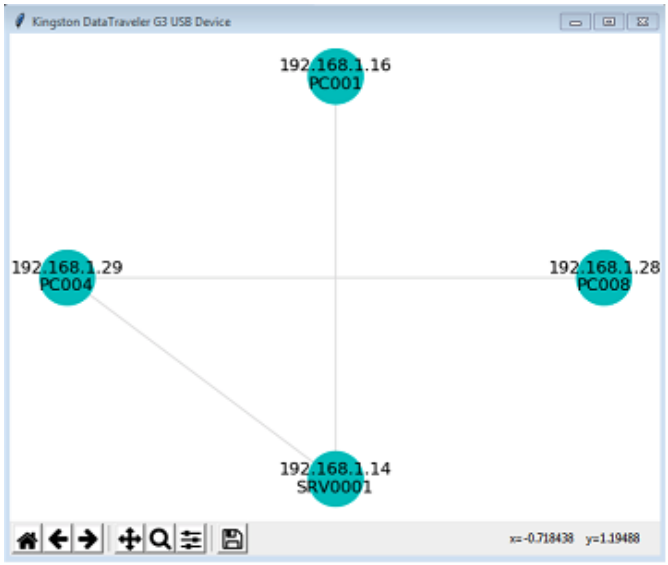


Fig. 7: Example of graph corresponding to USB "Kingston Data Traveler G3".

Graphs have been plotted using python, concretely, the NetworkX library [11] has been employed.

VII. EXPERIMENTATION

In order to test our tool, its functionalities have been tested in different environments.

In a first approach, the auditing implementations have been tested on a single-Domain network with an Active Directory (AD) to automate the information collection as much as possible. The scenario has six virtual machines, in particular five computers and a server connected to the same network. Regarding operating systems, the computers were running Windows 7 and Windows 2008 server. In this scenario, the Powershell scripts were run. As result, the server was able to connect with the computers and collect from the registry the USB information of the computers selected for that. The result obtained is shown in Fig. 8.

```
PS C:\scripts\HiddenNetworks> .\HiddenNetworks.ps1
Computer:  PC003
No USB data found
Computer:  PC004
USB found:  SanDisk Cruzer Blade USB Device
USB ID:  {1df90487-d45c-5a58-8509-dff4fae7bca6}
Computer:  PC005
USB found:  Kingston DataTraveler G3 USB Device
USB ID:  {2057d6e6-7725-52d5-8d5e-3fdab3357470}
Computer:  PC002
USB found:  Kingston DataTraveler G3 USB Device
USB ID:  {2057d6e6-7725-52d5-8d5e-3fdab3357470}
Computer:  PC002
USB found:  SanDisk Cruzer Blade USB Device
USB ID:  {1df90487-d45c-5a58-8509-dff4fae7bca6}
```

Fig. 8: USB devices connected to computers in the network.

Fig. 9 shows the result of plotting this information with our tool. There is one figure representing each USB device. PC003 is not appearing in the figures since it has no USB data.

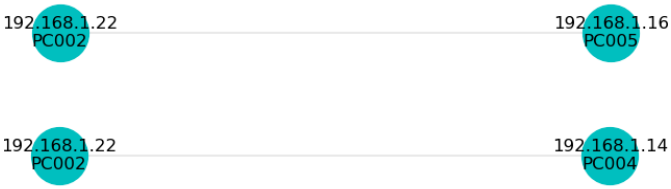


Fig. 9: Hidden networks discovered with Powershell scripts.

One remarkable aspect is the existence of connections between machines in different VLANs, that were connected via USB. This means the creation of a hidden network that connects machines that were otherwise isolated. This tool makes visible these links and helps in auditing networks and protecting them. This functionality is also very helpful in the incident response field.

Moreover, we also run other experiments with our tool and the WMI protocol. The first experiment was performed with four machines in the same network: three computers with Windows 7 and one with Windows 2008 server. The result plotted by the tool can be seen in Fig. 10.

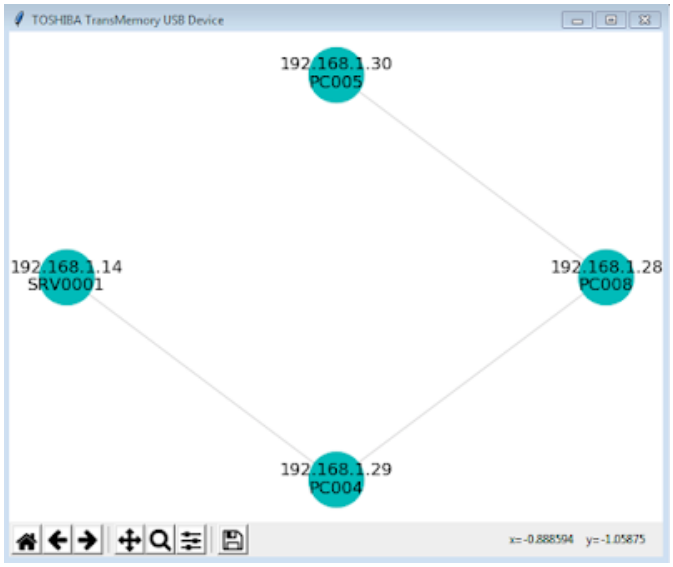


Fig. 10: Hidden networks discovered with our tool for computers in the same network. Graph corresponding to the USB "Toshiba TransMemory"

Another experiment was carried out with our tool. In this case we used four machines: two computers with Windows 7, one computer with Windows 10 and a server with Windows 2008 server. The difference with the previous experiment is that, in this case computers belonged to different subnetworks. The application is run from a machine that has visibility to both subnets. The result plotted by the tool for different USB devices can be seen in Fig. 11.

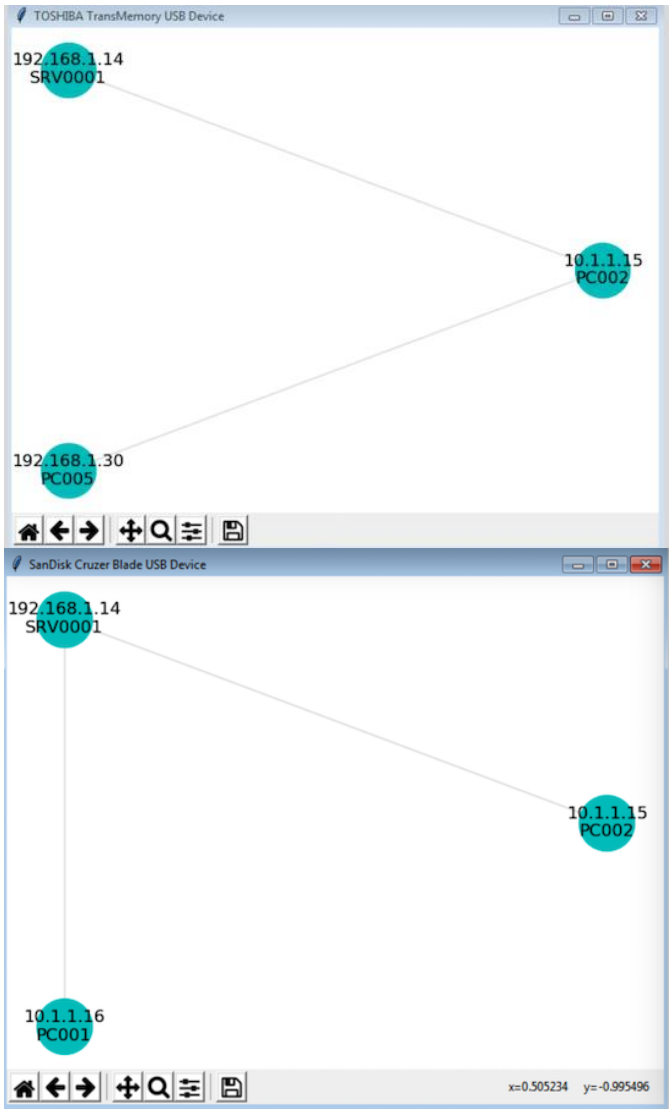


Fig. 11: Hidden networks discovered with our tool for computers in different subnetworks. Graph corresponding to the USB "Toshiba TransMemory" and "SanDisk Cruzer Blade"

As can be seen in the figure, PC001 and PC002 belong to a subnetwork and SRV001 and PC005 belong to a different one. Even though, the graphic shows that it is possible to communicate computers in different subnetworks via USB. Without our tool these connections could go unnoticed, since those computers were apparently not connected.

VIII. LIMITATIONS

Our tool relies on the information stored on the registry. In case that for any reason (malware, an error, etc.) the registry would have been altered and it does not contain the real information, this would not be detected by our tool, since it has been designed for forensic purposes and not for detection ones. In any case, the probabilities of this event are not high,

for example not any malware could make this effect, but only those that are able to get administrator privileges.

IX. MITIGATION

One way to prevent hidden networks is to control the use of USB devices in computers. Mitigation or prevention can be done through the forced use due to Active Directory policies, which restrict connection in a user computer to devices only approved by one user. The implementation of the safety policy along with a white list of approved devices for each user allows avoiding this kind of hidden links, but it is complex and expensive to maintain.

X. CONCLUSIONS AND FUTURE WORK

Having networks disconnected from the Internet provides this false security feeling regarding a higher level of protection before any incident, which increases the system vulnerability. The USB is an attack vector that frequently goes unnoticed, however, the malware infection through USB devices is a real and underlying problem. USB are also potentially dangerous for cases of data exfiltration and leakage. Furthermore, USB can connect computers that are apparently isolated physically or logically. Since being aware of such connections is essential for guaranteeing the security of our computers and networks, in this paper we present a solution able to automatically discover hidden links and visualize them. It can be done both remotely and locally. Furthermore, we have included several mechanisms for connecting with nodes and extracting information about USB connected to them. On the one hand, we have implemented Powershell scripts that connect through WinRM and SMB-PsExec. On the other hand, we have implemented a tool in Python able to connect using the WMI protocol and that also covers the local case. We have conducted several experiments, with computers with diverse operating systems, in different VLANs and different subnets and the result of all of them probe that it is possible to connect through USB computers that are apparently isolated. This is a main issue in network and computer security, therefore, we would like to raise awareness about that. Moreover, our tools are not only able to discover this hidden links but can be very helpful in cases of incident response, for example, if a data exfiltration case takes place, it could help in the forensic tasks of tracing a certain USB within a network or informing about the USB connections that were done in a particular computer. We would like to stress that the results of this paper are not restricted to USB memories, but it can also be applied to other devices connected through USB, such as external hard disks, Wi-Fi or Bluetooth dongle, etc. The aim of this solution is to reinforce computer and network security by helping in the prevention of incidents, facilitating audits and providing utilities useful for forensic analysis cases.

For future work, different colours can be used in the directed graph to indicate whether the USB device was introduced the same day, the week and so on. Additionally, the granularity of the tool capabilities could be finer, so that not only computers involved in the hidden network can be traced but also the files involved.

ACKOWLEDGEMENT

This is a work developed by the CDO unit of Telefonica.

REFERENCES

[1] S. Karnouskos, "Stuxnet worm impact on industrial cyber-physical system security," in *IECON 2011 - 37th Annual Conference of the IEEE Industrial Electronics Society*, Melbourne, VIC, pp. 4490-4494, 2011.

[2] Microsoft Computer Dictionary. Windows Registry. https://support.microsoft.com/en-us/help/256986/windows-registry-information-for-advanced-users. Last Updated: Jan 7, 2017.

[3] Nir Nissim, Ran Yahalom, Yuval Elovici, "USB-based attacks", in *Computers & Security*, vol. 70, pp. 675-688, 2017.

[4] Abhijeet Ramani, Somesh Kumar Dewangan: "Auditing Windows 7 Registry Keys to track the traces left out in copying files from system to external USB Device" in *International Journal of Computer Science and Information Technologies*, vol. 5 ,2, pp.1045-1052, 2014.

[5] Microsoft, "WS Management Protocol", 2018. https://msdn.microsoft.com/en-us/library/aa384470(v=vs.85).aspx

[6] Microsoft, "Windows Management Instrumentation", 2018. https://msdn.microsoft.com/en-us/library/aa394582(v=vs.85).aspx

[7] Microsoft, "Windows Remote Management", 2018. https://msdn.microsoft.com/en-us/library/aa384426(v=vs.85).aspx

[8] Microsoft, "PsExec v2.2", 2016. https://docs.microsoft.com/en-us/sysinternals/downloads/psexec

[9] Richard Sharpe, "Just What is SMB?", 2002. https://docentes.uaa.mx/guido/wp-ontent/uploads/sites/2/2015/04/What-is-SMB_.pdf

[10] Don Box, David Ehnebuske, Gopal Kakivaya, Andrew Layman, Noah Mendelsohn, Henrik Frystyk Nielsen, Satish Thatte, Dave Winer, "Simple Object Access Protocol (SOAP) 1.1", 2000. https://www.researchgate.net/profile/Satish_Thatte/publication/239553871_Simple_object_access_protocol_SOAP_11/links/54489e4e0cf2f14fb8142a59/Simple-object-access-protocol-SOAP-11.pdf

[11] A Hagberg, P Swart, DS Chult, "Exploring network structure, dynamics, and function using NetworkX", in *SCIPY*, 2008. http://permalink.lanl.gov/object/tr?what=info:lanl-repo/lareport/LA-UR-08-05495

Detección de vulnerabilidades basadas en Stack Overflow mediante DBI

Javier Carrillo Mondéjar
Instituto de Investigación en Informática de Albacete (I3A)
Universidad de Castilla-La Mancha (UCLM)
javier.carrillo@uclm.es

José Luis Martínez Martínez
Instituto de Investigación en Informática de Albacete (I3A)
Universidad de Castilla-La Mancha (UCLM)
joseluis.martinez@uclm.es

***Resumen*—A pesar de todos los esfuerzos realizados por la comunidad científica en materia de seguridad informática, las vulnerabilidades de tipo *buffer overflow* siguen siendo el mayor fallo de seguridad de las aplicaciones, puesto que comprometen la seguridad del sistema mediante la corrupción de memoria. Para atacar este problema, existen diferentes técnicas basadas en el análisis del binario de la aplicación en cuestión. Con este objetivo, el presente artículo propone un algoritmo basado en la instrumentación dinámica que realiza un enfoque híbrido; en una primera etapa, un análisis estático para obtener información sobre las funciones y, posteriormente y de manera dinámica, se comprueban las variables que pertenecen a cada una de las funciones, detectando si se produce desbordamiento de memoria entre ellas. Los resultados obtenidos muestran como el algoritmo propuesto es capaz de detectar errores del tipo *buffer overflow* en los marcos de pila de una función.**

***Index Terms*—Bugs, *buffer overflow*, Vulnerabilidades, Instrumentación Dinámica**

Tipo de contribución: *Investigación original*

I. Introducción

Hoy en día, un producto software es un gran proyecto de ingeniería que está formado por un gran número de componentes y librerías interconectados, con millones de líneas de código en cada uno de ellos. El desarrollo de este tipo de productos conlleva fallos o *bugs* que pueden suponer serios incidentes de seguridad donde el software sea desplegado.

Un *buffer overflow* o desbordamiento de memoria es una de las vulnerabilidades más peligrosas y, a la vez, más frecuentes en los desarrollos software. Ocurre durante la ejecución de un programa cuando una aplicación escribe datos fuera de los límites que tiene establecidos. Estos datos sobrescriben posiciones adyacentes de memoria y, dependiendo de cómo se haga, puede afectar al comportamiento del programa. Un *buffer overflow* puede producir un ataque de Denegación de Servicio (DoS), si el programa sobreescribe datos necesarios para el correcto funcionamiento. También, un *buffer overflow* puede producir una Ejecución Remota de Código (RCE) no intencionado si, mediante la escritura (o inyección de código), es capaz de robar el flujo de ejecución, y ejecutar una *shellcode* [1]. Estos ataques son cada vez más sofisticados y son capaces de realizar tareas muy diversas que pueden ir desde la obtención de una *shell*, robo de información sensible en el sistema comprometido, elevación de privilegios, hasta la descarga de *malware* y su posterior propagación dentro de la red. Esta situación se agrava cuando ese software se despliega en infraestructuras críticas.

La falta de operaciones que comprueben los límites de los *buffers* permiten este tipo de errores. Las aplicaciones escritas en lenguajes C o C++ son comúnmente asociadas a este tipo de vulnerabilidades, ya que éstos permiten sobrescribir parte de la memoria sin comprobar si el dato escrito se produce en posiciones no reservadas. Existen dos tipos bien diferenciados de vulnerabilidades *buffer overflow*, las de tipo pila o *stack* y las basadas en el *heap* o durante la reserva dinámica de memoria. Veremos más detalles acerca de estos tipos en la Sección II. La comunidad científica ha realizado un gran esfuerzo en mitigar y neutralizar este tipo de vulnerabilidades. Las diferentes técnicas incluyen análisis manuales y automáticos. Los analistas pueden optar por depurar el programa paso a paso introduciendo *breakpoints*, buscando este tipo de vulnerabilidades. Este trabajo es ineficiente y tedioso e, incluso, las aplicaciones pueden incluir técnicas para detectar el análisis de las mismas. Por otro lado, las técnicas automáticas pueden buscar este tipo de vulnerabilidades a partir del código fuente, aunque éste no siempre está disponible. Otra solución de protección pasa por habilitar diferentes opciones en el compilador. Por ejemplo, la protección *No eXecute* (NX) en *AMD* y *eXecute Disable* (XD) de *Intel*, trata de evitar que se ejecute código en zonas de memoria destinadas para datos, como la pila. Otra técnica es la denominada *Address Space Layout Randomization* (ASLR), cuya función es aleatorizar el espacio de direcciones virtuales de un proceso con el fin de evitar *exploits* que utilizan direcciones estáticas, dificultando así la ejecución de técnicas de explotación como la Programación Orientada al Retorno (ROP). Finalmente, las *canary cookies* implica añadir un valor en la pila al inicio de cada una de las funciones que contenga variables que son susceptibles a un desbordamiento; se comprueba si ese valor aleatorio está inalterado. En caso de que se haya modificado, el programa detendrá su ejecución ya que se ha sobrescrito intencionadamente.

En este artículo, se propone un algoritmo escalable y transparente para detectar vulnerabilidades de tipo *buffer overflow*. El enfoque del algoritmo es híbrido, ya que parte, primeramente, de un análisis estático para detectar sus funciones, su dirección de inicio y su tamaño en memoria para poder reconocer en tiempo de ejecución a que función pertenece cada instrucción. Posteriormente, un proceso dinámico, basado en *Dynamic Binary Instrumentation* (DBI) o instrumentación dinámica [2], detecta las variables de las funciones ejecutadas y los fallos de escritura fuera de los espacios reservados. Los experimentos realizados confirman la viabilidad del algoritmo propuesto, detectando las variables de cada marco de pila y los desbordamientos que se producen entre éstas. Con todo ello, se consigue la detección de desbordamientos que no lleguen a sobrescribir la dirección de retorno almacenada en la pila.

La organización del documento es la siguiente: la Sección II recoge los conceptos teóricos relativos a las vulnerabilidades *buffer overflow* y conceptos sobre arquitectura de computadores. La Sección III presenta el estado del arte relacionado con el problema abordado. El algoritmo propuesto se describe en detalle en la Sección IV y éste es evaluado en la Sección V. Finalmente, las conclusiones pertinentes son mostradas en la Sección VI.

II. Fundamentos técnicos

En esta sección se va a tratar los temas relativos a la arquitectura de computadores en la Sección II-A, los diferentes tipos de vulnerabilidades *buffer overflow* en el apartado II-B y, finalmente, la Sección II-C describirá el funcionamniento de la instrumentación dinámica de ejecutables.

II-A. Arquitectura x86

Inicialmente, cuando se ejecuta un programa, el cargador del sistema operativo lee la información necesaria de las cabeceras del archivo ejecutable. Una vez cargado aparecen 4 secciones: datos, instrucciones, *heap* y pila. Vamos a hacer hincapié en estas dos últimas: el *heap* se utiliza para asignar dinámicamente bloques de memoria en tiempo de ejecución, mientras que la pila se emplea en el almacenamiento de variables locales y para el paso de parámetros a las funciones que son invocadas por el programa. Físicamente, la pila es un área que ha sido definida para este propósito, y su diferencia con cualquier otro dato es sólo una diferencia lógica. Internamente, la pila es manejada como una estructura de datos *Last In, First Out* (LIFO) y su memoria se asigna de manera descendente. En la pila se almacena la dirección de retorno mediante la obtención del valor actual del registro *Instruction Pointer Register* (EIP). Este valor es importante puesto que la mayoría de ataques intentar sobreescribir su posición de memoria, con el objetivo de redirigir el flujo de ejecución hacia zonas de memoria controladas por el usuario malicioso o partes del programa no visibles por el usuario final [3].

II-B. Vulnerabilidades de corrupción de memoria

Los desbordamientos de memoria o *buffer overflow*, son un fallo de seguridad que consisten en la escritura de datos más allá del tamaño que fue reservado para una variable en memoria. Se producen debido a la falta o incorrecta comprobación de los límites del *buffer* y pueden provocar que un atacante consiga el control de un sistema afectado.

II-B1. Desbordamiento basado en pila: Se produce en el área de memoria reservada para la pila y es el tipo de desbordamiento de *buffer* más sencillo de explotar. La explotación de este fallo de seguridad puede llegar a sobrescribir variables almacenadas en la pila o la dirección de retorno de la función afectada, pudiendo provocar un cambio en el flujo de ejecución de un programa.

II-B2. Desbordamiento basado en el heap: A diferencia de la pila, en la que el programador se abstrae de la asignación y liberación de la memoria (es realizada por el compilador), en el *heap* es necesario que el programador realice llamadas a funciones del sistema operativo para asignar o liberar dicho espacio. Cuando se realiza una petición de memoria, el gestor del *heap* asignará un bloque de memoria en el área del *heap* y devolverá un puntero a esa porción de memoria que está disponible. Por tanto, debe existir una estructura de datos para la gestión de memoria y un algoritmo de asignación. Debido a ello, la explotación de este tipo de vulnerabilidades no es tan trivial como ocurría en la pila. Un *heap overflow* podría permitir a un atacante modificar datos de los objetos instanciados en el *heap* e incluso la dirección de su *vtable*, donde se encuentran las direcciones de los métodos virtuales de dicho objeto y, de esta manera, se podría cambiar el flujo de ejecución de un programa hacia una zona de memoria controlada.

II-B3. Fallos en la asignación y liberación de memoria: Entre este tipo de fallos, se encuentra el error *Use After Free* (UaF), que se produce cuando se utiliza un puntero que ya ha sido liberado. Suelen darse por errores en la lógica del programa y, por tanto, resultan complicados de detectar mediante un análisis estático. Una vez que se ha asignado la memoria, las funciones encargadas de la liberación de la misma se encargan de desasignar las porciones de memoria correspondientes. Tras la liberación, lo que realmente ocurre es que se marcan como disponibles dichas porciones para un uso posterior, pero el puntero sigue existiendo y apunta a dicha zona de memoria. Por otro lado, existen los fallos del tipo *Double Free* (DF), los cuales se producen cuando se intenta liberar un puntero que ya ha sido liberado previamente y, aunque en este tipo de errores es muy complicado conseguir la ejecución de código arbitrario, pueden producir que el proceso termine su ejecución de manera inesperada.

II-C. Instrumentación de ejecutables

La instrumentación consiste en insertar código en una aplicación con el fin de analizar su comportamiento. Se puede introducir de manera estática en tiempo de compilación, por lo que es necesario disponer del código fuente de la aplicación. También es posible realizar la instrumentación dinámicamente, mediante la inserción de código en tiempo de ejecución. La principal ventaja de la instrumentación dinámica es que permite controlar la ejecución de un programa y no es necesario disponer de su código fuente. Este aspecto es muy importante, por ejemplo, en el análisis de *malware* y búsqueda de vulnerabilidades ya que, en la mayoría de los casos, sólo se dispone del ejecutable de la aplicación [4].

Existen distintos sistemas de instrumentación dinámica de ejecutables (DBI) que ofrecen una interfaz de programación de aplicaciones para crear nuestras propias herramientas, como por ejemplo, *PIN* [5], *Valgrind* [6], *Frida* [7] o *DynamoRIO* [8].

III. Estado del arte

Hoy en día existen diversos usos de DBI aplicados en el ámbito de la seguridad. DBI puede ser aplicado para el análisis de *malware* ya que permite seguir de manera transparente las llamadas a librerías y funciones del sistema. En [9] se presenta un clasificador de software malicioso basado en la creación de un grafo a partir del comportamiento de una muestra en tiempo de ejecución. Propone una técnica para la detección de familias de *malware* usando estos grafos. Para ello, emplea varias técnicas de distancia de edición de grafos que miden su similaridad. Las distancias calculadas

se utilizan para clasificar nuevas muestras utilizando algoritmos de aprendizaje automático. Además, también puede ser utilizado para desempaquetar *malware* de manera automática detectando accesos de escritura en regiones de memoria que son ejecutadas posteriormente [10]. En el trabajo [11] se propone el uso de DBI para evitar las medidas de protección incorporadas por el *malware* para evitar su análisis y ejecución en entornos controlados.

Otras propuestas de uso de DBI se centran en la detección de ataques, como en [12], donde se propone una herramienta para detectar ataques ROP en tiempo de ejecución creando una pila alternativa de manera que cada vez que se ejecute una instrucción de retorno se pueda comprobar si coincide con la que tiene almacenada en la pila alternativa. En caso contrario, la dirección de retorno en la pila ha sido sobreescrita, por lo que se ha producido un ataque.

En [13] se describe una herramienta utilizando DBI para el análisis automático de código inyectado. Su propuesta se basa en la detección de ejecución de código en zonas de memoria diferentes a los módulos que han sido cargados en memoria, es decir, si por ejemplo la dirección de memoria pertenece a un área de la pila o del *heap*. Como en el trabajo anterior, también hace uso de una pila alternativa para detectar ataques de tipo ROP.

En [14] se implementa una técnica para evitar el uso de fuerza bruta en las *canary cookies* en programas que hacen uso de la función *fork*. Cuando se crea un proceso hijo, éste hereda todo el espacio de direcciones del proceso padre, por lo que hereda también el valor de la *canary cookie* y, por tanto, un atacante podría aprovechar para realizar fuerza bruta y evitar así este mecanismo de protección. Su propuesta se basa en una *pintool* que modifique el valor de la *canary cookie* del proceso hijo y de todos los marcos de pila que hereda del proceso padre.

Para la búsqueda de vulnerabilidades, [15] propone el uso de DBI para realizar un análisis del flujo de datos de un programa para determinar que variables son contaminadas. Posteriormente utiliza esta información para dirigir el proceso de ejecución simbólica y encontrar posibles vulnerabilidades en la aplicación.

En [16] se plantea una herramienta que se divide en tres partes retro-alimentadas entre ellas. La primera, se encarga de la monitorización y almacenamiento del contexto de las instrucciones ejecutadas por el programa. La segunda, se encarga de utilizar ese fichero de trazas y de reproducir la ejecución para analizar las variables contaminadas creando un archivo de índices para consultas rápidas. Finalmente, la última etapa analiza el resultado proporcionado en busca de fallos de seguridad basándose en un serie de reglas predefinidas para detectar errores de escritura y de asignación de memoria.

En [17] describe una herramienta basada en DBI para identificar desbordamientos en los marcos de pila de un programa. En su estudio, expone dos formas para identificar los desbordamientos. La primera consiste en verificar los accesos a la pila frente a los limites de las variables locales de esa función. La segunda, se centra en detectar cuando la dirección de retorno ha sido sobreescrita. Finalmente, su implementación se basa en esta segunda forma debido a la dificultad de determinar los limites de inicio y fin de las variables por la falta de tipos y otras semánticas a nivel de código máquina.

Para la detección de errores en el área del *heap*, [4] y [3] se encarga de instrumentar las funciones relacionadas con la gestión del *heap* añadiendo marcas antes y después de cada bloque de memoria asignado para comprobar si alguna lectura o escritura de memoria se realiza en esas zonas de memoria. Su principal diferencia radica en que [4] es completamente transparente a la aplicación instrumentada, ya que crea un mapa de memoria alternativo para mapear los bloques de memoria asignados en *heap*.

En [18] se plantea una herramienta para la detección de errores de corrupción de memoria tanto en el *heap* como en la pila. La desventaja de esta herramienta es que, al utilizar instrumentación a nivel de compilador, es necesario disponer del código fuente, por lo que no puede ser utilizada para instrumentar aplicaciones de código cerrado.

En general, existen diversos campos de la seguridad informática donde se aplica el uso de la instrumentación dinámica. En cuanto a la detección de vulnerabilidades de memoria, la mayoría de ellos se centran en detectar problemas en el *heap* y ataques de tipo ROP o inyección de código. La propuesta planteada en este artículo se basa en la detección de desbordamientos en el área de la pila y la detección de las variables y sus tamaños en el marco de pila de las funciones que están siendo ejecutadas.

IV. Propuesta

Aunque el concepto de *buffer overflow* se produce de manera similar tanto en el *heap* como en el *stack*, es decir, por una mala o inexistente comprobación de los límites de un *buffer*, las técnicas para diagnosticarlo son distintas. En el *heap*, es posible detectar la vulnerabilidad entre los distintos bloques de memoria asignados mediante DBI, ya que se puede obtener en tiempo de ejecución la dirección y tamaño de los bloques de memoria asignados instrumentando las funciones que son utilizadas para la gestión de la memoria dinámica y, por tanto, almacenar los bloques que están siendo utilizados y aquellos que han sido liberados. Finalmente, cuando se produce una escritura en memoria, se puede comprobar si ésta se produce dentro de los límites del bloque asignado. En caso de que se encuentre fuera del área, se trata de un problema de *heap overflow*. También, al almacenar los bloques de memoria que han sido liberados, es posible diagnosticar errores del tipo UaF, en caso de que se produzca un acceso de lectura o escritura en un bloque ya liberado, y del tipo DF si se libera un bloque que había sido liberado previamente.

En cuanto a la detección de *buffer overflow* en el stack, es más complicado de detectar ya que pueden existir varias variables en el *stack* y es necesario conocer su dirección de memoria. Aunque como vimos en la Sección III existen otros enfoques diferentes en la literatura que basan la detección de este tipo de errores comprobando si la dirección de retorno almacenada en la pila ha sido sobreescrita. Este tipo de enfoques no detectaría el problema cuando haya un desbordamiento entre variables de un mismo *stack frame*.

Para resolver la problemática asociada a la detección del *stack overflow*, es imprescindible aislar las funciones y averiguar el número de variables que contiene cada una de

ellas, es decir, es necesario saber a que función pertenece la instrucción que se ejecuta. Este problema puede ser resuelto mediante análisis estático, obteniendo el inicio y tamaño de cada una de las funciones del programa. Después, en tiempo de ejecución es posible calcular, mediante *Ec. (1)*, la dirección de memoria y soportar así, aquellas aplicaciones que tengan ASLR incorporado en sus módulos.

$$\begin{aligned} offset &= address - imageBase \\ newAddress &= offset + newImageBase \end{aligned} \quad (1)$$

Posteriormente, se crea un *stack frame* en cada instrucción del tipo *CALL* y se almacenan en éste las variables detectadas que pertenecen a esa función. Para tal efecto, conociendo que, a bajo nivel, las variables locales de una función son referenciadas a través del registro EBP o ESP y, por tanto, cada vez que se produzca una instrucción de carga efectiva (LEA) o una instrucción de escritura donde el destino final venga referenciado por uno de estos dos registros, es posible almacenar la dirección de memoria en el *stack frame* creado previamente, y así actualizar el tamaño de estas variables. La Figura 1 muestra un ejemplo de las variables que pueden ser detectadas en tiempo de ejecución. En esta implementación se han obviado los argumentos, ya que para que éstos sean sobreescritos en el *stack frame* actual, habría que desbordar las direcciones anteriores como la dirección de retorno. Además, en caso de que sea un puntero y obtenga su dirección efectiva, esta dirección pertenecerá a una variable de otro marco de pila.

Figura 1. Ejemplo de variables detectadas.

```
mov eax, __security_cookie
xor eax, ebp
mov [ebp-0x1C], eax ; 1
mov eax, [ebp+0x8]
lea edi, [ebp-0x124]; 2
xor ebx, ebx
mov [ebp-0x130], ebx; 3
or [ebp+0x12], 0xFFFFFFFF
```

Puesto que se detectan variables de manera dinámica, éstas pueden aparecer en cualquier punto de la función y podría producirse un desbordamiento de una variable antes de haber detectado todas las que pertenecen a ese *stack frame* y por tanto, no detectar el desbordamiento. Para solucionar este aspecto, es necesario almacenar el número de bytes escritos en cada una de las variables para comprobar antes de la destrucción del *stack frame* si alguna de esas escrituras ha sobreescrito a la variable consecutiva. El tamaño escrito en cada variable puede conseguirse detectando escrituras consecutivas en memoria mediante el Algoritmo 1.

El procedimiento a realizar comprende analizar aquellas instrucciones de escritura en memoria y comprobar si las direcciones de memoria son adyacentes y si se trata de la misma instrucción de escritura en memoria ejecutada anteriormente. En caso afirmativo, se aumenta el número de bytes escritos. Por el contrario, se añade el tamaño escrito en esa variable. Gracias a ello, es posible comprobar si alguna de esas escrituras produce un desbordamiento en la variable permitiendo detectar éstos en tiempo de ejecución. La comprobación se realizará cada vez que un *stack frame* sea destruido, es decir, cuando se produce una instrucción de tipo RET. También, esta comprobación se realizará si se produce algún tipo de excepción en la aplicación que esta siendo analizada.

Algorithm 1 Detección de copia de datos

Require: insAddress, memAddr, sizeWritten, threadid
1: Recuperar UltimaIns ejecutada, ContadorBytes y StackFrames activos.
2: **if** ultimaDireccion de memoria es cero **then**
3: $UltimaDireccion \leftarrow memAddr$
4: $ContadorBytes \leftarrow sizeWritten$
5: **end if**
6: **if** UltimaIns es cero **then**
7: $UltimaIns \leftarrow insAddr$
8: **end if**
9: **if** Mem es adyacente a UltimaDireccion y insAddr es igual a UltimaInst **then**
10: $contadorbytes \leftarrow contadorbytes + sizeWritten$
11: **end if**
12: **if** insAddr no es igual a la ultima instruccion de escritura **then**
13: $FrameID \leftarrow BuscaStackFrame(UltimaDireccion)$
14: $Var \leftarrow BuscaVariable(FrameID)$
15: $InsertaDireccionMemoriayContadorBytes$
16: $ContadorBytes \leftarrow sizeWritten$
17: $UltimaDireccion \leftarrow memAddr$
18: $UltimaInstruccion \leftarrow insAddr$
19: **end if**

V. Efectividad y rendimiento de la propuesta

En esta sección, se ofrece una evaluación experimental de la propuesta descrita en la Sección IV. Para las pruebas realizadas, se ha desplegado una máquina virtual equipada con Windows 7 SP1 x86, 3GB de RAM y Intel Core i7-2670QM @ 2.20 GHz.

V-A. Evaluación de aplicaciones

Para comprobar el funcionamiento de la propuesta presentada, se ha desarrollado una herramienta utilizando el framework PIN. PIN [5] es una herramienta creada por Intel que permite instrumentar aplicaciones de manera transparente a la aplicación. La Figura 2 muestra un esquema de su arquitectura. Se puede observar que está compuesto principalmente por dos componentes: una máquina virtual que contiene el compilador JIT y la unidad de emulación, y una caché de código que se utiliza para almacenar el código ya instrumentado, reduciendo así la sobrecarga [19].

Aunque soporta diferentes tipos de granularidad a la hora de realizar la instrumentación, en nuestro caso se realiza la instrumentación a nivel de imagen y código máquina. Las herramientas creadas con PIN se denominan *pintools* y permiten configurar el comportamiento de éste. La *pintool* implementada recibe un fichero de texto en formato *json* con cada una de las funciones a instrumentar, así como su dirección, tamaño y base del módulo. Esto puede conseguirse de manera automática haciendo uso de las interfaces que

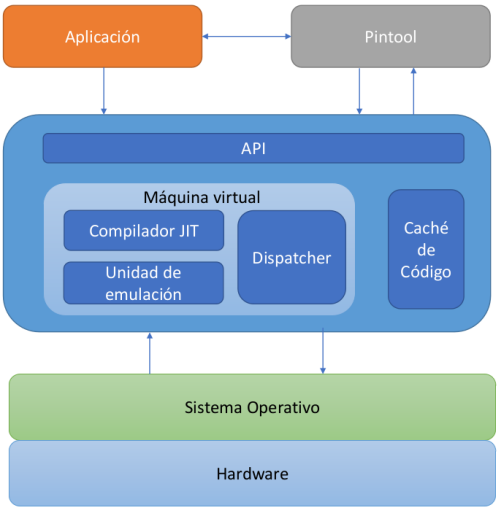


Figura 2. Esquema arquitectura PIN.

proporcionan *radare2* [20] o *IDA* [21] para la creación de *scripts*. En nuestro caso, se ha decido hacer uso de *radare2*, ya que se trata de una herramienta de código libre. El *script* desarrollado utiliza *radare2* para obtener información de todas las funciones del archivo ejecutable pasado como parámetro. Finalmente, se crea un archivo *json* con toda esa información, el cual será la entrada a la herramienta desarrollada, tal y como se comentó anteriormente.

Cada vez que se carga una imagen en la aplicación instrumentada se obtendrá su dirección de carga y se recalcularán las direcciones de todas las funciones de esa imagen proporcionadas en el archivo *json*.

Aunque la *pintool* diseñada soporta la instrumentación de cualquier módulo, en los experimentos realizados sólo se ha considerado instrumentar el módulo del ejecutable principal, obviando las librerías asociadas y DLLs del sistema, reduciendo así la sobrecarga. Se han seleccionado aplicaciones que presentan una vulnerabilidad del tipo *stack overflow* en su modulo principal y cuyos *exploits* se encuentran disponibles públicamente en *exploit-db* [22].

Las aplicaciones seleccionadas han sido ejecutadas con la *pintool* y, posteriormente, se ha ejecutado el *exploit* que desencadena la vulnerabilidad. Los resultados arrojados por la herramienta han sido satisfactorios mostrando con precisión la instrucción donde se produce la copia de datos que provoca el fallo de tipo *stack overflow* y el *stack frame* de la función a la que pertenece la variable que ha sido desbordada.

Las aplicaciones que han sido testeadas con la herramienta desarrollada son las siguientes:

- *ASX2MP3Converter*. Se trata de una aplicación que convierte archivos con formato *asx* a formato *mp3* y puede ser explotada de manera local [23]. La *pintool* detecta que se ha producido un desbordamiento en la instrucción situada en 0x42b820 y el desbordamiento se produce en el *stack frame* perteneciente a la función 0x42b420.
- *PCMANFTPD2*. Se trata de un servidor FTP que puede ser explotado de manera remota a través del comando *mkd* [24]. El desbordamiento se produce en la instrucción 0x4173ad y desborda una variable perteneciente al *stack frame* de la función 0x403e60.
- *UltraMiniHTTPd*. Consiste en un servidor HTTP que se puede atacar de manera remota a través de una petición *POST* [25]. El desbordamiento se produce en la instrucción 0x406a48 y desborda una variable que pertenece al área de pila de la función 0x402660.
- *CoreFTPServer*. Mediante el comando *TYPE*, este servidor FTP puede ser explotado, provocando una denegación de servicio en el servidor. El desbordamiento se produce en la instrucción 0x4de56d y desborda la variable perteneciente al *stack frame* de la función 0x439140. La variable sobrescrita contiene un puntero a un objeto y al acceder a su posición de memoria provoca el fallo en la aplicación.

```
STACK OVERFLOW
Se ha producido en la instrucción 0x42b820
STACK FRAME DE LA FUNCION 0x42b420
0x131b10
0x131b14
0x131b18
0x131b20
0x131b24
0x131b30
0x133d30
0x135f30
0x138130
0x138134 <---------------- 26060 bytes en 0x138134
0x13a338
0x13c544
0x13c560
```

Figura 3. Pintool: Detección *stack overflow*.

La Figura 3 muestra un extracto del archivo de salida generado, donde se puede observar que se arroja la información de la instrucción que produjo el desbordamiento, así como las variables pertenecientes al *stack frame* y el tamaño en *bytes* escrito en esas variables. Además, es importante destacar que, aunque detecta correctamente el desbordamiento de memoria, también genera algún falso positivo. Por ejemplo, en la aplicación *PCMANFTPD2*, indica que se ha producido un desbordamiento entre las distintas variables de la función 0x0041dde9. La *pintool* ha conseguido detectar y mostrar las variables involucradas en el desbordamiento, pero realmente son campos de una estructura que se rellena posteriormente en una instrucción de tipo *REP MOVSD*. Si observamos el desensamblado en *radare2* de esta función también nos indica que existen siete variables en ese marco de función, mientras que IDA reconoce la función a la que se le pasa como argumento el puntero de esa estructura y, por tanto, reconoce el tipo y tamaño de la variable de ese *stack frame*. Por tanto, aunque la detección de las direcciones de cada campo de la estructura es correcta, se trata de un falso positivo puesto que es un caso de memoria controlada.

V-B. Análisis de sobrecarga de la instrumentación

Para medir la sobrecarga que provoca la instrumentación dinámica, se ha utilizado el mismo conjunto de aplicaciones testeadas anteriormente. Puesto que se trata de aplicaciones con interfaz gráfica en las que es necesaria la intervención del usuario o de un evento para continuar, éstas han sido modificadas de manera que el tiempo de ejecución no se vea afectado por esta interacción. Para ello, se han parcheado las aplicaciones para que finalicen su ejecución llamando

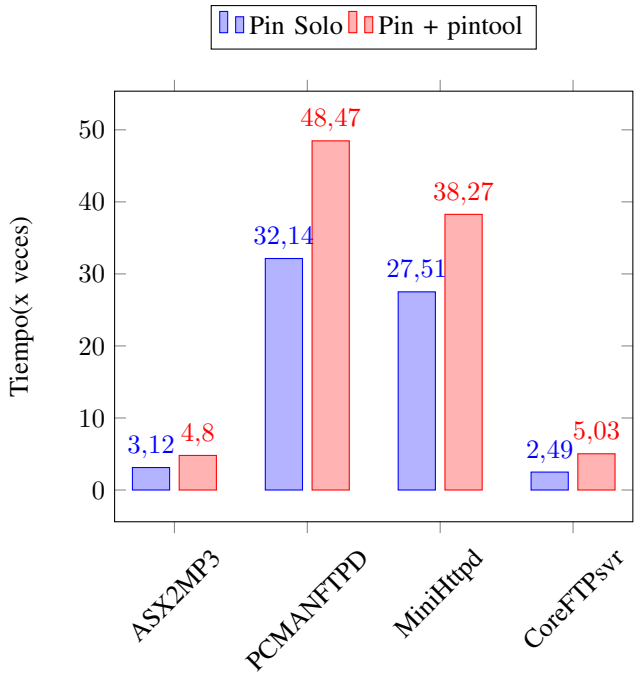


Figura 4. Sobrecarga provocada por DBI

a la función ExitProcess una vez que se muestra la GUI del programa o al esperar una conexión entrante en el caso de las aplicaciones de tipo servidor. Además, para medir la sobrecarga que provoca la pintool, se ha medido también la carga que conlleva ejecutar a través del framework de PIN sin instrumentar ninguna instrucción. La Figura 4 muestra los resultados obtenidos en las pruebas realizadas para medir la sobrecarga de DBI.

En función de los resultados logrados, es evidente que DBI introduce una sobrecarga importante en el tiempo de ejecución de las aplicaciones, especialmente en el caso de PCMAN-FTPD2 y UltraMiniHTTPd. Esta sobrecarga introducida viene dada por el código de instrumentación insertado para detectar las variables del *stack frame* y por la instrumentación de todos los accesos de escritura en memoria que se dan en el módulo principal, partes necesarias para la detección de errores del tipo *buffer overflow*. También, cabe destacar que el framework PIN de Intel introduce una sobrecarga independiente de la inserción código de instrumentación en la aplicación.

VI. Conclusiones

En este artículo se ha presentado un algoritmo original para la detección de variables y errores basados en *buffer overflow* en el área de la pila. Para demostrar la viabilidad de la propuesta realizada se ha desarrollado una herramienta de instrumentación dinámica a modo de prueba de concepto para detectar este tipo de errores. El sistema diseñado permite analizar aplicaciones en busca de este tipo de fallos y puede ser integrado en las fases de prueba en el ciclo de desarrollo software.

Los resultados obtenidos en los experimentos indican que es viable la detección de variables y desbordamientos entre éstas en tiempo de ejecución. Aunque es cierto que se puede producir la detección de algún falso positivo y, por tanto, es necesario analizar manualmente si se trata de un error o no, permite automatizar la búsqueda de errores de este tipo. Además, debido a la semántica de ensamblador y a la dificultad de obtener el número y tamaño exacto de cada una de las variables es complicado evitar este tipo de falsos positivos.

Como se ha comentado previamente, en los experimentos realizados se ha instrumentado sólo el módulo del programa principal, pero la *pintool* está diseñada para poder instrumentar DLLs compartidas o del sistema. El problema de instrumentar estas DLLs es la sobrecarga que esto conlleva a la aplicación, ya que necesitaría detectar cada uno de los *stack frame* de las diferentes funciones que se ejecuten de cada una de las librerías incluidas en el análisis, así como la instrumentación de cada una de las instrucciones de escritura en memoria.

Una posible mejora para reducir la sobrecarga del sistema en caso de querer instrumentar todos los módulos utilizados por la aplicación consiste en aprovechar ese análisis estático realizado previamente y almacenar sólo aquellas funciones que son susceptibles de poseer un *buffer overflow*. Esto se podría conseguir almacenando aquellas funciones que contienen bucles y *buffers* en su interior y aplicar un nivel de profundidad para analizar también las funciones que llaman o son llamadas desde dicha función.

Para la realización de esta tarea, se puede utilizar las interfaces de programación ofrecidas por *radare2* o *IDA* para la creación de *scripts* o *plugins* y utilizar la información ofrecida por su motor de análisis. Esto permite reducir el coste extra añadido por la instrumentación de funciones que no son propensas a contener vulnerabilidades de este tipo.

Agradecimientos

Este trabajo ha sido cofinanciado por el Ministerio de Economía y Competitividad y la Comisión Europea bajo el proyecto TIN2015-66972-C5-2-R (MINECO/FEDER) y por el fondo social europeo y la Junta de Comunidades de Castilla-La Mancha bajo el programa de formación integral para el fomento del empleo de jóvenes investigadores (OE-154).

Referencias

[1] "INTRODUCCION A LA EXPLOTACION DE SOFTWARE." [Online]. Available: http://sistemasoperativos.forogratuito.net/t91-introduccion-a-la-explotacion-de-software

[2] I. Arce, "The shellcode generation," *IEEE Security Privacy*, vol. 2, no. 5, pp. 72–76, Sep. 2004.

[3] D. Bruening and Q. Zhao, "Practical Memory Checking with Dr. Memory," in *Proceedings of the 9th Annual IEEE/ACM International Symposium on Code Generation and Optimization*, ser. CGO '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 213–223. [Online]. Available: http://dl.acm.org/citation.cfm?id=2190025.2190067

[4] M. Shudrak, "WinHeap Explorer: Efficient and Transparent Heap-Based Bug Detection in Machine Code," in *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, Jul. 2017, pp. 94–101.

[5] "Pin - a dynamic binary instrumentation tool," https://software.intel.com/enus/articles/pin-a-dynamic-binary-instrumentation-tool.

[6] "Valgrind developers," http://valgrind.org/.

[7] "O. ravnas. frida," http://www.dynamorio.org/.

[8] "Mit and hewlett-packard. dynamorio," http://www.dynamorio.org/.

[9] H. H. Jazi and A. A. Ghorbani, "Dynamic graph-based malware classifier," in *2016 14th Annual Conference on Privacy, Security and Trust (PST)*, Dec 2016, pp. 112–120.

[10] S. D' Alessio and S. Mariani, "Pindemonium: a dbi-based generic unpacker for windows executables," 2016. [Online]. Available: http://hdl.handle.net/10589/120861

[11] R. J. Rodriguez, I. R. Gaston, and J. Alonso, "Towards the detection of isolation-aware malware," *IEEE Latin America Transactions*, vol. 14, no. 2, pp. 1024–1036, Feb 2016.
[12] L. Davi, A.-R. Sadeghi, and M. Winandy, "Ropdefender: A detection tool to defend against return-oriented programming attacks," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security*, ser. ASIACCS '11. New York, NY, USA: ACM, 2011, pp. 40–51. [Online]. Available: http://doi.acm.org/10.1145/1966913.1966920
[13] D. Radu and B. Dang, "Shellcode analysis using dynamic binary instrumentation," 2011.
[14] T. Petsios, V. P. Kemerlis, M. Polychronakis, and A. D. Keromytis, "Dynaguard: Armoring canary-based protections against brute-force attacks," in *Proceedings of the 31st Annual Computer Security Applications Conference*, ser. ACSAC 2015. New York, NY, USA: ACM, 2015, pp. 351–360. [Online]. Available: http://doi.acm.org/10.1145/2818000.2818031
[15] B. Wu, M. Li, B. Zhang, Q. Zhang, and C. Tang, "Directed symbolic execution for binary vulnerability mining," in *2014 IEEE Workshop on Electronics, Computer and Applications*, May 2014, pp. 614–617.
[16] J. Ma, P. Zhang, G. Dong, S. Shao, and J. Zhang, "Twalker: An efficient taint analysis tool," in *2014 10th International Conference on Information Assurance and Security*, Nov 2014, pp. 18–22.
[17] Q. Zou, W. Huang, J. An, and W. Fan, "Identify stack overflow exploits with dynamic binary instrumentation," in *2015 International Conference on Industrial Informatics - Computing Technology, Intelligent Technology, Industrial Information Integration*, Dec 2015, pp. 263–267.
[18] K. Serebryany, D. Bruening, A. Potapenko, and D. Vyukov, "Addresssanitizer: A fast address sanity checker," in *Proceedings of the 2012 USENIX Conference on Annual Technical Conference*, ser. USENIX ATC'12. Berkeley, CA, USA: USENIX Association, 2012, pp. 28–28. [Online]. Available: http://dl.acm.org/citation.cfm?id=2342821.2342849
[19] C. Carballal, "Estudio, análisis y modelado de memorias caché compartidas bajo administración dinámica," Master's thesis, Universidad de Buenos Aires, 2011.
[20] S. Álvarez, "Radare2," http://radare.org/r/index.html.
[21] "Ida dissasembler," https://www.hex-rays.com/products/ida/.
[22] "Offensive security exploit database archive," https://www.exploit-db.com.
[23] "Common vulnerabilities and exposures," https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2009-1642.
[24] "Common vulnerabilities and exposures," https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-4730.
[25] "Common vulnerabilities and exposures," https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2013-5019.

RAMSES: Plataforma Forense contra el Cibercrimen

Esteban Alejandro Armas Vega[1], Ana Lucila Sandoval Orozco[1], Edgar González Fernández[1]
Carlos Quinto Huaman[1], Daniel Povedano Álvarez[1], Antonio López Vivar[1], Luis Javier García Villalba[1*]
Julio Hernandez-Castro[2], Tatiana Silva[3], Alejandro Prada[3]

[1]Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, España
Emails: {esarmas, edggonza, cquinto, dpovedano}@ucm.es, {asandoval, alopezvivar, javiergv}@fdi.ucm.es

[2]School of Computing, University of Kent
Cornwallis South Building, Office 120
Canterbury CT2 7NF, United Kingdom
Email: j.c.hernandez-castro@kent.ac.uk

[3]Treelogic, Avda. Manoteras 38, Oficina D614, 28050 Madrid, España
Email: {tatiana.silva, alejandro.prada}@treelogic.com

***Resumen*—Tomar posesión de elementos, personas o información que tenga valor para otros es una actividad criminal bien documentada y con un fuerte motivo de lucro. Esta práctica se ha adaptado para satisfacer las oportunidades que presenta Internet. El ransomware y los troyanos bancarios son un tipo particular de malware que explota la confianza y/o conocimiento técnico limitado del usuario para tomar control de su ordenador con el objetivo de cifrar sus datos y pedir un rescate o adquirir datos de acceso de cuentas bancarias almacenadas en el mismo. La forma de lograrlo varía, pero el motivo de lucro de los perpetradores proporciona un elemento común e independiente al vector de infección. El proyecto Europeo RAMSES centra sus esfuerzos en el diseño y desarrollo de una plataforma de software inteligente que facilite y digitalice las investigaciones forenses llevadas a cabo por las fuerzas de seguridad. El proyecto se encuentra en el ecuador de su desarrollo y en este trabajo se presenta los avances más significativos dentro del marco de investigación que se lleva acabo en este proyecto europeo.**

***Index Terms*—Análisis Forense, Bitcoin, Cibercrimen, Ciberseguridad, Criptomoneda, Dark Net, Deep Web, Horizonte 2020, Malware, Plataforma, Ransomware, Sistema Inteligente, Troyano Bancario**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

En la última década, la presencia de "males" de Internet se ha convertido en un asunto serio de combatir y mas aún si se considera el número de usuarios de Internet alrededor del mundo y el uso de nuevas tecnologías como IoT. Con el paso del tiempo el malware desarrollado es más sofisticado, inteligente y versátil que afecta a una amplia variedad de dispositivos [1]. La mayoría de los tipos de malware se centran en el lucro monetario a través de la extracción de valor en efectivo de sus víctimas. Pero la trazabilidad inherente de las monedas fiduciarias y la naturaleza altamente centralizada del sistema bancario convencional hacen que demandar dinero en metálico por parte de los cibercriminales sea muy complicado. Las transferencias de efectivo, a menos que sean rigurosamente lavadas, son altamente rastreables, aumentando el riesgo de los cibercriminales de ser expuestos y arrestados. Sin embargo, con el surgimiento de criptomonedas como el Bitcoin, una alternativa digital descentralizada relativamente anónima que elimina casi por completo los problemas de trazabilidad y rastreo les permite operar de manera prácticamente anónima y sin consecuencias.

Tanto los troyanos bancarios como el ransomware permiten a los atacantes monetizar cada infección casi directamente. Esta característica puede justificar el repentino crecimiento en el número de casos reportados en los últimos años. Su propagación explosiva está alimentada por el hecho de que, cualquier persona pueda usarlos independientemente de su nivel de habilidad. Esto se debe a que una economía clandestina activa (a veces referida como "Crimen como un servicio") que proporciona los recursos requeridos para llevar a cabo la propagación y monetización del delito.

Así mismo, el creciente número de aplicaciones de software que permiten alterar imágenes digitales y la facilidad de utilizarlas debilitan su credibilidad. Este problema unido a la facilidad de distribución de la información a través de Internet ha provocado que la sociedad tienda a aceptar como cierto todo lo que ve sin cuestionar su veracidad. En el caso de que una de estas imágenes (vídeo o fotografía) sea utilizada en una corte judicial, se hace necesario el uso de herramientas de análisis forense (técnicas de detección de manipulación e identificación de fuente de origen) adecuadas a las nuevas tecnologías. Es por eso que el proyecto RAMSES centra su esfuerzo en el desarrollo de una plataforma de herramientas forenses que faciliten el trabajo de las agencias policiales en su lucha contra el cibercrimen, desde un punto de vista operativo y forense. La plataforma RAMSES combina la extracción de datos de la web pública, la detección de manipulación y de mensajes ocultos en imágenes y vídeos, así como el seguimiento de pagos en relativos a campañas de ransomware y *banking trojans*, además del análisis de grandes volúmenes de datos provenientes de la internet superficial y profunda.

II. Plataforma

El objetivo de RAMSES es diseñar y desarrollar una plataforma inteligente dirigida a las fuerzas de seguridad para facilitar sus investigaciones forenses digitales. En la Figura 1 se puede observar el esquema general de funcionamiento de la plataforma. A continuación, se describen brevemente cada uno de sus componentes y el estado actual de su desarrollo.

II-A. Adquisición de Datos

Esta herramienta se encarga de la adquisición de información proveniente tanto de la red superficial de Internet como de la red profunda y de la red oscura. Utiliza una infraestructura basada en big data para proveerla de amplias capacidades de adquisición, almacenamiento y procesamiento de los datos adquiridos. Estas capacidades son importantes debido a la cantidad de información que debe procesar y a lo poco estructurado de los datos obtenidos. De esta forma, RAMSES permite el uso de datos sin procesar y también de información procesada para apoyar el trabajo del conjunto de herramientas que conforman la plataforma. Asimismo, permite obtener información de inteligencia que permite relacionar sospechosos con algún otro tipo de crimen del cual no se haya tenido conocimiento previo. En esta etapa del proyecto, la herramienta de adquisición de datos junto con la API de comunicación correspondiente, se encuentra en fase final de desarrollo. Posteriormente, se realizará la integración con el resto de herramientas que conforman RAMSES.

II-B. Análisis de Muestras de Malware

Esta herramienta es capaz de ejecutar cualquier muestra de malware que se adquiera y se envíe a través de la plataforma RAMSES. Con su ejecución se adquiere datos de comportamiento, recursos utilizados y las comunicaciones que realiza hacia servidores externos. De esta forma, el analista forense será capaz de obtener información suficiente para relacionar las campañas de malware llevadas a cabo por una misma organización criminal. Esta herramienta se encuentra en la fase final de su desarrollo.

Figura 1. Esquema de la plataforma RAMSES.

II-C. Análisis Forense de Ficheros Multimedia

Como parte de las herramientas de análisis forense de ficheros multimedia que RAMSES posee se encuentran: (1) Identificación de la fuente de adquisición, (2) Análisis de manipulaciones en imágenes y vídeos y (3) Análisis esteganográfico de imágenes y vídeos. Todas utilizan algoritmos de estado del arte.

- La herramienta de identificación de la fuente de adquisición es capaz de determinar la fuente de origen de una fotografía digital o un vídeo digital.
- La herramienta de análisis de manipulaciones utiliza algoritmos capaces de detectar la presencia de modificaciones dentro de la escena así como también dentro de la estructura del fichero.
- La herramienta de análisis esteganográfico permite identificar la presencia de datos ocultos y en la mayor parte de los casos es capaz de extraer dicha información.

Este componente de la plataforma RAMSES se encuentra en su etapa final de desarrollo.

II-D. Rastreo de Pagos con Bitcoin

El rastreo de las transacciones realizadas por concepto de pago de rescate se realiza de forma automática, mediante el uso de una herramienta específicamente desarrollada para este fin. Para ello, RAMSES requiere de la dirección de monedero de bitcoin sospechosa y mediante el uso de los metadatos públicos del blockchain de bitcoin se recrea todo el flujo de transacciones relacionadas que se siguió desde el primer momento. Gracias a su integración con el resto de herramientas, una vez que se determinan las carteras vinculadas al proceso de blanqueo de dinero, esta información es introducida a la herramienta de adquisición de datos con el fin de correlacionar datos y de esta manera hallar sospechosos. Esta herramienta se encuentra en la etapa final de su desarrollo.

III. Conclusiones y Trabajo Futuro

RAMSES se encuentra en el ecuador de su desarrollo.Todos sus componentes se encuentran en la etapa final de su desarrollo y el siguiente paso es integrarlos para consolidar la plataforma completa, lo cual sucederá en los siguientes 6 meses. Posterior a esto la plataforma y todos sus componentes entraran en la etapa de prueba con los cuerpos de policía que colaboran con el consorcio, de esta manera todas y cada una de las necesidades especificas de estos usuarios serán ajustadas y agregadas (de ser necesario) a la plataforma.

Agradecimientos

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700326. Website: http://ramses2020.eu

Referencias

[1] EUROPOL, "The Internet Organised Threat Assessment (iOCTA) 2014," http://tinyurl.com/javlulb, March 2017.

A Review of "A Tool to Compute Approximation Matching between Windows Processes"

Miguel Martín-Pérez, Iñaki Abadía
Dpto. de Informática e Ingeniería de Sistemas
Universidad de Zaragoza, Zaragoza, Spain
{miguelmartinperez, 685867}@unizar.es

Ricardo J. Rodríguez
Centro Universitario de la Defensa de Zaragoza, Spain
Università degli Studi della Campania "Luigi Vanvitelli", Italy
rjrodriguez@unizar.es

***Abstract*—Finding identical digital objects (or artifacts) during a forensic analysis is commonly achieved by means of cryptographic hashing functions, such as MD5, SHA1, or SHA-256, to name a few. However, these functions suffer from the *avalanche* effect property, which guarantees that if an input is changed slightly the output changes significantly. Hence, these functions are unsuitable for typical digital forensics scenarios where a forensics memory image from a likely compromised machine shall be analyzed. This memory image file contains a snapshot of processes (instances of executable files) which were up on execution when the dumping process was done. However, processes are relocated at memory and contain dynamic data that depend on the current execution and environmental conditions. Therefore, the comparison of cryptographic hash values of different processes from the same executable file will be negative. Bytewise approximation matching algorithms may help in these scenarios, since they provide a similarity measurement in the range $[0, 1]$ between similar inputs instead of a yes/no answer (in the range $\{0, 1\}$). In this paper, we introduce `ProcessFuzzyHash`, a Volatility plugin that enables us to compute approximation hash values of processes contained in a Windows memory dump.**

Tipo de contribución: *Investigación ya publicada*

I. Extended Abstract

Computer forensic analysts usually try to find identical digital objects (or *artifacts*, defined as an arbitrary byte sequence) as a source of evidences for a posterior analysis. For instance, they can look for the presence of known malicious software (*malware*) in a forensic disk image as an evidence of a compromise computer. When some malware files are detected, they are analyzed in detail to measure the impact of the threats found.

A common approach to compute the similarity between artifacts is to use cryptographic hash functions like MD5, SHA-1, or SHA-256. A cryptographic hash function is an algorithm that takes a relatively (arbitrary) large amount of input and returns a fixed-size hexadecimal string. The output string is called the *hash value* or *digest* of the input. Hence, a hash value of an artifact serves to unequivocally identify it. Cryptographic hash functions are designed to be one-way function [1], i.e., it is easy to compute on every input, but hard to compute its inverse function.

A desirable property of cryptographic hash functions is the avalanche effect property [2], which guarantees that the hash values of two similar, but not identical, inputs (e.g., inputs in which only a single bit is flipped) produce radically different outputs. Hence, cryptographic hash functions are commonly used for data integrity and file identification of a seized device [3].

However, the avalanche effect property has some drawbacks since an active adversary may easily defeat this detection and keep their files hidden by just flipping a single bit. To overcome this situation, approximate matching has emerged in the recent years as a prominent approach that is more robust to active adversaries than traditional hashing [3]. Approximate matching identifies similarities between two digital artifacts providing a measure of similarity, in the range of $[0, 1]$. Hence, it is used to find artifacts that resemble each other or to find artifacts that are contained in another artifacts [4], providing a percentage of similarity among them.

An approximate matching method becomes especially suitable for the analysis of forensic memory images, which contains a snapshot of the set of processes up on execution when the memory image was acquired. By the intrinsic characteristics of the underlying machine's OS, different executions of the same binary file generates similar but not identical processes, residing in the addressable space memory of the machine. Hence, although cryptographic hash functions are useful to identify the binary file in a forensic disk image, they become unsuitable for the same purpose when analyzing a forensic memory image from the same machine.

An approximate matching method is classified based on what is considered to perform the similarity analysis [4]: *bytewise*, when the method relies only on the byte sequence of the digital artifact (i.e., no structures or meaning of the byte stream are considered); *syntactic*, when the method relies on internal structures present in digital artifacts under analysis; and *semantic*, when the method uses contextual attributes of the digital artifact to interpret it. In this paper, we focus on bytewise approximation matching algorithms (also known as fuzzy hashing) from a syntactic approach, since we consider process sections as input data for the algorithms.

Let us illustrate the avalanche effect problem in processes by means of a running example. Consider the "hello world" program (coded in C) shown in Listing 1 that has a `getchar` function as a way to wait for user interaction before exiting, compiled with GNU `gcc` version 5.3.0 running in a Windows 7 SP1 Professional. Now, we execute the binary file three times in the same machine, and every time we dump the corresponding process to a file. The MD5 hashes of each dumped process file are, namely:

- `c5045fa95f880dc1cf09c97ae8d32e28`
- `37304f15fe397b51927ce3ba247ca397`
- `016e7557c48cb91feb487b6e0919037e`

As it can be clearly seen, there is not any match between any of the hashes of the dumped process files. However, if

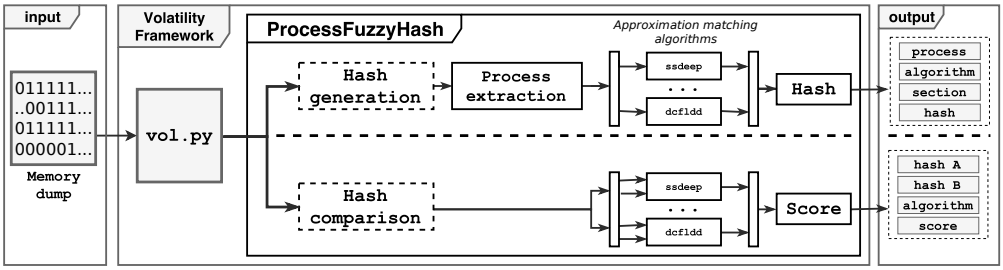


Figure 1: System overview diagram of `ProcessFuzzyHash`.

we compute the ssdeep hash instead (a bytewise approximate matching algorithm), we obtain similarities that range between 97% to 99%, and up to 100%, depending on the byte stream of the dumped process files that we analyze. For instance, the ssdeep hash of the byte stream that contains the binary code executed is exactly the same value, `384:fb02XGJLkOj40sqJnwCQoVrKcBdgE/daL07ticdJ5bjn+lUpNNK/7RV:fxGGOdwCJxRuYDJJraR`.

Listing 1: A "hello world" C program.

```
#include <stdio.h>

int main(int argc, char *argv[]){
    printf("hello world!\n");
    getchar();
    return 0;
}
```

In this paper, we focus on processes executed on top of Windows OS. The main reason of the differences regarding cryptographic hashes relies on the own nature of Windows processes. While an executable file is a *static binary file*, a process of such an executable file is a *dynamic binary file*. There exist several parts in a process which strongly depend on the current time of execution, such as the content of the stack or the heap. Besides, relocation and relative addresses of import data or functions are likely to differ between subsequent executions.

Furthermore, the intrinsic nature of Address Space Layout Randomization (ASLR) [5], [6], a defense mechanism which randomize the base address of sections and libraries to prevent control-flow hijacking attacks [7] also causes this issue. This defense produces that the Windows image loader may randomly relocate the executable image file when mapped to memory. Therefore, code and data in the program are adjusted to reflect those randomly assigned addresses.

In this paper, we introduce `ProcessFuzzyHash`, a plugin of the Volatility framework that allows us to compute approximation matching hashes of the processes inside a memory dump. Currently, the tool supports four approximate matching hashing methods, `dcfldd`, `ssdeep` [8], `sdhash` [9] , and `TLSH` [10].

`ProcessFuzzyHash` operates in two different manners: *hash generation* and *hash comparison*. A high-level description of `ProcessFuzzyHash` is depicted in Figure 1. As input, our tool needs a dumped memory image file for a machine running any Windows OS accepted by Volatility.

Hash Generation. `ProcessFuzzyHash` relies on `procdump` to obtain the processes inside a memory dump, and on `memdump` to obtain all memory pages related to a given process (given also by parameter). Those are already existing Volatility plugins. Besides the processes for which approximation matching hashes shall be computed, `ProcessFuzzyHash` also needs the user to provide as argument which sections of the processes shall be considered.

Hash Comparison. `ProcessFuzzyHash` requires one (or more) hashe(s) and the approximation matching algorithm used as base comparison. `ProcessFuzzyHash` supports two comparison modes: either with a file containing a bunch of hash values line separated, or with the hash values of processes contained in a dumped memory image file.

Tool Availability. `ProcessFuzzyHash` was released as a plug-in included in the Volatility Framework under GNU GPLv3 license: https://github.com/volatilityfoundation/community/tree/master/ProcessFuzzyHash. The full research paper was published in [11].

Acknowledgements

This work was supported by Spanish National Cybersecurity Institute (INCIBE) "Ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad", grant numbers INCIBEC-2015-02486 and INCIBEI-2015-27300.

References

[1] O. Goldreich, *Foundations of Cryptography: Volume 1*. New York, NY, USA: Cambridge University Press, 2006.

[2] A. F. Webster and S. E. Tavares, "On the Design of S-Boxes," in *Advances in Cryptology — CRYPTO '85 Proceedings*. Springer Berlin Heidelberg, 1986, pp. 523–534.

[3] V. S. Harichandran, F. Breitinger, and I. Baggili, "Bytewise Approximate Matching: the Good, the Bad, and the Unknown," *Journal of Digital Forensics, Security and Law*, vol. 11, no. 2, 2016.

[4] F. Breitinger, B. Guttman, M. McCarrin, V. Roussev, and D. White, "Approximate Matching: Definition and Terminology," National Institute of Standards and Technology, techreport NIST Special Publication 800-168, May 2014.

[5] E. Bhatkar, D. C. Duvarney, and R. Sekar, "Address Obfuscation: An Efficient Approach to Combat a Broad Range of Memory Error Exploits," in *Proceedings of the 12th USENIX Security Symposium*, 2003, pp. 105–120.

[6] PaX Team, "PaX address space layout randomization (ASLR)," https://pax.grsecurity.net/docs/aslr.txt.

[7] L. Szekeres, M. Payer, T. Wei, and D. Song, "SoK: Eternal War in Memory," in *2013 IEEE Symposium on Security and Privacy*, May 2013, pp. 48–62.

[8] J. Kornblum, "Identifying Almost Identical Files Using Context Triggered Piecewise Hashing," *Digital Investigation*, vol. 3, no. Supplement, pp. 91–97, 2006.

[9] V. Roussev, "Data Fingerprinting with Similarity Digests," in *Advances in Digital Forensics VI*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 207–226.

[10] J. Oliver, C. Cheng, and Y. Chen, "TLSH – A Locality Sensitive Hash," in *Proceedings of the 2013 Fourth Cybercrime and Trustworthy Computing Workshop*, Nov. 2013, pp. 7–13.

[11] R. J. Rodríguez, M. Martín-Pérez, and I. Abadía, "A Tool to Compute Approximation Matching between Windows Processes," in *Proceedings of the 2018 6th International Symposium on Digital Forensic and Security (ISDFS)*, 2018, accepted for publication. To appear.

Detección de ataques de Denegación de Sostenibilidad Económica en redes Autoorganizadas

M.A. Sotelo Monge, J. Maestre Vidal y L.J. García Villalba

Grupo de Análisis, Seguridad y Sistemas (GASS, http://gass.ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, España
E-mail: {masotelo, jmaestre}ucm.es, javiergv@fdi.ucm.es

***Resumen*—En este artículo se revisa el problema de los ataques de Denegación de Sostenibilidad Económica (EDoS) en escenarios emergentes de red. En particular se profundiza en los problemas de seguridad inherentes a soluciones que adoptan estrategias de autoorganización y virtualización de funciones de red. A lo largo de la investigación realizada se han definido dos tipos de amenazas: EDoS basada en la explotación de la carga de trabajo, y EDoS basada en su instanciación. Con el fin de contribuir a su mitigación se propone una arquitectura de seguridad con capacidades de detección de intrusiones en red, las cuales implementan métodos de aprendizaje automático, modelado del comportamiento de la red, construcción de umbrales de decisión y agrupamiento de instancias de red en base a su productividad. La experimentación realizada ha considerado diferentes parámetros de ajuste y circunstancias de red, en las cuales se ha demostrado su efectividad.**

***Index Terms*—autoorganización de redes, denegación económica de sostenibilidad, detección de intrusiones, seguridad de la información, virtualización de funciones de red**

Tipo de contribución: *Investigación original*

I. Introducción

La complejidad y sofisticación de las arquitecturas de red crece día a día, pero también lo hace la demanda de paradigmas de gestión más rápidos, efectivos y escalables. En los últimos años las redes 5G se han postulado como una prometedora manera de alcanzar estos requisitos y superar los desafíos que plantean los nuevos escenarios de comunicación [1]. Éstos motivan la integración inteligente de los avances más innovadores en redes de comunicaciones, como virtualización de funciones de red o NFV (del inglés *Network Function Virtualization*), computación en la nube (del inglés *Cloud Computing*), redes definidas por software o SDN (del inglés *Software-Defined Networking*), inteligencia artificial y redes autoorganizadas o SON (del inglés *Self-Organizing Networks*). En particular, la sinergia entre tecnologías SDN y SON es considerada uno de los elementos más relevantes para cumplir los indicadores claves de rendimiento a los que aspiran las redes 5G [2]. Debido a esto, es frecuente que proyectos 5G integren estas herramientas con el fin de alcanzar un conocimiento cognitivo del estado de la red con el fin de mejorar su capacidad de adaptación [3]. Un ejemplo claro de esta aproximación se observa en el proyecto SELFNET [4], donde se propone una estrategia de gestión y autoorganización de redes de nueva generación capaz de aprovechar estas capacidades. La investigación descrita en este artículo se centra en las redes SON como soluciones prometedoras a los retos anteriormente mencionados. Originalmente fueron propuestas como respuesta al problema de la optimización de la eficiencia de redes móviles LTE [5]. Consecuentemente, en el marco estandarizado por el 3GPP para su desarrollo se resalta su capacidad de reducción de costes operacionales por medio de procesos de automatización de tareas [6], postulando una transición de los esquemas de gestión convencionales que requieren de la intervención de operadores humanos (bucle abierto) a su completa automatización (bucle cerrado). Otro aspecto fundamental en la investigación realizada es el estudio del papel que juega en este contexto la virtualización en entornos de computación en la nube con el fin de mejorar la escalabilidad de las funciones de red [7], lo cual permite reducir el coste de los despliegues de sensores/actuadores involucrados en esquemas SON. Estas capacidades a menudo se orquestan mediante políticas de autoescalado, las cuales pueden llegar a ser explotadas con la motivación de causar sobrecostes en los sistemas víctima, acción típicamente reconocida como ataques de Denegación Económica de Sostenibilidad o EDoS (del inglés *Economical Denial of Sustainability*)[8][14]. Éstas plantean amenazas bien conocidas, pero que apenas han llamado la atención de la comunidad investigadora, entre la cual es frecuente su confusión con ataques de denegación de servicio basados en inundación o complejidad. Su crecimiento y evolución va de la mano del avance de las tecnologías de red involucradas en escenarios de nueva generación, por lo que cada vez es más necesario el desarrollo de medidas para su mitigación. La investigación descrita en este artículo contribuye a los esfuerzos realizados mediante la revisión en profundidad del problema de los ataques EDoS en redes convencionales y su adaptación a escenarios autoorganizados. Esto ha llevado a la distinción de dos familias principales de amenazas: EDoS basada en explotar la carga de trabajo de elementos de red (W-EDoS) y EDoS basada en la instanciación por causas fraudulentas de funciones de virtualización de red (I-EDoS). También se propone una arquitectura multicapa para su detección, en la que se combinan técnicas de análisis basadas en aprendizaje automático, predicción y agrupamiento. La eficacia tanto de los ataques descritos como de las contramedidas desarrolladas ha sido evaluada en un escenario de SON real, arrojando resultados preliminares prometedores. Este artículo se divide en siete secciones, siendo la primera de ellas la presente introducción. En la Sección II se revisan el problema de

los ataques EDoS en entornos de red SON y las diferentes propuestas para su mitigación. En la Sección III se definen los ataques W-EDoS e I-EDoS. En la Sección IV se introduce un sistema para la detección de las amenazas EDoS. En la sección V se explica la metodología de evaluación adoptada. En la Sección VI se discuten los resultados obtenidos. Finalmente, en la Sección VII se presentan las conclusiones y propuestas de trabajo futuro.

II. Trabajos Relacionados

Esta sección describe las principales características de los ataques EDoS y el progreso de la comunidad investigadora hacia su mitigación.

II-A. Denegación de la Sostenibilidad Económica

La expresión Denegación de Sostenibilidad Económica fue acuñada por C. Hoff en el año 2008 [9][10] para referirse a la familia de ataques en red originalmente dirigidos contra plataformas de computación en la nube, en la que el intruso tiene por objetivo el aumento fraudulento del mantenimiento de los servicios contratados por los clientes. Por lo tanto, su primera consecuencia es el dañar la viabilidad de un contrato a raíz de lo cual el cliente debe de pagar cantidades altas, llevando al cese de su negoció o forzando su migración a otro proveedor. Interesado por esta nueva amenaza, R. Cohen [11] extendió su definición señalando la explotación de vulnerabilidades de los procesos de autoescalado como principal medio para lograr dicho fraude, planteamiento que hasta ahora ha respaldado mayoritariamente la comunidad investigadora. Estas intrusiones también son denominadas ataques de reducción de la calidad de servicio o RoQ (del inglés *Reduction of Quality*) [12], o se refieren a amenaza relacionadas con el consumo fraudulento de recursos o FCR (del inglés *Fraudulent Resource Consumption*) [13] que afecta a los servicios de pago-por-servicio ofrecidos por los proveedores de computación en la nube [14]. A diferencia de los ataques DoS basados en inundación, tratan de pasar desapercibidas ante elementos de monitorización mediante el registro de distribuciones de clientes y peticiones que asemejen a las del tráfico habitual y legítimo [9][10]. Por lo tanto, es frecuente que encaminen la intrusión mediante solicitud al sistema víctima de peticiones computacionalmente caras [13]. Esto también establece una diferencia representativa con eventos de naturaleza legítima capaces de comprometer la disponibilidad del sistema protegido, como por ejemplo el acceso masivo de usuarios legítimos a los servicios prestados (del inglés *flash crowds*) [15]. En la actualidad existen diferentes técnicas para perpetrar amenazas EDoS, como por ejemplo la solicitud de ficheros grandes o consultas complejas a bases de datos [16], peticiones HTTP lazadas desde contenido XML [17], o la explotación de vulnerabilidades específicas de las diversas plataformas que soportan servicios web [18]. Además de causar un impacto económico, los ataques EDoS pueden derivar en otro tipo de riesgos. G. Sonami et al. revisaron este problema señalando sus diferentes consecuencias colaterales, las cuales varían en función del papel que desempeña cada actor en la computación en la nube. Por ejemplo, el proveedor tiende a perder reputación y clientes que deciden contratar servicios más baratos a la competencia. Por otro lado, el cliente pagará una cantidad excesiva de dinero por servicios que no ha estado utilizando. Estos ataques también afectan la capacidad operacional de los servicios y capas de procesamiento de información que soportan los servicios atacados, como por ejemplo infraestructura, virtualización de funciones de red o *multi-tenacy* [19][12].

II-B. Defensa contra ataques EDoS

A pesar de la relevancia de esta amenaza en las redes venideras, la bibliografía no recoge una extensa cantidad de publicaciones centradas en los desafíos que plantea. Los estudios que afrontan este problema habitualmente proponen soluciones basadas en el análisis de métricas a nivel de red típicas de la denegación de servicio basada en inundación. Con el propósito de facilitar su entendimiento, éstos se clasifican tal y como se organizan en la investigación relacionada con ataques DDoS [19]: detección, mitigación o prevención, e identificación de la fuente.

Detección. Las aproximaciones en este campo tienen como objetivo la identificación de amenazas. Algunas publicaciones al respecto analizan métricas a nivel local para modelar el consumo de recursos y la aplicación de procesos de autoescalado [19]. Otras publicaciones analizan información a nivel de red [21], y los hábitos de navegación de los clientes [20]. Las investigaciones enfocadas en métricas locales han demostrado ser eficaces, siendo las que más se ajustan a la definición de ataques EDoS propuesta por Hoff [9][10]. Sin embargo, los métodos basados en métricas de red aprovechan las ventajas del estado del arte con respecto a los ataques de denegación de servicio basados en inundación.

Mitigación y prevención. Las propuestas de mitigación en este campo se enfocan en incrementar las restricciones del sistema protegido a través de técnicas de control de acceso. Los test de Turing basados en reconocimiento de imágenes [22] o resolución de puzles criptográficos [23], son usualmente las técnicas más empleadas. En contraste con las técnicas de detección, estos métodos no requieren la identificación de la amenaza. Sin embargo, muchas de las propuestas categorizadas como mitigación pueden ser implementadas como medidas de prevención.

Identificación de la fuente. Finalmente, las investigaciones que buscan identificar el origen de los ataques intentan descubrir al propio atacante. No obstante, y dada la complejidad que esto implica, la identificación del origen se reduce frecuentemente a localizar al atacante lo más cerca posible. El estado del arte en defensa contra ataques DDoS sirven para este propósito [24], entre ellos los que se basan en análisis de mensajes de error [25], despliegue de tarros de miel (del inglés *honeypots*) [27] o marcado de paquetes [26].

III. Ataques EDoS en redes autoorganizadas

Hoff [9][10] destaca la similitud que poseen los ataques EDoS con respecto al tráfico legítimo. Es asumible entonces que, en el contexto de una arquitectura cliente-servidor, aquella similitud se exprese en función del conjunto de clientes y las peticiones generadas, tomando en cuenta en cada caso su número y distribución en el tiempo. Estas condiciones se presentan tanto en escenarios de tráfico normal como de ataque EDoS y son, por consiguiente, asumidas a lo largo de este trabajo. Las siguientes secciones definen cada tipo de ataque, sus características e impacto en el modelo de costes.

III-A. EDoS basada en Carga de Trabajo (W-EDoS)

Un ataque de Denegación de Sostenibilidad Económica basada en Carga de Trabajo (W-EDoS) está caracterizado por la ejecución de operaciones de alto coste computacional en las instancias virtuales alojadas en algún proveedor de servicios en la nube. Dichas operaciones se ejecutan del lado del servidor y generan una alta carga de trabajo en respuesta a peticiones cliente aparentemente legítimas. Bajo esta premisa, se asume la existencia de un ataque W-EDoS cuando un entorno de red monitorizado presenta condiciones de similitud con tráfico de red legítimo, pero cuya carga de trabajo promedio por petición es significativamente mayor, tanto en número como en distribución en el tiempo. La Fig. 1 muestra una representación de un ataque W-EDoS lanzado sobre una función de red virtual instanciada. El efecto de un ataque W-EDoS es la necesidad de escalar vertical u horizontalmente las instancias desplegadas, añadiendo para ello recursos (cómputo, almacenamiento, etc.) cuyo pago por uso afecta negativamente la sostenibilidad económica con el proveedor debido a los sobrecostes generados.

III-B. EDoS basada en Instanciación (I-EDoS)

Un ataque de Denegación de Sostenibilidad Económica basada en Instanciación (I-EDoS) está caracterizado por la explotación de alguna vulnerabilidad existente bien en la plataforma de servicios en la nube o bien en la función virtual en sí misma. ocasionando la creación automática de instancias adicionales en uno o varios puntos de la red. De este modo, se observa un incremento en el número de instancias desplegadas, pero con productividad promedio por instancia considerablemente inferior. Bajo esta premisa, se asume la existencia de un ataque I-EDoS cuando un entorno de red monitorizado presenta condiciones de similitud con tráfico de red legítimo; pero con un significativo incremento en el número y distribución de las instancias virtuales, así como un decremento de su productividad media. La Fig. 2 muestra una representación gráfica de un ataque I-EDoS en la que la plataforma de servicios en la nube expone una vulnerabilidad que desencadena la creación de instancias virtuales adicionales con distinto grado de productividad. Aquel grupo de instancias poco productivas generadas por causas fraudulentas ocasiona sobrecostes derivados por el tiempo que permanecen en ejecución. Se produce así, al igual que en el caso de ataques W-EDoS, un impacto negativo sobre la sostenibilidad económica del despliegue realizado.

IV. Principios de Diseño y Arquitectura

La propuesta presentada en esta sección tiene como objetivo distinguir situaciones legítimas, de aquellas relacionadas con ataques EDoS en redes autoorganizadas. A continuación se explican sus principios de diseño y arquitectura.

IV-A. Principios de Diseño

Con el fin de limitar las tareas involucradas en las fases de diseño e implementación, las siguientes funcionalidades y restricciones han sido asumidas.

- La arquitectura propuesta debe ser capaz de detectar ataques W-EDoS e I-EDoS asumiendo las características descritas en la sección anterior, diferenciándolos de actividades legítimas (tráfico normal y *flash crowds*).

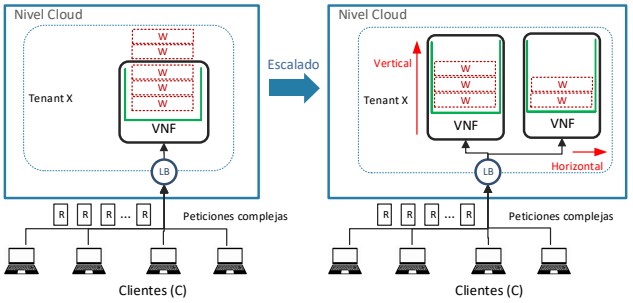


Figura 1: Autoescalado generado por un ataque W-EDoS.

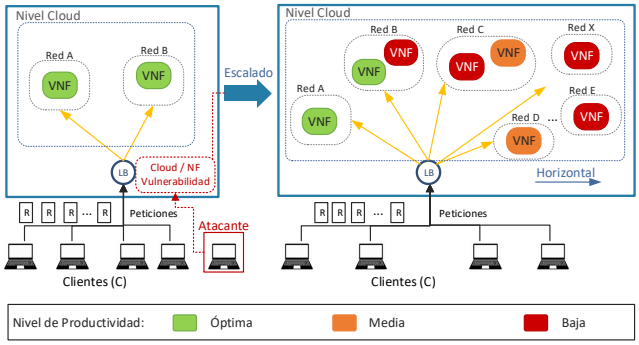


Figura 2: Autoescalado generado por un ataque I-EDoS.

- La detección de ataques convencionales DoS basados en inundación y DDoS están fuera del alcance de esta publicación.
- Se asume un entorno de monitorización no estacionario [29].
- Por razones como las discutidas anteriormente, no se consideran los ataques basados en la imitación o el robo de identidad [28] que tengan como propósito eludir la estrategia de detección propuesta.
- Las redes autoorganizadas son escenarios de monitorización complejos en los que una gran cantidad de sensores recolecta información sobre el estado de la red en tiempo real. Esta información debe ser agregada en observaciones que pueden ser tratadas por herramientas analíticas de alto nivel. Aunque en la experimentación se revisa brevemente el impacto de la granularidad con la que se extraen los datos, la introducción de métodos para su calibración se pospone para futuras investigaciones.
- La correlación y administración de las alertas descubiertas [30] quedan fuera del alcance de esta publicación. Sin embargo, los conocimientos adquiridos deben notificarse.

IV-B. Arquitectura

La arquitectura propuesta que se ilustra en la Fig. 3 ha sido diseñada en concordancia con las arquitecturas de Virtualización de Funciones de Red (ETSI-NFV) [31] y redes de quinta generación, en donde el desacoplamiento de los planos de datos y de gestión hace posible la distinción de las distintas capas funcionales. La Capa de Virtualización es ejecutada sobre la Capa Física comúnmente implementada con hardware comercial de propósito general o COTS (del inglés *Commercial-Off-The-Shelf*). En un nivel superior, la Capa de Cloud gestiona la instanciación automática de funciones

virtuales de red (VNFs) a través de su interacción con la Capa de Virtualización, la cual se encarga de proveer los recursos solicitados. El entorno Cloud desplegado interconecta VNFs a través de la red virtual subyacente componiendo uno o más servicios de red (SR) accesibles a los usuarios. Se asume además que la Capa Cloud posee la capacidad de extraer métricas de monitorización que permitirán su posterior agregación y análisis en la capa Autonómica SON. Los componentes de este último nivel de la arquitectura son detallados a continuación.

Colector de datos. En escenarios de red autoorganizados, los sensores (S) desempeñan un rol importante debido a que son capaces de monitorizar métricas personalizadas a nivel de aplicación, como tiempos de respuesta, consumo de memoria por proceso, etc. Asimismo, las plataformas de computación en la nube poseen herramientas de monitorización (e.g. Ceilometer [32]) capaces de ofrecer un número importante de métricas relacionados con el uso o desempeño de los recursos instanciados; por ejemplo, consumo de CPU o memoria, uso de red, etc. De este modo, se cuenta con información recolectada tanto por los sensores (MCE) como por la plataforma de servicios en la nube sirve (MIV).

Agregación. El alto volumen de datos generados por la monitorización requiere de operaciones de agregación periódicas que permitan reducir su volumen a la vez de generar series temporales. A nivel de aplicación esto se consigue mediante la extracción de características (EC) utilizando métodos como, por ejemplo, el cálculo del grado de entropía. Por su parte, las métricas de la plataforma en la nube son extraídas y agregadas (ARV) a través de consultas a la API de la herramienta de monitorización. En ambos casos, la granularidad de la serie temporal está determinada por la periodicidad con la que se ejecutan las operaciones de agregación.

Detección de ataques EDoS. La detección está compuesta por las etapas de análisis y toma de decisiones. El análisis comprende la generación de un modelo predictivo (Md) aplicado sobre las series temporales y sus resultados sirven para la generación de un intervalo de predicción (UA) basado en el error estimado para cada observación. Se infieren entonces comportamientos inesperados cuando el valor de una métrica analizada se encuentra fuera del intervalo de predicción. De forma complementaria, se clasifican grupos de instancias basados en la similitud (SM) observada en sus indicadores de productividad, dando lugar a la identificación de grupos con baja productividad cuyo origen no se ajusta a un patrón legítimo. En la fase de toma de decisiones, se tienen en cuenta los datos analizados para crear reglas de inferencia destinadas a detectar anomalías (DA) que reflejen la presencia de un ataque EDoS.

Notificación. Las conclusiones inferidas son notificadas como posibles ataques EDoS, y tienen el propósito de evitar la creación de instancias cuyo origen fraudulento genere sobrecostes derivados del uso.

IV-C. Detección de EDoS basada en carga de trabajo

A continuación se describen las métricas y el proceso de análisis implementado para la detección de ataques EDoS basados en la explotación de la carga de trabajo de las funciones de red.

Métricas W-EDoS. De acuerdo con la definición presentada en III-A, este tipo de ataques mantienen las condiciones de similitud de red, pero presentan variaciones significativas en la carga de trabajo, por lo que la estrategia de detección considera como métricas W-EDoS el consumo de CPU (X_{cpu}) y el tiempo de respuesta a nivel de aplicación (X_{app}). El primero mide el consumo de CPU a nivel de sistema operativo, mientras que el segundo mide el tiempo total requerido para procesar cada petición en el servidor. Para descubrir comportamientos inesperados, el primer paso es analizar las variaciones en X_{app}, lo que se consigue estudiando su grado de desorden en intervalos de tiempo fijos. La bibliografía [21], [28] sugiere la correlación de dichas observaciones en términos de entropía. Tal como lo indican Bhuyan et. al. [33], la entropía definida por Rènyi proporciona una solución de propósito general especialmente eficaz en este tipo de problemas. Dicha entropía está definida por $H_\alpha(X_{app})$ en la siguiente ecuación, siendo α el orden ($\alpha \geq 0$ y $\alpha \neq 1$):

$$H_\alpha(X_{app}) = \frac{1}{1-\alpha} \log \sum_{i=1}^{n} P_i^\alpha \tag{1}$$

donde X es la variable aleatoria con n posibles resultados y correspondientes P_i (i=1,2,...,n) probabilidades. A efectos de la experimentación, se considera la versión normalizada $H_\alpha(X_{app})/\log n$. Cuando $\alpha = 1$ se observa el caso particular en el que la entropía de Rènyi coincide con la de Shannon. Las sucesivas mediciones de la entropía dan lugar a la creación de una serie temporal del tipo:

$$H_\alpha(x_{app})_{t=0}, H_\alpha(x_{app})_{t=1}, \cdots, H_\alpha(x_{app})_{t=n} \tag{2}$$

Y las medidas del consumo de CPU están representadas por la serie temporal:

$$(x_{cpu})_{t=0}, (x_{cpu})_{t=1}, \cdots, (x_{cpu})_{t=n} \tag{3}$$

Dado que el resto de procesos analíticos son iguales para X_{cpu} y X_{app}, en los siguientes puntos se utiliza X para referirse indistintamente a cualquiera de ellos.

Comportamientos inesperados W-EDoS. El método de detección propuesto recae en decidir si la estimación $\hat{X}_{t=m}$ en el horizonte m, difiere significativamente de $X_{t=m}$. Para ello, es necesario predecir la serie temporal de la variable X hasta un horizonte predeterminado, lo que permite comparar el valor estimado con la observación real. Para la predicción, se ha utilizado el algoritmo de Alisamiento Exponencial Doble o DES (del inglés *Double Exponential Smoothing*) [34] debido a que reduce el tiempo de adaptación al requerir series temporales más cortas para el modelado, lo que deja de lado enfoques autorregresivos como ARIMA. Sus parametros de ajuste son autocalibrados tal y como es descrito en [35]. A su vez, en lugar de estimar las variaciones con respecto a estimaciones puntuales, se construyen intervalos de predicción, como se sugiere en [36]. Estos se expresan considerando el error de predicción ϵ_t basado en la distancia Mahalanobis en t, en particular cuando $t = m$, según la siguiente fórmula.

$$\epsilon_t = \sqrt{(x_m - \hat{x}_m)^2} \tag{4}$$

De esta forma, el intervalo de predicción (PI) es expresado como:

$$PI = x_{t=n} \pm \eta \sqrt{\sigma^2(\epsilon_t)} \tag{5}$$

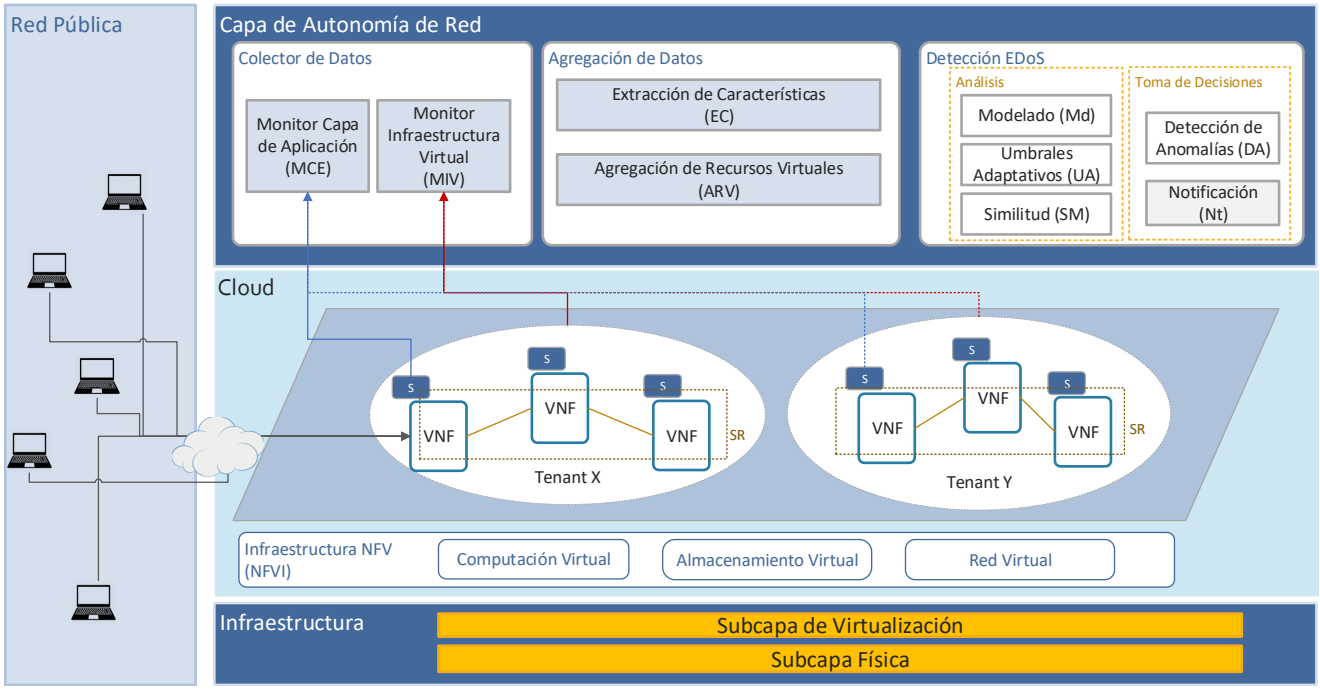


Figura 3: Arquitectura SON para la Detección de Ataques EDoS

donde $\sigma^2(\epsilon_t)$ es la varianza del error de predicción. Por lo tanto, sea $X^n_{t=0}$ y su predicción $\hat{X}_{t=n+m}$ en el horizonte m, la observación $X_{t=n+m}$ representa un comportamiento inesperado de carga de trabajo cuando $\epsilon_t \notin PI$, es decir, cuando $\hat{x}_{t=n+m}$ y $x_{t=n+m}$ difieren significativamente. Dado que X_{cpu} es independiente de X_{app}, el sistema propuesto considera que una observación en t constituye un ataque W-EDoS si ambos son discordantes en ese instante, y además X_{cpu} presenta una tendencia creciente.

IV-D. Detección de EDoS basada en instanciación

A continuación se describen las métricas y procesos analíticos involucrados en la detección de amenazas I-EDoS.

Métricas I-EDoS. De acuerdo con la definición presentada en III-B, en este tipo de ataques se mantienen las condiciones de similitud de red. Sin embargo, se observan variaciones significativas en el comportamiento de las instancias virtuales y su productividad. Además, estos ataques están caracterizados por la aparición de nuevas instancias, observándose una relación directa entre dichas funciones de red y niveles de productividad bajos. En consecuencia, se consideran dos métricas para su evaluación: el número de funciones virtuales instanciadas por observación (Y), y su productividad (Z), siendo Z el conjunto $Z = \{z_1 \cdots z_Y, Y \geq 0\}$ que define la productividad de las diferentes instancias virtuales para una determinada observación t. Análogamente al caso de W-EDoS, estas son monitorizadas en el tiempo, dando lugar a la generación de las siguientes series temporales:

$$Y_{t=0}, Y_{t=1}, \cdots, Y_{t=n}; (Y^n_{t=0}) \quad (6)$$

$$Z_{t=0}, Z_{t=1}, \cdots, Z_{t=n}; (Z^n_{t=0}) \quad (7)$$

donde una observación $0 \leq t \leq n$ es potencialmente maliciosa cuando Y_t presenta un crecimiento significativo y $Z_t = \{z_1, \cdots, z_{Y(t)}\}$ contiene un grupo de instancias que muestran claramente baja productividad, y por lo tanto sospechosas de hacer un consumo indebido de recurs; siendo este subconjunto de instancias las que explican el crecimiento anómalo de Y_t.

Comportamientos inesperados I-EDoS. Tal y como ocurre en los ataques W-EDoS, se observa un crecimiento significativo del número de instancias Y cuando para un horizonte m el error calculado entre su valor pronosticado $\hat{Y}_{t=n+m}$ y su observación en $Y_{t=n+m}$ se encuentra fuera del intervalo de predicción definido en ec. 5. Cuando una acción de autoescalado ha generado la creación de nuevas instancias con productividad $Z_t = \{z_1, \cdots, z_{Y_t}\}$ es posible detectar si parte de aquellas están involucradas en un ataque I-EDoS mediante la aplicación de un criterio de agrupamiento basado en densidad; en particular, a través del algoritmo de Agrupación Espacial Basada en Densidad de Aplicaciones con Ruido o DBSCAN (del inglés *Density-Based Spatial Clustering of Applications with Noise*) [37]. Este algoritmo considera la existencia de grupos de observaciones basados en la densidad de sus K-vecinos más cercanos, donde las observaciones que no son alcanzables dentro de un mismo grupo se consideran valores atípicos [38]. Se ha elegido DBSCAN por ser tolerante al ruido y no requerir la estimación previa del número de grupos a conformar, habiéndose configurado por la aproximación heurística recomendada en [39]. DBSCAN es ejecutado en cada conjunto de valores de productividad $Z_t = \{z_1, \cdots, z_{Y_t}\}$, y su resultado es un conjunto de K clusters representados por $C_t = \{c_1, \cdots, c_k\}$. Sea $Z_t = \{z_1, \cdots, z_{Y_t}\}$ el conjunto de medidas de productividad de las instancias en t, clasificados como $C_t = \{c_1, \cdots, c_k\}$ con $K \geq 0$ y ordenados como $s(C_t) = [c_1, \cdots, c_K]$, se presenta un comportamiento inesperado basado en instanciación (etiquetado como posible ataque I-EDoS en t) cuando se ha registrado un crecimiento significativo en el momento de creación de las instancias pertenecientes a c_1, agrupando c_1 las menos productividad.

V. Experimentos

Esta sección presenta el entorno de red en el cuál se han desarrollado los experimentos. La Capa Cloud ha sido complementada con los componentes de red autoorganizada (SON) destinados a la detección de ataques EDoS.

V-A. Escenario de Pruebas

El escenario de pruebas se ilustra en Fig. 3. La Capa Cloud se ha implementado con Openstack [40], que a su vez se ha desplegado en dos servidores: controlador (Controller) y cómputo (Compute). El controlador tiene configurados además el servicio de red (Neutron). El nodo de cómputo cuenta con los servicios de orquestación (Heat), clustering (Senlin) y telemetría (Ceilometer); sobre los que se configuran las políticas de autoescalado. Todos los servicios de Openstack están comunicados por un bus para el intercambio de mensajes (RabbitMQ). Por otra parte, los módulos de la capa autonómica SON combinan implementaciones propias con herramientas de código abierto para cada etapa de procesamiento. El nodo de recolección se encarga de recibir, en intervalos periódicos, los tiempos de respuesta calculados en cada instancia; mientras que las métricas de uso de las instancias son recopiladas por Ceilometer. Seguidamente, la agregación de datos tiene lugar, en primer lugar, con el cálculo de la entropía a partir de los datos del nodo central; y en segundo lugar, a través de consultas a la API de Ceilometer obteniendo el promedio de consumo de CPU de las instancias virtuales por unidad de tiempo. Las series temporales sirven como datos de entradas para los algoritmos implementados en la fase de detección. El conocimiento factual adquirido es utilizado en reglas de producción configuradas en Drools con el objetivo de inferir ataques EDoS.

V-B. Detección de Ataques W-EDoS

Se ha implementado en una instancia virtual Openstack un servicio web HTTP REST que soporta peticiones GET a siete URIs (numeradas de 1 a 7), cada una de las cuales genera tiempos de ejecución que varían, de la más simple a la más compleja, entre 18.56 y 36.73 ms. Una octava URI con 226.04 ms de tiempo promedio se implementa además para diferenciarla de las anteriores, representando el punto de mayor coste computacional que puede ser explotado como vulnerabilidad. Estas mediciones son recopiladas a cada segundo, y son usadas para calcular el grado de entropía de Rènyi. A su vez, se obtienen las medidas de CPU por instancia desde la API de Ceilometer, creándose una segunda serie temporal. En los experimentos se han lanzado peticiones desde 500 clientes, implementados como hilos de Python, que en situaciones de tráfico normal se comunican aleatoriamente con las URIs 1 al 8, mientras que en escenarios de ataque lo hacen únicamente a la URI 8. Para ambos escenarios se ha configurado una política de autoescalado que crea una nueva instancia del servicio web cuando el consumo de CPU promedio reportado es superior al 60 % en un periodo de un minuto. Dos factores de ajuste han permitido modificar la intensidad del ataque: el número de nodos comprometidos y la tasa de conexiones por segundo. Con los elementos descritos anteriormente, se han configurado las reglas de detección de comportamientos inesperados W-EDoS.

V-C. Detección de Ataques I-EDoS

En este escenario, la aplicación REST implementada ha sido modificada para que exponer una única URI que suponga un tiempo de ejecución promedio de 27.89 ms. Para alojar las instancias de imagen virtual se ha creado un cluster Openstack cuya longitud mínima es de 2 y la máxima es 12, aplicándose en él una política de autoescalado que crea una nueva instancia cuando el consumo de CPU promedio es superior al 80 %, mientras que elimina una instancia del cluster cuando este valor es inferior al 40 %. Se ha lanzado asimismo una prueba de esfuerzo sobre el servidor para establecer los niveles de productividad. Ésta ha sido evaluada con Httperf [41] y sus resultados reflejan la productividad más baja cuando la tasa de conexiones por segundo es inferior a 10, causando un consumo máximo de 39.1 % del CPU, que se aproxima al umbral más bajo de la política de autoescalado. Los niveles de rendimiento óptimos se registraron con una tasa de entre 10 y 40 conexiones por segundo, produciéndose un consumo promedio del CPU de entre 41.2 y 81.6 %. En todos los casos anteriores, el porcentaje de errores de conexión es del 0 %. Sin embargo, cuando se generó tráfico por encima de 40 conexiones por segundo el consumo de CPU alcanzó sus mayores niveles, registrándose valores entre el 82.7 y 99.6 % que superaron el umbral de auto escalado, además de errores de conexión superiores al 10 %. Los parámetros de red y la productividad resultante sirven para que DBSCAN identifique los grupos de instancias virtuales que por sus características podrían estar comprometidos en un ataque I-EDoS. Finalmente, la carga de trabajo a la que se ha sometido sigue una distribución aleatoria de Poisson [42] cuyo valor esperado λ es el número de conexiones del cluster en un instante de tiempo determinado, para lo cual se ha ensayado con tasas de entre 53 y 286 conexiones por segundo en un lapso de tres horas. Esta misma carga de trabajo se ha utilizado tanto en el escenario de tráfico normal como en el de ataque. En este último, el autoescalado se ha forzado a través de la manipulación de métricas recopiladas por Ceilometer, asumiéndose la capacidad del atacante de acceder al bus de datos RabbitMQ con las credenciales adecuadas, o su capacidad de explotar alguna vulnerabilidad como CVE-2016-9877 [43], la cual permite el envenenamiento de la información recolectada. De este modo se intercambian las lecturas de CPU originales (mensajes JSON) por otras con valores aleatorios entre 90 y 100 %. Estas métricas son las que finalmente se registran en la base de datos de Ceilometer, ocasionando el despliegue de instancias adicionales por una explotación fraudulenta de la política de autoescalado.

VI. Resultados

La efectividad de las estrategias de detección frente a ataques EDoS propuestas es descrita a continuación.

VI-A. Evaluación de detección de ataques W-EDoS

En Fig. 4 se observa la eficacia de la propuesta al variar el grado de la entropía de Rènyi. Los valores más bajos minimizan el impacto del ruido registrado, por lo que han demostrado arrojar resultados más precisos. Debido a esto, durante el resto de la experimentación se ha asumido el mejor ajuste logrado, $\alpha = 1$. Los ataques W-EDoS han sido calculados en intervalos de 1 %, 5 % y 10 %, donde el

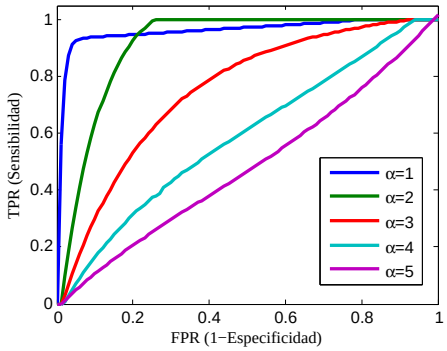


Figura 4: Eficacia al variar el grado de entropía.

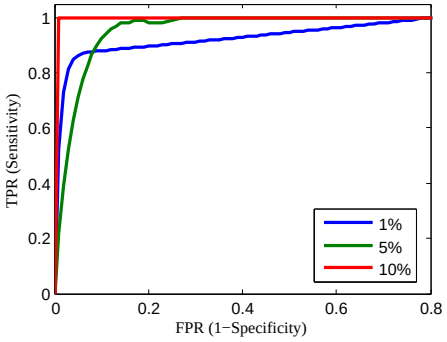


Figura 5: Curva ROC para 80px.

porcentaje representa la proporción de peticiones maliciosas o intensidad del ataque. Adicionalmente, cuatro escenarios han sido estudiados basados en el promedio de peticiones por segundo (px) generadas por los clientes: $50, 60, 70, 80$, donde K es el valor de ajuste para la creación de los intervalos de predicción. Se ha experimentado con distintos valores de K (en el rango de 0.1 y 6 con incrementos de 0.1), siendo este el parámetro que varía el grado de sensibilidad de la detección. Los mejores resultados se obtienen cuando la tasa de peticiones es de 80px y la intensidad del 10 % (Fig. 5), siendo la aproximación trapezoidal del Área Bajo la Curva (AUC) de un 0.995, con una Tasa de Verdaderos Positivos (TPR) de 1 y una Tasa de Falsos Positivos (FPR) de 0.01. En el caso opuesto, los peores resultados se registran con una tasa de 60px e intensidad del 1 %, con un AUC del 0.901, TPR de 0.816 y la FPR de 0.15. A partir de estos resultados es posible concluir que, a medida que la intensidad se hace más visible y la tasa de peticiones se incrementa, la precisión del sistema mejora, debido a que esas condiciones propician variaciones más notorias del grado de entropía y sobrecarga del CPU. En términos generales, la precisión obtenida demuestra la capacidad del método propuesto para detectar ataques W-EDoS en escenarios similares al considerado para su evaluación.

VI-B. Evaluación de detección de ataques I-EDoS

Los ataques I-EDoS han sido evaluados también en función de la intensidad registrada, cuyo impacto se traduce en un crecimiento del 10 %, 20 %, 30 % 40 % y 50 % del número de funciones virtuales instanciadas. Como puede suponerse, la intensidad de los ataques I-EDoS han influenciado directamente en la capacidad de detección del método propuesto. La Fig. 6 muestra la curva ROC obtenida en los distintos

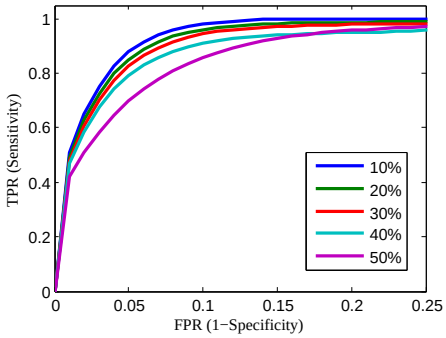


Figura 6: Curvas ROC en detección I-EDoS.

escenarios de experimentación. En términos generales, la tasa de acierto (TPR) ha experimentado pequeñas variaciones. En el primer grupo de ataques (10 %, 20 %, 30 %, 40 %), se ha observado una distancia del 0.022 (0.025 %) entre la mínima tasa de aciertos (TPR=0.89 y 10 % de intensidad) y la mejor tasa (TPR=0.91 en 40 %). Asimismo, cuando el ataque incrementa su intensidad (50 %) la tasa de aciertos se incrementa ligeramente (TPR=0.94). Sin embargo, tomando en cuenta el porcentaje de falsos positivos, los resultados fueron más significativos; en particular, si el ataque presenta una intensidad más baja, el método de detección ha registrado hasta un FPR=0.12 (al 10 %); pero al incrementarse la intensidad, la mejor configuración (al 40 % y 50 %) ha registrado un FPR=0.07, lo cual representa una mejora del 58.3 % sobre el peor resultado. Este patrón puede observarse en la Fig. 6 donde el área bajo la curva varía según la intensidad del ataque, siendo AUC=0.9811 en el mejor caso y AUC=0.9483 en el peor. Las variaciones de la efectividad se deben a la etapa de agrupamiento (*clustering*) por productividad. Cuanto más visible es el ataque, mayor es el número de instancias creadas que pertenecen al grupo de instancias poco productivas. En vista de los resultados obtenidos, puede concluirse que la estrategia propuesta es capaz de detectar ataques I-EDoS satisfactoriamente en escenarios similares al considerado para su evaluación.

VII. Conclusiones

El problema de los ataques de denegación de sostenibilidad (EDoS) en el escenario de redes autoorganizadas ha sido estudiado y definido desde dos perspectivas: carga de trabajo (W-EDoS) e instanciación (I-EDoS). En consecuencia, dos estrategias de detección han sido propuestas a lo largo de este artículo, una para cada una de ellas. Ambas se fundamentan en el modelamiento del comportamiento normal del sistema protegido y el descubrimiento de actividades anómalas. Para la detección de ataques W-EDoS se ha considerado el estudio de los errores de predicción significativos, tanto en el consumo de CPU como en la entropía estimada sobre los tiempos de respuesta a nivel de aplicación calculados en las instancias VNF. Por otra parte, para la detección de ataques I-EDoS se han analizado las relaciones entre la instanciación de funciones de red poco productivas y el crecimiento sospechoso del número de instancias desplegadas. Su efectividad ha sido comprobada a través de la experimentación llevada a cabo, en la cual se ha estudiado la variación de distintos parámetros de ajuste, como la intensidad del ataque, los intervalos de

confianza de los umbrales adaptativos, o el grado de entropía adoptado. Consecuentemente, ha sido posible demostrar que la propuesta cumple los objetivos de detección sobre el escenario de prueba implementado. No obstante cabe destacar que algunos aspectos necesarios para su despliegue en casos de uso reales no han sido abordados en el presente trabajo, como por ejemplo su fortalecimiento frente a métodos de evasión, o la inclusión de políticas de protección de datos, trazándose así diferentes líneas de trabajo futuro.

AGRADECIMIENTOS

Los autores agradecen la financiación que les brinda el Programa Marco de Investigación e Innovación Horizonte 2020 de la Comisión Europea a través del proyecto SELFNET: *Framework for Self-Organized Network Management in Virtualized and Software-Defined Networks* (H2020-ICT-2014-2).

REFERENCIAS

[1] NGMN Alliance. 5G White Paper. https://www.ngmn.org/uploads/media/NGMN_5G_White_Paper_V1_0.pdf

[2] Expert Working Group on 5G: Challenges, Research Priorities, and Recommendations. European Technology Platform for Communications Networks and Services (NetWorld2020 ETP). https://networld2020.eu/wp-content/uploads/2014/02/NetWorld2020_Joint-Whitepaper-V8_public-consultation.pdf.

[3] L.I. Barona López, A.L. Valdivieso Caraguay, M.A. Sotelo Monge, L.J. García Villalba. "Key Technologies in the Context of Future Networks: Operational and Management Requirements", *Future Internet*, Vol.9, No. 1, 2016.

[4] EU SELFNET Project "Self-Organized Network Management in Virtualized and Software Defined Networks". Project reference: H2020-ICT-2014-2/671672. Funded under: H2020. http://www.selfnet-5g.eu.

[5] A. A. Atayero, O. I. Adu, A. A. Alatishe, "Self organizing networks for 3GPP LTE", Proc. *Int. Conference on Computational Science and Its Applications*, pp. 242-254, 2014.

[6] 3GPP TS 32.500, "Self-Organising Networks (SON): Concepts and requirements", https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2031.

[7] M. Mao, J. Li, M. Humphrey. "Cloud auto-scaling with deadline and budget constraints", Proc. *11th IEEE/ACM International Conference on Grid Computing (GRID)*, pp. 41-48, 2010.

[8] P.S. Bawa, S. Manickam, "Critical Review of Economical Denial of Sustainability (EDoS) Mitigation Techniques", *Journal of Computer Science*, 11(7):855-862, 2015.

[9] C. Hoff, "Cloud Computing Security: From DDoS (Distributed Denial Of Service) to EDoS (Economic Denial of Sustainability)", 2008. http://rationalsecurity.typepad.com/blog/2008/11/cloud-computing-security-from-ddos-distributed-denial-of-service-to-\\edos-economic-denial-of-sustaina.html.

[10] C. Hoff, "A Couple of Follow-Ups On The EDoS (Economic Denial Of Sustainability) Concept...", 2009, http://rationalsecurity.typepad.com/blog/edos/.

[11] R. Cohen. "Cloud attack: Economic denial of sustainability (edos)". http://www.elasticvapor.com/2009/01/cloud-attack-economic-denial-of.html.

[12] A. Bremler-Barr, E.Brosh, M. Sides, "DDoS attack on cloud auto-scaling mechanisms", Proc. *IEEE Conference on Computer Communications (INFOCOM 2017)*, pp. 1-9, 2017.

[13] G. Somani, M.S. Gaur, D. Sanghi, M. Conti, "DDoS attacks in cloud computing: Collateral damage to non-targets", *Computer Networks*, 109:157-171, 2016.

[14] P. Singh, S. Manickam, S. U. Rehman, "A survey of mitigation techniques against Economic Denial of Sustainability (EDoS) attack on cloud computing architecture", Proc. *3rd International Conference on Reliability, Infocom Technologies and Optimization (ICRITO)*, pp. 1-4, 2014.

[15] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, F. Tangm, "Discriminating DDoS Attacks from Flash Crowds Using Flow Correlation Coefficient", *IEEE Transactions on Parallel and Distributed Systems*, 23 (6):1073-1080, 2012.

[16] A.S. Bhingarkar, B.D. Shah, "A survey: Securing cloud infrastructure against edos attack". Proc. *International Conference on Grid Computing and Applications (GCA)*, pp. 16-22, 2015.

[17] S. Vivinsandar, S. Shenai, "Economic Denial of Sustainability (EDoS) in Cloud Services using HTTP and XML based DDoS Attacks". *International Journal of Computer Applications*, 41(20): 11-16, March 2012.

[18] K. Singh, P. Singh, K. Kumar, "Application layer HTTP-GET flood DDoS attacks: Research landscape and challenges", *Computers & Security*, 65:344-372, 2017.

[19] A. Singh, K. Chatterjee, "Cloud security issues and challenges: A survey", *Journal of Network and Computer Applications*, 79:88-115, 2017.

[20] J. Idziorek, M. Tannian, D. Jacobson. "Attribution of fraudulent resource consumption in the cloud". Proc. *5th IEEE International Conference on Cloud Computing*, pp. 99-106, 2012.

[21] K.J. Singh, K. Thongam, T. De, "Entropy-Based Application Layer DDoS Attack Detection Using Artificial Neural Networks", *Entropy*, Vol. 18(10), No. 350, 2016.

[22] M.N. Kumar, P. Sujatha, V. Kalva, R. Nagori, A.K. Katukojwala, M. Kumar, "Mitigating Economic Denial of Sustainability (EDoS) in Cloud Computing Using In-cloud Scrubber Service". Proc. *4th International Conference on Computational Intelligence and Communication Networks (CICN)*, pp. 535-539, 2012.

[23] M. Masood, Z. Anwar, S.A. Raza, M.A. Hur, "EDoS Armor: A cost effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments". Proc *16th International Multi Topic Conference (INMIC)*, pp. 37-42, 2013.

[24] J. Idziorek, M. Tannian, D. Jacobson. "Attribution of fraudulent resource consumption in the cloud". Proc. *5th IEEE International Conference on Cloud Computing*, pp. 99-106, 2012.

[25] N.M. Alenezi, M.J. Reed. "Uniform DoS traceback", *Computers & Security*, Vol. 45, No. 1, pp. 17-26, 2014.

[26] G. Yao, J. Bi, A. V. Vasilakos. "Passive IP Traceback: Disclosing the Locations of IP Spoofers From Path Backscatter", *IEEE Transactions on Information Forensics and Security*, 10 (3):471-484, 2015.

[27] K. Wang, M. Du, S. Maharjan, Y. Sun, "Strategic Honeypot Game Model for Distributed Denial of Service Attacks in the Smart Grid", *IEEE Transactions on Smart Grid*, 8(5):2474-2482, 2017.

[28] I. Ozcelik, R.R. Brooks. "Deceiving entropy based DoS detection", *Computers & Security*, Vol. 48, no. 1, pp. 234-245, 2015.

[29] G. Ditzler, M. Roveri, C. Alippi, R. Polikar, "Learning in Nonstationary Environments: A Survey", *IEEE Computational Intelligence Magazine*, Vol. 10, No. 4, pp. 12-25, 2015.

[30] S. Salah, G. Maciá-Fernández, J.E. Díaz-Verdejo, "A model-based survey of alert correlation techniques", *Computer Networks*, Vol. 57, No. 5, pp. 1289-1317, 2013.

[31] "ETSI GS NFV 002 V1.2.1: Network Functions Virtualisation (NFV); Architectural Framework". http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_NFV002v010201p.pdf

[32] Ceilometer measurements. https://docs.openstack.org/ceilometer/pike/admin/telemetry-measurements.html.

[33] M. Bhuyan, D.Bhattacharyya, J. Kalita,"An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection". *Pattern Recognition Letters*, 51(1):1-7, 2015.

[34] E. Gardner, D. Dannenbring, "Forecasting with Exponential Smoothing: Some Guidelines for Model Selection". *Decision Sciences*, 11(2):370-383, 1980.

[35] S. Makridakis, S. Wheelwright, S. Hyndman, "Forecasting Methods and Applications", John Wiley and Sons, New York, NY, US, 1998.

[36] R. J. Hyndman, A. B. Koehler, J. K. Ord, R.D. Snyder, "Prediction intervals for exponential smoothing state space models", *Journal of Forecasting*, Vol. 24, pp. 17-37, 2005.

[37] M. Ester, H.P. Kriegel, J. Sander, X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise", Proc. *Second International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pp. 226-231, 1996.

[38] V. Chandola, A. Banerjee, V. Kumar, "Anomaly Detection: A Survey", *ACM Computing Surveys*, vol. 41, issue 3, No. 15, 2009.

[39] E. Shubert, J. Sander, M. Ester, H.P. Kriegel, X. Xu "DBSCAN Revisited: Why and How You Should (Still) Use DBSCAN", *ACM Transactions on Database Systems*, Vol. 42(3), No. 19, 2017.

[40] Open Source Software for Creating Private and Public Clouds. https://www.openstack.org.

[41] C. Barakat, P. Thiran,G. ; Iannaccone, C. Diot, P. Owezarski, "Modeling Internet backbone traffic at the flow level". *IEEE Trans. Signal Process.*, 51: 2111–2124, 2003.

[42] The Httperf HTTP load generator. https://github.com/httperf/httperf.

[43] CVE-2016-9877. https://www.cvedetails.com/cve/CVE-2016-9877/.

MITHRA: Multi-distributed Intelligence towards Higher Resilience Assets

Juan F. García
RIASC, Univ. de León
León, España
jfgars@unileon.es

José Manuel Martínez
RIASC, Univ. de León
León, España
jmartg@unileon.es

Adrián Sánchez
RIASC, Univ. de León
León, España
asanca@unileon.es

Miguel V. Carriegos
RIASC, Univ. de León
León, España
miguel.garriegos@unileon.es

Resumen- Las Redes Definidas por Software (SDN) están hoy asentadas en la industria IT. A los riesgos y amenazas de seguridad habituales en cualquier infraestructura de red hay que sumar los específicos de esta tecnología. En este trabajo presentamos el proyecto MITHRA (Multi-distributed Intelligence towards Higher Resilience Assets), un sistema que tiene como objetivo aumentar la resiliencia y seguridad de este tipo de redes. Actualmente se encuentra en estado de prototipo y es capaz de responder a tres tipos de amenazas bien definidas: Ataque de denegación de servicio (DoS) generales y específicos SDN y ataques *Man in the Middle* (MitM).

***Index Terms*- seguridad, SDN, MITHRA**

Tipo de contribución: Resumen extendido: proyectos que están comenzados, desarrollo de herramientas o software

I. INTRODUCCIÓN

Las redes informáticas están constituidas esquemáticamente por dos capas llamadas Plano de Datos (que forma el camino de los datos y establece las comunicaciones efectivas) y Plano de Control (que forma el sistema de programación, mantenimiento y monitorización de la propia red).

Los dos planos suelen estar físicamente juntos en los dispositivos que conforman las redes como switches y routers.

Software Defined Network (SDN) es una arquitectura para el control de redes informáticas en la que el plano de datos se separa del plano de control. Este último reside en lo que se conoce como Controlador de red SDN [1].

Las Redes Definidas por Software, SDN, y la tecnología que las implementan están hoy asentadas en la industria IT. Se considera una tecnología preparada para producción y como tal está prestando servicio en múltiples instalaciones, siendo su uso para despliegue de IaaS o SaaS uno de los más interesantes.

Sin embargo, a los riesgos y amenazas de seguridad habituales en cualquier infraestructura de red hay que sumar los específicos de esta tecnología [2][3].

El proyecto MITHRA (*Multi-distributed Intelligence towards Higher Resilience Assets*) tiene como objetivo aumentar la resiliencia y seguridad de las redes SDN.

II. ARQUITECTURA

MITHRA es un sistema inteligente y distribuido de detección y respuesta ante ataques e incidentes. El sistema se apoya y se centra para su desarrollo y operatividad en:

- SDN y en el concepto de controlador centralizado que conoce el estado completo de la red y es capaz de actuar o ser consultado sobre ella a través de protocolos conocidos.
- Despliegue sobre las redes anteriores de un sistema de agentes (software o hardware) intercomunicados capaces de analizar el estado de la red y detectar patrones en tiempo real para actuar de forma adecuada ante fallos de seguridad. La pieza central del sistema es un módulo orquestador.
- La red de agentes añade una capa de inteligencia a la red de forma que esta es capaz por si sola de prevenir determinados problemas que impacten en la operatividad o seguridad de la misma y de reaccionar cuando dichos problemas ya han aparecido para recuperar el servicio.
- El análisis de patrones, la correlación de logs y la realimentación por parte del operador humano son parte fundamental de los procesos de toma de decisiones del sistema para actuar sobre la red.
- El objetivo es que la red de agentes sea capaz de responder ante cualquier tipo de problema, real o potencial, que se presente en la misma.

En [5] puede verse el diagrama conceptual inicial del sistema (en concreto se detalla la detección y respuesta frente a un ataque DoS, ver apartado III.a).

El sistema presenta cierta similitud conceptual con OrchSec [4], pero no utiliza monitorización exhaustiva de la red (por los problemas de rendimiento) ni se restringe a actuar sólo sobre el controlador como mecanismo de mitigación.

Actualmente disponemos de un prototipo modular y extensible a integrar nuevas funcionalidades: Para detectar y mitigar los ataques de forma incremental hemos creado un sistema extensible que pueda ir abordando cada problema de seguridad uno por uno e independientemente del resto, de forma que se puedan ir añadiendo nuevos a medida que se desarrollan o mejorando los ya existentes.

Cada problema de seguridad a abordar es tratado por uno o varios **agentes** de software, que trabajan de forma específica en la detección del problema y en su mitigación.

El punto central en el sistema se conoce como **orquestador**, y se encarga de controlar a todos los **agentes** que hay funcionando en la red, reconfigurarlos si es necesario, intercomunicarlos y notificar al dueño/operador de la red. Su función es la de informar al operador y enlazar agentes detectores con agentes ejecutores.

Para cada problema de seguridad o amenaza (vg.- que una máquina virtual de un *tenant* está participando en un DoS contra otro *tenant*) se implementa uno o varios **agentes detectores o recolectores** capaces de detectar ese comportamiento (siguiendo con el ejemplo, el agente identifica el origen del problema - la máquina(s) atacante(s)) y lo notifica al orquestador).

([1]) Un '*tenant*' es *cada uno de los entornos virtuales alojados en la*

infraestructura.

El orquestador, en base a una serie de reglas configurables y establecidas por el operador, activa a un segundo agente conocido como **agente ejecutor o mitigador** que aislará la VM atacante desconectándola de la red. Esto se consigue enviando una orden al NorthBound-API del controlador de la SDN para que no permita tráfico de esa máquina virtual o red del *tenant* y para que corte todo el tráfico con destino a la máquina atacada.

III. TECNOLOGÍA Y AMENAZAS MITIGADAS

MITHRA se sustenta en una nube All-in-one (AIO) de OpenStack (MITAKA) con infraestructura de red SDN controlada por OpenDaylight (ODL).

Para el orquestador se ha utilizado Nagios, (software de gestión de alertas en redes) modificado y configurado para aceptar los mensajes de los agentes detectores como alertas. En respuesta a estas alertas, activa los agentes mitigadores.

A fecha de redacción de este trabajo, el núcleo de MITHRA (la red de comunicación de agentes, orquestador e infraestructura cloud-SDN subyacente) es plenamente operativo, y el sistema es capaz de responder a tres tipos de amenazas: Ataque de denegación de servicio (DoS) con origen dentro de la red, generales y específicos SDN y ataques *Man in the Middle* (MitM) que afectan al controlador.

Para validar el sistema, hemos utilizado varias FPGA, Raspberry Pi y software específico que emula los ataques.

A. Ataques DoS generales

MITHRA detecta y responde a ataques DoS generales que afectan a redes clásicas, y por tanto también a SDN.

El agente recolector es un IDS (Snort) adapatado que analiza tráfico en las interfaces públicas de los *tenants*. El agente se ejecuta a su vez en una VM del *tenant* de gestión, con recursos controlados. Mediante port-mirror el tráfico público se copia a este agente.

El agente detecta los ataques DoS y envía una alerta al orquestador (Log + RSyslog + nrpe) con información específica de donde se está el origen del problema.

Al recibir el mensaje, el orquestador se comunica con el agente ejecutor correspondiente. Este agente es un software en el propio orquestador que se conecta al controlador SDN y le ordena desactivar la interfaz virtual de la máquina atacante.

El orquestador es el encargado de guiar al agente ejecutor sobre la interfaz que debe desactivar en función de la información que le pasó el agente detector.

Este flujo de acciones aparece ejemplificado en el diagrama [5] anteriormente indicado; en [6] puede verse un vídeo del sistema MITHRA reaccionando a este ataque.

B. Ataques MitM específico SDN.

El segundo vector de ataque es específico SDN. Consiste en detectar que desde la red de algún *tenant* está intentando interceptar el tráfico de configuración (Northbound API) del controlador.

El agente detector de este comportamiento consiste en un software de detección de ARP-Poisoning (ARPON), que es una de las formas habituales de ejecutar MITM. El agente se ejecuta en el mismo controlador SDN, analizando las tablas ARP y ejecutando un algoritmo para identificar si se está intentando confundir a la pila de red del controlador de aceptar paquetes con origen falseado. Si detecta un intento de envenenamiento, puede bloquear las comunicaciones sospechosas o, en este caso, enviar una alerta al orquestador indicando la detección del problema y la información de que disponga sobre el origen del mismo.

El orquestador recibirá la alarma y según las reglas establecidas por el operador envía una orden al agente mitigador para tratar el problema.

En este caso el agente ejecutor es el propio agente recolector, que recibe la comunicación del orquestador y bloquea todos los paquetes sospechosos, además de alertar al usuario.

C. Ataques DoS específico SDN

El tercer vector de ataque es un caso mixto donde se protege un elemento específico SDN (el controlador) de un tipo de ataque no específico de SDN (DoS).

Se aprovecha gran parte del desarrollo del primer caso. El agente detector es el mismo, pero el orquestador lo maneja activando un agente ejecutor diferente.

En este caso el agente ejecutor se conecta al controlador y lo reconfigura para que en la red no se permita el tráfico (*flow*[2]) sospechoso entre el atacante y el atacado. Es una prueba para determinar que no solo es posible aislar la máquina atacante de forma completa, sino que es posible filtrar solo el tráfico problemático.

([2]) Un '*flow*' es una comunicación en la red identificada de forma inequívoca por al menos origen (dirección, puerto), destino (dirección, puerto) y protocolo.

IV. CONCLUSIONES Y TRABAJO FUTURO

MITHRA es un sistema de seguridad específico para redes SDN, que tiene en cuenta los problemas de seguridad de las redes clásicas además de los específicos de SDN. El sistema es modular y escalable, y se encuentra actualmente en fase de prototipo. Actualmente funciona contra ataques DoS generales, así como DoS y MitM específicos SDN.

A lo largo de 2018 esperamos tener la primera versión de MITHRA disponible para el público.

AGRADECIMIENTOS

Esta investigación ha sido parcialmente financiada por el Instituto Nacional de Ciberseguridad (INCIBE).

REFERENCIAS

[1] Diego Kreutz, Fernando M. V. Ramos, Paulo Esteves Verı´ssimo, Christian Esteve Rothenberg, Siamak Azodolmolky, andSteveUhlig,M: "Software-Defined Networking: A Comprehensive Survey", Proceedings of the IEEE, vol. 103, n. 1, 2015.

[2] Sandra Scott-Hayward, Sriram Natarajan, and Sakir Seze: "A Survey of Security in Software Defined Networks", IEEE Communications Surveys & Tutorials, vol. 18, n. 1, 2016.

[3] Sandra Scott-Hayward, Gemma O'Callaghan and Sakir Sezer: "SDN Security: A Survey", Workshop on Software Defined Networks for Future Networks and Services, SDN4FNS 2013.

[4] Adel Zaalouk Rahamatullah Khondoker, Ronald Marx, Kpatcha Bayarou: "OrchSec: An Orchestrator-Based Architecture For Enhancing Network-Security Using Network Monitoring And SDN Control Functions", Network Operations and Management Symposium (NOMS), 2014.

[5] Diagrama conceptual MIHRA, riasc.unileon.es/archivos/multimedia/diagramaMithra.png

[6] Demostración MIHRA DoS, riasc.unileon.es/archivos/multimedia/MITHRA4_H264_12000.mp4

Extracción de Características en Big Data para la Detección de Anomalías en Ciberseguridad

José Manuel García Giménez*, Alejandro Pérez Villegas†, José Camacho Páez*
* Universidad de Granada
*Departamento de Teoría de la Señal, Telemática y Comunicaciones
CITIC
†4iQ Identity Threat Intelligence
Email: {jgarciag, josecamacho}@ugr.es* alejandro.perez@4iq.com†

***Resumen*—El volumen de datos a utilizar en el ámbito de la ciberseguridad está creciendo exponencialmente de la mano de las nuevas tecnologías. Esto está motivando la utilización de nuevas técnicas que permitan analizar todos estos datos de forma eficiente. La gestión de toda esta información es complicada por su disparidad en estructura y formato. En este trabajo se introduce una herramienta, el FCParser, que permite procesar datos basados en texto, reduciendo grandes volúmenes de información a observaciones adecuadas para su análisis con distintas técnicas de *machine learning*. La eficacia de la herramienta se ha evaluado con un experimento basado en un conjunto de datos de, que contiene dos fuentes de datos de seguridad parcialmente no estructuradas para la detección y diagnosis de anomalías. Este experimento se encuentra en una máquina virtual disponible públicamente para su reproducibilidad.**

***Index Terms*—Seguridad en red, Detección de intrusos, Big Data, Aprendizaje automático, Parseo, Diagnóstico.**

Tipo de contribución: *Investigación original*

I. Introducción

El volumen de datos que se puede obtener a partir de una red actual se ha visto incrementado de forma exponencial en los últimos años [1]. Todos estos datos tienen formatos dispares y estructuras distintas, por lo que es necesario buscar formas eficientes de gestionar y analizar tal cantidad de datos, como el *machine learning* [2] [3].

El crecimiento en el volumen de datos ha venido acompañado de un crecimiento similar en la cantidad y diversidad de los incidentes de ciberseguridad. El ingente volumen de eventos de seguridad que se pueden generar en un red corporativa es difícil de asumir por los gestores de seguridad. Por este motivo, en este campo siempre se buscan nuevas técnicas para establecer un sistema de priorización de eventos, que permitan reducir el tiempo de detección y respuesta, con el fin de minimizar el coste asociado a los incidentes. En este contexto, la monitorización estadística multivariante de red (MSNM, del inglés, Multivariate Statistical Network Monitoring) [4] es una técnica prometedora, ya que proporciona una buena capacidad de detección, a la par que suministra información relevante para la diagnosis de los eventos detectados.

Mientras que en otros sectores se trabaja con datos sencillos de procesar, los datos que se utilizan en ciberseguridad tienen estructuras y formatos muy dispares. Estos datos pueden provenir de distintas fuentes y dependiendo del origen de los mismos se pueden clasificar en datos de registros e información de estado, datos de tráfico y datos de seguridad. Asimismo, según su formato, se pueden clasificar en datos estructurados y no estructurados. En este trabajo se presenta una herramienta, el FCParser, para procesar datos basados en texto de cualquier estructura u origen, transformándolos a un formato adecuado para su análisis. Esta herramienta tiene dos módulos: el módulo principal para parsear los datos y otro módulo que permite reducir el tiempo de respuesta ante una anomalía simplificando el proceso de diagnosis.

El resto del artículo está organizado de la siguiente manera: En la sección II se pone en contexto el trabajo, explicando las técnicas sobre las que se construyen las herramientas desarrolladas. En las secciones III y IV se muestra cómo se han diseñado e implementado las herramientas en cuestión. En la sección V se presenta una prueba experimental donde se han evaluado las herramienta. Por último, a partir de los resultados obtenidos, se exponen las conclusiones en la sección VI.

II. Detección de anomalías

Las herramientas que se han desarrollado en este trabajo son extensibles a distintos tipos de técnicas de análisis de datos, como el *machine learning* (ML). El FCParser permite transformar los datos de distintos formatos y estructuras en una matriz adecuada para su análisis. Por otra parte, el *deparser* permite mejorar la interpretación de los resultados de diagnosis.

Aunque los dos módulos se pueden usar de forma independiente, su origen se acota en una metodología en 5 pasos para la detección de anomalías en *Big Data* para la ciberseguridad [5].

II-A. Metodología en 5 pasos

La metodología en 5 pasos para la detección de anomalías consiste en: *parsing*, fusión, detección, diagnosis y el paso opcional de *deparsing*.

1. *Parsing*: Consiste en transformar datos de red de diversas fuentes en observaciones numéricas adecuadas para su análisis. En el contexto del análisis multivariante basado en anásis de compoenetes principales (PCA, del inglés Principal Component Analysis) para la monitorización de red, Lakhina et al. [6] introdujo el uso de contadores acumulativos como características para la información de flujos de tráfico. A partir de esta idea, se ha llegado a una definición más general denominada feature-as-a-counter (FaaC) [7], que permite considerar cualquier fuente de datos en el análisis. La principal ventaja de esta estrategia de parseo hace que las características sean fácilmente interpretables. Por sus peculiaridades, esta tipo de enfoque está siendo utilizado

en otras herramientas comerciales para la detección de anomalías como X-Pack [8].

2. *Fusion*: La utilización de este tipo de características basadas en contadores acumulativos homogeneiza los datos de entrada, simplificando la tarea de combinarlos. De esta manera, el procedimiento de fusión consiste en concatenar las observaciones de las distintas fuentes de datos, teniendo en cuenta el tiempo de muestreo de cada una.
3. *Detección*: La detección es el procedimiento que identifica las observaciones que se consideran anómalas. En el ámbito de la ciberseguridad, es crucial que se pueda priorizar entre las distintos eventos anómalos para que se pueda responder con la mayor celeridad a los eventos más importantes.
   En este proyecto, para ilustrar la detección, se ha utilizado MSNM, ya que además de tener una buena tasa de detección, proporciona información relevante para el proceso de diagnosis.
4. *Diagnosis*: El proceso de diagnosis consiste en identificar las características relacionadas con las anomalías seleccionadas en el paso de detección. En el caso de MSNM se pueden utilizar algoritmos como los gráficos de contribución o el algoritmo oMEDA [9][10]. Este algoritmo permite descubrir las relaciones entre las observaciones y las características para encontrar las que afectan más a las *scores* de una observación o un grupo de observaciones frente a otras. A partir de esta información, se puede arrojar luz sobre el origen del comportamiento anómalo.
5. *Deparsing*: Este procedimiento es opcional ya que depende de las técnicas de detección y diagnosis utilizadas. Para la ejecución de este paso, se necesita información de las características asociadas a cada anomalía. Por ejemplo, en el caso de MSNM, la salida del algoritmo oMEDA. Si se cumplen las condiciones necesarias, consiste en extraer fragmentos de los datos de entrada que tengan que ver directamente con las anomalías detectadas a partir de la información que se obtiene en los pasos de detección y diagnosis. De esta manera el analista de seguridad puede interpretar los resultados de forma más rápida al revertir el proceso de *parsing* y volver a los datos en su formato original, reduciendo el tiempo entre detección y diagnosis y, por tanto, minimizando el coste asociado a una anomalía.
   En caso de que no se cumplan las condiciones de información de diagnosis, como no se tienen la lista de características asociada a cada anomalías, este quinto paso consistiría en extraer todos los datos relativos al tiempo de la observación que se detecta como anómala.

III. Parser

Los datos en bruto extraídos de una red no están preparados para su análisis, por lo que es necesario adaptar su formato al tipo de análisis que se vaya a realizar. Estos datos son muy dispares, por este motivo, existe la necesidad de crear herramientas que permitan procesar datos de diversas fuentes de forma sencilla. Este trabajo se centra en la creación de una herramienta que permita tratar cualquier información basada en texto para su estudio con distintas técnicas de análisis de datos como el *machine learning*, independientemente de su formato u origen.

Para este fin, se sigue la estrategia FaaC [7] que hace una representación numérica de los datos, transformando la información de entrada en una serie de contadores acumulativos que contienen la cantidad de veces que sucede un evento en un intervalo temporal, como por ejemplo la cantidad de veces que aparece el puerto 80 en una traza de tráfico. De forma adicional, la definición de características con este tipo de contadores hace más sencilla la interpretación de las propias características.

Hay sistemas comerciales que utilizan contadores similares a los descritos anteriormente para la detección de anomalías basadas en técnicas de ML como X-pack [8]. Este tipo de soluciones basan su funcionamiento en el almacenamiento de la información en bases de datos no relacionales, construyendo los contadores en base a peticiones a las bases de datos.

La herramienta propuesta en este trabajo, permite procesar los datos sin necesidad de almacenarlos en una base de datos. De esta manera se comprimen los datos a la vez que se procesan. Por otro lado, esta herramienta se plantea como un módulo de Python y no necesita el despliegue de ningún servicio, por lo que se integración con otros sistemas es simple.

III-A. Diseño

A partir de esta idea, se plantea el diseño de un programa configurable que permita procesar fuentes de datos basadas en texto estructuradas y no estructuradas, adaptándose a cada fuente sin necesidad de programar, apoyándose en la utilización de archivos de configuración distintos para cada fuente. Para este fin, se propone un procedimiento en cinco pasos: muestreo temporal, extracción de registros, agregación, transformación a observaciones y fusión de datos, como se muestra el diagrama de la Fig. 1.

Muestreo temporal: Consiste en dividir los datos de entrada en ventanas temporales de igual duración. A partir cada ventana se generará una observación con los contadores definidos.

Extracción de variables: Este es un paso intermedio para la metodología FaaC que consiste en representar los datos de entrada con una serie de variables. Las variables se definen como los campos que contienen la información necesaria para identificar los eventos. En la *Tabla* I se puede encontrar un ejemplo del registro de variables que se podría extraer a partir de la siguiente entrada:

Feb 27 00:56:43 bastion snort: [1:2000345:3] BLEEDING-EDGE IRC - Nick change on non-std port [Classification: A Network Trojan was detected] [Priority: 1]: TCP 11.11.79.67:1755 - 66.198.160.2:8888

Es importante mencionar que una entrada se define como cualquier texto entre separadores, siendo un separador cualquier cadena de caracteres. Comúnmente, en las fuentes de datos de red, el separador es un salto de línea o un doble salto de línea. No obstante, se puede utilizar cualquier otro patrón. De esta manera, utilizando esta definición flexible de entrada de datos, se podría procesar cualquier fuente de información basada en texto en la que se puedan identificar variables que contengan información relevante.

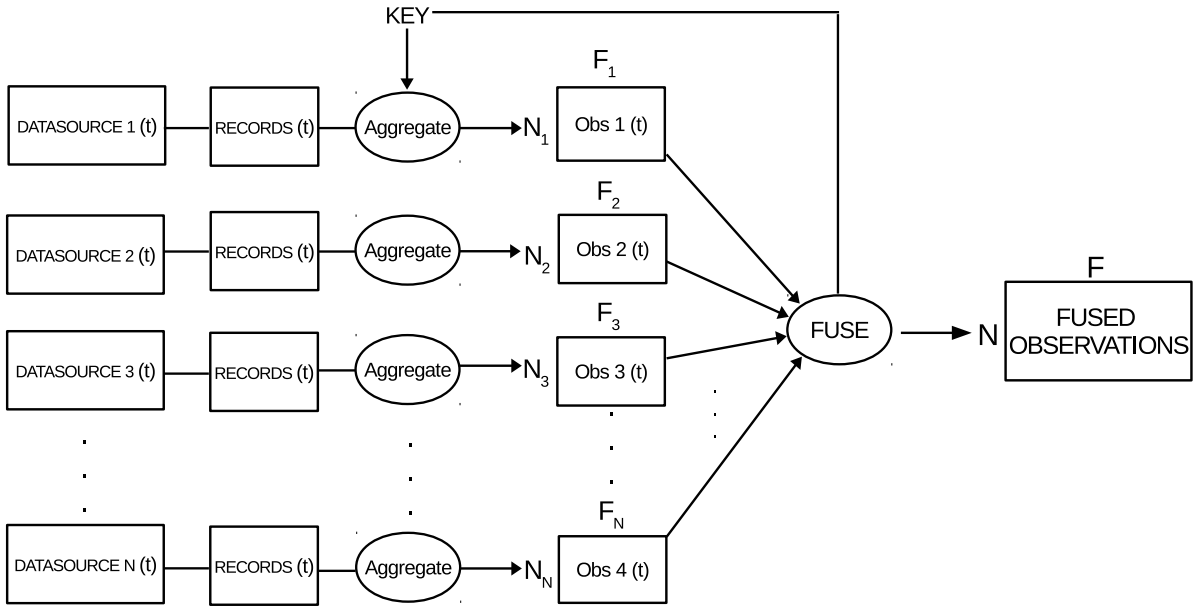


Figura 1. Diagrama de bloques parser

Tabla I
EJEMPLO EXTRACCIÓN DE VARIABLES A PARTIR DE UNA ENTRADA DE DATOS NO ESTRUCTURADA

Variable	Valor
Source ip	11.11.79.67
Protocol	TCP
Classification	A Network Trojan was detected
Destination IP	66.198.160.2
Label	BLEEDING-EDGE IRC - Nick...
Priority	1
Destination port	8888
Timestamp	Feb 27 00:56:43
Source port	1755

Agregación: Este proceso sirve para agrupar los datos de entrada en función de valores distintos de alguna variables para estratificar la información. La variable o variables que se eligen para este proceso se denominan *aggregation key*. Por ejemplo, si se elige la IP origen como *aggregation key*, se obtendrá una matriz de observaciones para cada IP distinta.

Transformación a observaciones: En este paso se transforman los registros a vectores de características contando los eventos. Por ejemplo, si la variable puerto destino vale 80, se aumenta el valor del contador de la característica asociada a conexiones a dicho puerto. A partir de cada variable, se diseñan varias características con los distintos valores posibles de dicha variable, por lo que a partir de cada registro se forma un vector de longitud igual al número de características. Este vector se denomina observación parcial y por su naturaleza, está formado solamente por unos y ceros, por ejemplo: *[0 1 0 0 0 1 1 0 0 1 0 0 1 0, ...]*.

No obstante, una observación completa corresponde a un intervalo de muestreo y no a un solo registro, por lo que es necesario agrupar las observaciones parciales provenientes de un mismo intervalo de tiempo. Debido a la naturaleza acumulativa de los contadores, se pueden construir las observaciones, simplemente, sumando las observaciones parciales. De esta manera, se consigue un vector que recoge la cantidad de apariciones de los eventos definidos en las características en el periodo de tiempo determinado. La observación, por tanto, es un vector de longitud igual al número de características del tipo: *[10 21 4 0 5 112 3 0 0 52 6 ...]*.

En el caso de que se realice el paso opcional de agregación, el procedimiento que se realiza es muy similar al descrito anteriormente. La única diferencia es que se tiene en cuenta la *aggregation key*, de manera que solo se suman las observaciones parciales de los registros cuya *aggregation key* sea igual, dando lugar a tantas observaciones como valores distintos posibles de dicha clave para cada intervalo de tiempo. Para un intervalo de timpo se podría tener una salida de este tipo:

'192.168.1.1': [4 32 1 0 0 12 43 0 20 2 7 ...]
'192.168.1.22': [2 1 0 2 12 1 3 0 4 2 71 ...]
'192.168.1.121': [1 0 0 0 0 2 3 0 0 12 27 ...]

Fusión. Este último paso consiste en combinar la información de las distintas fuentes de datos. Como la transformación en contadores unifica los tipos de datos de entrada, la fusión se simplifica en concatenar las observaciones con la misma marca temporal suponiendo un mismo intervalo de muestreo en todas las fuentes. Cuando el intervalo de muestreo no es igual, se pueden utilizar técnicas de *upsampling* o *downsampling* para unificar las frecuencias de muestreo.

III-B. Implementación

Para la implementación de todas estos procedimientos, se ha decidido crear una libreríaen Python, donde se implementan todas las funciones necesarias para aplicar FaaC a cualquier fuente de datos basada en texto. Partiendo de dicha librería se ha desarrollado una herramienta de *parsing* configurable y adaptable a partir de archivos de configuración, denominada FCParser. Esta herramienta realiza los 5 pasos descritos para procesar datos de red basados en texto. La configuración del programa a cada fuente de datos se hace definiendo las variables y características que se pretenden

extraer de dicha fuente. Tanto la librería como la herramienta se encuentran disponibles en un repositorio público [11].

Con el conjunto de variables y el conjunto de características de una fuente de datos, junto con algunos parámetros adicionales como el formato de marca de tiempo, una fuente queda totalmente definida y el FCParser podrá utilizarla como entrada para generar una matriz de observaciones adecuada para su análisis. Por este motivo es necesario detallar cómo se pueden definir las variables y las características.

Las variables, que son los campos que se pueden encontrar en las entradas de datos, quedan definidas con 3 parámetros: *name, matchtype* y *where*. *Name* es el nombre que se le da a la variable. *Where* es el parámetro que sirve para encontrar la variable dentro de una entrada de datos. En el caso de las fuentes estructuradas, es la posición del campo donde se encuentra la información. Por contrario, en las fuentes no estructuradas, es una expresión regular que sirve como patrón para encontrar el dato que se desea, aportando una gran flexibilidad. Por último *Matchtype* hace referencia al tipo de dato que contiene la variable y puede ser *string, number, regexp, IP, time, timedelta* o *múltiple*. En la *Tabla* II se puede ver un ejemplo de definición de variable con sus parámetros y unos posibles valores.

Tabla II
EJEMPLO DE DEFINICIÓN DE VARIABLES

Parámetro	Valor
Name	operation
Matchtype	string
Where	'(?<=Priority:)(?P<match >[0-9]+)'

Por otro lado, las caracterísitcas, que son los contadores acumulativos de eventos, se pueden definir con 4 parámetros: *name, variable, matchtype* y *value*. *Name* es el nombre que se le quiera dar a la característica, *variable* es el campo que hay que mirar para ver si se cumplen las condiciones para que suceda el evento en cuestión. *Value* es el valor que debe tomar la variables para que se considere que el evento ha sucedido. Por último, *matchtype* es el tipo de dato del campo *value* y puede ser *single, múltiple* o *range*, dependiendo si hay uno, varios o un rango de valores con los que comparar. En la *Tabla* III se muestra un ejemplo de definición de una características con los parámetros asociados.

En el repositorio donde está alojada la herramienta [11], se pueden encontrar algunos ejemplo completos de archivos de configuración de fuentes de datos de red que son utilizadas habitualmente: *netflow* e IDS recogido en servidor de *Syslog*.

Tabla III
EJEMPLO DE DEFINICIÓN DE CARACTERÍSTICAS

Parametro	Valor
Name	ids_prio2
Variable	priority
Matchtype	single
Value	2

Una vez se han generado los archivos de configuración de las fuentes de información, El procedimiento para *parsear* los datos consiste en los siguientes pasos:

1. Primero se comienza iterando por todos los archivos de cada fuente de datos.
2. Para cada archivo se itera por porciones de información necesarias para formar un registro. Si la fuente es estructurada, se itera de línea en línea. Por contrario, si la fuente es no estructurada, se itera por los fragmentos de información que hay entre cada delimitador.
3. Con la información seleccionada de construye un registro.
4. (Opcional) En caso de que se quiera hacer agregación, se separan los datos por la *aggregation key*.
5. El registro se transforma en una observación preliminar, comprobando si las variables tienen un valor tal como para que ocurran los eventos definidos en las características.
6. Las observaciones preliminares se van acumulando en una observación final.
7. Las observaciones finales a su vez se van agrupando en un diccionario en función de la fuente de la que provienen.
8. Por último se genera la salida del programa, escribiendo los ficheros de salida con las observaciones de las distintas fuentes fusionadas.

IV. DEPARSER

El *deparser* es un módulo de la herramienta FCParser que sirve para extraer los datos que tienen que ver directamente con las anomalías detectadas en su formato original. De este modo, se acota drásticamente la cantidad de información que debe revisar el analista de seguridad para encontrar el origen de la incidencia. Este módulo surge en el seno de la metodología en 5 pasos para el análisis de Big Data en el ámbito de la ciberseguridad y se fundamenta sobre la librería faaclib.

El *deparser* se ha diseñado para tomar como entrada la información de detección y la información de diagnosis, es decir, las marcas de tiempo y las características asociadas a una anomalía. Con esta información se busca revertir la metodología FaaC con los registros que tienen que ver con las anomalías, utilizando los mismos archivos de configuración que el módulo de *parsing*. El proceso consiste en buscar en los registros originales las características selecionadas. Esto se consigue tratando de extraer las características implicadas en los registros de los intervalos de tiempo que se obtienen en el paso de detección. Para este fin, se buscan en los registros las variables asociadas a las características y se comprueba si su valor es tal como para que uno de los eventos suceda. En caso afirmativo, el registro implicado se extraería.

Para priorizar la información que se extrae de cada anomalía, el programa genera como salida los registros asociados a las anomalías, ordenados por cantidad de características implicadas.

Dependiendo del volumen de datos con el que se esté trabajando, si se extraen todos los registros relacionados con alguna de las características, se puede tener demasiada información. Para limitar la cantidad de información que pasa al analista, se propone la utilización de un umbral máximo de registros a extraer por cada fuente de datos. A priori, entre dos registros con la misma cantidad de características implicadas, no se puede determinar cuál de ellas es la más relevante. Por este motivo, el umbral no puede ser un número absoluto de registros a extraer. Para utilizar el umbral, se forman grupos

indivisibles con los registros que contienen la misma cantidad de características implicadas. Entonces, se comprueba si el umbral se sobrepasa utilizando los bloques completos. Es decir, se toma el bloque de registros con mayor número de características y se comprueba que no se supere el umbral con el número de registros de dicho bloque. Si se supera el umbral, el proceso de *deparsing* finaliza. En caso contrario, se toma el siguiente bloque, se suma la cantidad de registros a los extraídos en el paso anterior y se vuelve a comprobar el umbral, y así sucesivamente hasta que se supere en umbral o se acaben los registros con alguna características implicada. Este límite superior de registros se computa para cada fuente de datos por separado.

Recapitulando, el procedimiento que se sigue en la implementación es el siguiente:

1. Se consigue la información referente a la anomalía y se extraen tanto la marca o marcas de tiempo en las que sucede, como las características implicadas.
2. Se buscan las fuentes de datos correspondientes a las características en cuestión.
3. Para cada registro se comprueba si la marca de tiempo coincide con las que se están buscando. En caso afirmativo, se comprueba la cantidad de las características que concuerdan con la anomalía. Este proceso de búsqueda es prácticamente igual al del proceso de *parsing* y depende de si la fuente es estructurada o no estructurada.
4. Se comprueba el umbral. El algoritmo es el siguiente: se busca el bloque de registros que tenga el mayor número de características implicadas. Si el número de registros del bloque es mayor que el umbral, esos son los registros que extraen. Por contrario, si no se llega al umbral, se toma el siguiente bloque, que tiene una característica implicada menos y se vuelve a comprobar el umbral. Este proceso se repite hasta que se alcanza el umbral, o se acaban los registros.
5. Se genera el archivo de salida con los registros relacionados con la anomalía.

V. Experimentación

Para probar y evaluar las herramientas desarrolladas se ha realizado un experimento donde se ha utilizado la metodología en 5 pasos para la detección de anomalías, usando las dos herramientas desarrolladas. En este caso, para el paso de detección se ha utilizado MSNM. Para la implementación de la parte de análisis multivariante, se ha utilizado la MEDA Toolbox [12]. Todo esto se ha implementado en una máquina virtual en la que se incluye tanto el conjunto de datos como las herramientas necesarias para el experimento [13]. Junto a este material, se puede encontrar toda la documentación necesaria para la reproducción de las pruebas.

Para la evaluación, se ha utilizado un conjunto de datos propuesto para un desafío de visualización para la ciberseguridad en el congreso internacional VAST (del inglés, *Visual Analytics Science and Technology*) [14]. En este caso se ha utilizado el desafío 2 del año 2012: *Bank of Money Regional Office Network Operations Forensics*. En este reto se presentan datos de seguridad de una red en explotación que ha tenido problemas de funcionamiento para tratar de arrojar algo de luz sobre lo que ha pasado en la red.

V-A. VAST 2012 MC2

El escenario del reto es una red bancaria en explotación, que cuenta con multitud de oficinas alrededor del mundo. Cada oficina tiene una gran cantidad de equipos funcionando a la vez. En total se estima que se cuenta con de un millón de máquinas [15].

Para el desafío con el que se está trabajando, MC2 (del inglés, *Mini Challenge 2*), se supone que una de las sucursales está experimentado fallos de rendimiento y problemas de *adware* y se debe intentar descubrir el origen de los mismos. Para la consecución de este objetivo, se proporcionan registros del sistema de cortafuegos (FW) y del sistema de detección de intrusos (IDS) de la red de esa sucursal. La porción de red a estudiar cuenta con unas 5000 máquinas.

Se ha elegido este conjunto de datos ya que presenta dos de las fuentes de datos de seguridad que se suelen encontrar, típicamente, en una red en explotación. Los datos que se presentan son parcialmente estructurados en su formato. Por un lado, los registros de FW vienen de *IPTables* y se estructuran en líneas con campos similares. No obstante, según la acción del FW que se registre, el formato varía considerablemente. Por otro lado, los registros del IDS vienen de *Snort* y constan de un número indeterminado de líneas y campos de tamaño variable. De esta manera, este experimento sirve para ilustrar la capacidad de analizar información de fuentes de datos de red de formatos dispares.

Como se trata de un reto pasado, ya resuelto, se dispone de información de lo que realmente ocurrió en la red: una *botnet* se introdujo en la red y se fue propagando hasta causar fallos de rendimiento [16].

V-B. Metodología de pruebas

A partir de este conjunto de datos se ha aplicado la metodología en 5 pasos para la detección de anomalías descrita anteriormente: *parsing*, fusión, detección, diagnosis y *deparsing*.

Parsing y fusión

A partir del conjunto de datos se ha realizado un archivo de configuración para cada fuente, donde se definen las variables y características. En total se extraen 305 características, 142 del FW y 163 del IDS como se presenta en las *Tablas* IV y V. Estos archivos de configuración son extensibles a otros conjuntos de datos del mismo tipo. No obstante, estos archivos están diseñados con este conjunto de datos en mente y las características están optimizadas para su análisis. Por este motivo, para otros conjunto de datos, aunque las variables si se maantendrían, es conveniente ajustar ligeramente las características dependiendo de los datos que se quieran analizar.

En este caso, a las dos fuentes de datos se les ha aplicado un periodo de muestreo de 1 minuto, obteniendo así 2345 observaciones. De esta manera, con ayuda de la herramienta FCParser, se han transformado los 4.2GB de datos (23762273 registros) en una matriz de 23345 x 305 con la información relevante para el análisis.

Detección

El proceso de detección se hace con MSNM utilizando la MEDA toolbox. En MSNM, la detección se fundamenta en la utilización de técnicas de reducción de dimensionlidad,

Tabla IV
CARACTERÍSTICAS FW

Cantidad	Tipo de característica
6	Tipos de IP origen y destino
103	Puertos origen y destino reservados o habituales
14	Mensajes ASA
7	Prioridad de syslog
2	Protocolo de transporte

Tabla V
CARACTERÍSTICAS IDS

Cantidad	Tipo de característica
6	Tipos de IP origen y destino
103	Puertos origen y destino reservados o habituales
4	Prioridad Snort
34	Clasificación Snort
10	Descripcción evento Snort
6	Información paquete IP

en este caso PCA [4]. La detección de anomalías se realiza monitorizando los estadísticos D-st y Q-st. Estos estadísticos resumen la información que contienen las características y se computan a partir de PCA [17] [18]. Además, se pueden establecer unos límites de control para estos estadísticos.

En el ámbito de la ciberseguridad, debido a la gran cantidad de incidencias que pueden ocurrir en un corto periodo de tiempo, generalmente se trata de priorizar los eventos. De esta manera, se concentra el esfuerzo de diagnosticar y solventar problemas en los eventos más prioritarios.

Las observaciones que tengan valores más altos de D-st y Q-st, serán las que más se alejan del modelo y, por tanto, las más anómalas. De esta manera, para decidir qué eventos se deben estudiar en profundidad, se han utilizado los percentiles de estadísticos mencionados para establecer un sistema de prioridad. A partir de estos valores, se determina que se analizarán las anomalías que los superen por un margen amplio. En las Fig. 2 y 3 se muestran los valores de estos estadísticos y los límites con los percentiles 95 y 99. Siguiendo esta estrategia, se seleccionan 9 observaciones que se consideran más prioritarias: la 370, 1413, 369, 1475, 389, 430, 1429, 1430, 1669 y 1671.

Diagnosis

A estas nueve observaciones, se les aplica el algoritmo oMEDA [9]. Este algoritmo extrae las características que más influyen en el valor de detección de cada observación, permitiendo arrojar luz sobre el origen de las anomalías. En la Fig. 4, se muestra el resultado de aplicar este algoritmo a la anomalía cuyo análisis es más prioritario según los estadísticos D-st y Q-st: la anomalía de la observación 370. En este gráfico se puede ver el subconjunto de características relacionadas con la anomalía. Este procedimiento se repite para las nueve observaciones y los resultados se recogen en la *Tabla* VI. En esta tabla se puede encontrar, para cada una de las 9 anomalías detectadas, las características que más afectan a que dichos eventos se consideren anómalos. En la tabla se aprecia que con estos resultados se puede comenzar a vislumbrar el origen de los incidentes. Sin embargo, con el siguiente paso, el *deparsing*, se lleva la diagnosis a otro nivel.

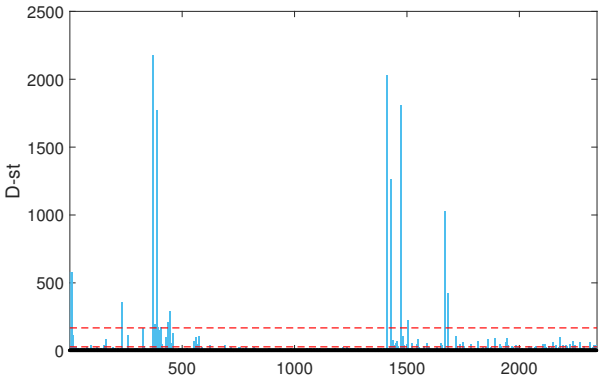


Figura 2. Estadístico D-st

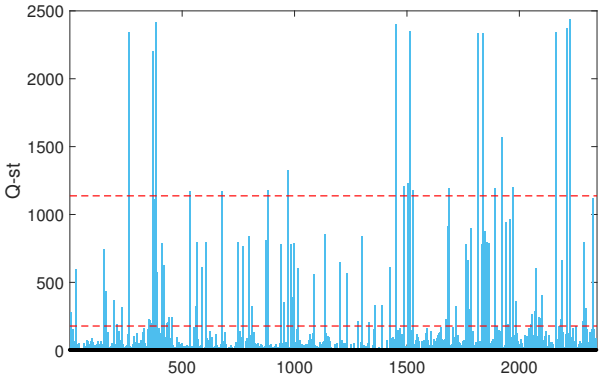


Figura 3. Estadístico Q-st

Tabla VI
CARACTERÍSTICAS IDS

Marca de tiempo	Características
06/04/2012 00:05	fw_dport_snmp fw_error fw_denyacl ids_ttl_low ids_snmp_req ids_brute_force ids_dport_pop3 ids_dport_imap4 ids_dport_snmp
06/04/2012 17:28	ids_prio1 ids_policy-violation ids_dns_update ids_sport_socks ids_dport_dns
06/04/2012 00:04	fw_dport_ssh fw_dport_telnet fw_error fw_denyacl ids_ttl_low ids_ssh_scan ids_ssh_scan_outbound ids_dport_ssh
06/04/2012 18:30	fw_sport_mssql fw_dport_ftp_control fw_asa_106023 fw_warning
06/04/2012 00:24	ids_ttl_low ids_prio2 ids_attempted-recon ids_successful-recon-limited ids_vnc_scan ids_sport_private ids_dport_register
06/04/2012 17:45-44	fw_asa_305009 fw_asa_305010 fw_notice
06/04/2012 21:44-46	fw_dport_multiplex fw_asa_405001

Deparsing

Como se observa en la *Tabla* VI, los datos de diagnosis no son fáciles de interpretar si no se tiene un amplio conocimiento de la red y del proceso de detección. Por esta razón, a partir de las marcas de tiempo de los eventos y de las características asociadas a los mismos, se realiza el proceso de *deparsing*. Con este procedimiento se consiguen extraer los registros originales que tienen que ver con las anomalías detectadas, revirtiendo el proceso de *parsing*. En la *Tabla* VII

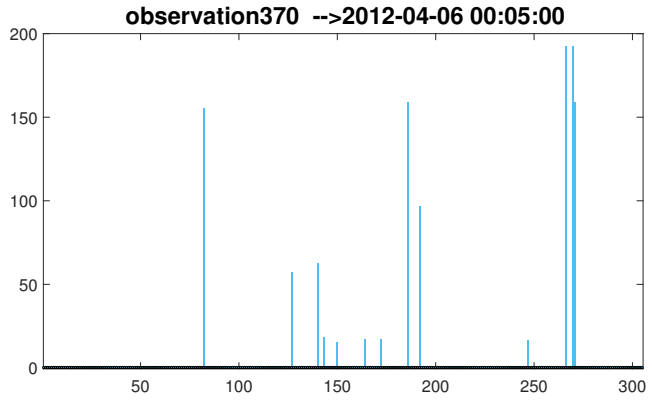


Figura 4. Gráfico oMEDA para la observación 370

se muestra un resumen de los resultados, donde se recoge el número de registros que se obtienen, los registros únicos y las conclusiones que se derivan a partir de los mismos. Con registro único se hace referencia a los distintos tipos de registro que se encuentran, ya que, a menudo, los registros aparecen repetidos varias veces en el intervalo de tiempo. Conociendo el registros y el número de repeticiones, se tiene la mayor parte de la información necesaria para inferir lo que ha pasado en un incidente.

En los instantes 06/04 00:04 y 00:05, se encuentran varios intentos de exfiltración de datos. Por una parte, en el registro de FW se encuentran muchas conexiones desde un equipo mediante protocolos Telnet y SNMP. Todas estas conexiones han sido bloqueadas por las listas de acceso del FW. Por otro lado, en el registros del IDS, se pueden encontrar muchas conexiones de SSH al exterior que, en este caso, sí pasan por el FW.

En el instante 06/04 00:24, en el registro del IDS, aparecen multitud de intentos de escaneo de puertos en busca de vulnerabilidades en los puertos habituales de VNC (del inglés, *Virtual Network Computing*).

En torno a 06/04 17:28, se puede observar un ataque al servidor de DNS con IP 172.23.0.10. En el registro del IDS aparece una gran cantidad de intentos de actualizar el servidor de DNS desde varios equipos. Si este ataque hubiera sido exitoso, podría ser el causante de la aparición de *adware* en los distintos equipos de la red, ya que podría traducir los nombres de dominio de forma maliciosa.

Alrededor del momento 06/04 17:45 se puede apreciar un cambio de configuración del FW. En el registro del FW se encuentran mensajes de como se hace un cambio de configuración en el firewall desde el usuario *enable_15* para activar SNAT (del inglés, *Source Network Address Translation*). De la solución del reto, se sabe que en torno al 17:30, los administradores de la red reiniciaron gran parte de los sistemas para tratar de solucionar los problemas de la red. Esta configuración del FW puede ser debido al reinicio de los sistemas.

En el instante 06/04 18:30, se pueden encontrar multitud de intentos de exfiltración de información en el registro de FW. En este caso, hay una gran cantidad de conexiones desde múltiples sistemas con el protocolo FTP. Sin embargo, estas conexiones son bloqueadas por las listas de acceso del FW.

Por último, en torno a 06/04 21:45, se observan algunas colisiones de ARP en el resgitro de FW.

A partir de los eventos detectados, se puede obtener una información general de lo que ha sucedido en la red. Se ven muchos tipos de intentos de exfiltración de datos y escaneos en busca de vulnerabilidades. Además, se aprecia el posible ataque al servidor de DNS en el instante 06/04 17:28.

Fijándose en los números, se puede ver que en total se han estudiado 1240 registros, esto supone tan solo un **0.005 %** de la cantidad total de registros. Además, de entre estos registros, solo se encuentran 16 tipos distintos, ya que la mayoría de las anomalías derivan en un cantidad reducida de registros. Por esto, realmente, con mirar 16 registros y el número de apariciones, se tienen indicios de lo que ha sucedido en la red con tan solo el **0.00007 %** de los registros. Esto supone una reducción masiva en la cantidad de información que el analista debe utilizar para diagnosticar un problema o evento, disminuyendo drásticamente el tiempo entre la detección y la diagnosis.

VI. Conclusiones

Se ha desarrollado una herramienta altamente configurable para procesar datos para su análisis, de manera que pueda servir para transformar datos heterogéneos basados en texto, sin necesidad de modificar el programa. Asimismo, se ha desarrollado el módulo de *deparsing* que, acotada en la metodología en 5 pasos para la detección de anomalías, lleva la diagnosis un paso más allá, reduciendo radicalmente la información que el analista utilizaría para encontrar el origen de un evento o incidencia. Estos dos módulos se han integrado en el paquete FCParser disponible públicamente en *github*.

La eficacia de la herramientas ha quedado patente con la experimentación basada en un conjunto de datos semi-estructurados de varias fuentes de seguridad, utilizadas habitualmente en el ámbito de la monitorización de red. A partir del conjunto de datos se ha efectuado la metodología en cinco pasos para la detección de anomalías encontrando los problemas de la red. Además, la realización de las pruebas en una máquina virtual, permite que el experimento sea reproducible.

Con el módulo de extracción de característica se han logrado transformar los 4.2 GB de datos de seguridad no estructurados en una matriz de 2345 x 305. Esta información se ha utilizado para la detección de anomalías con MSNM. A partir de los resultados, se ha comprobado que con el módulo de *deparsing* se consigue reducir la cantidad de información a que el analista tiene que inspeccionar en un 99.99993 %, pero con los datos obtenidos se ha podido inferir el origen de los problemas de la red, reduciendo el tiempo entre detección y diagnosis.

Agradecimientos

Este trabajo está parcialmente financiado por el Ministerio de Economía y competitividad y los fondos FEDER a través de los proyectos TIN2014-60346-R y TIN2017-83494-R y la Universidad de Granada, la Junta de Andalucía y el Fondo Social Europeo por la contratación de joven personal investigador, en el marco del sistema nacional de garantía juvenil y del programa operativo de empleo juvenil.

Tabla VII
DEPARSING DE LAS ANOMALÍAS DETECTADAS

Marca de tiempo	N registros	registros únicos	Descripción
06/04 00:04	101	4	Intento de exfiltración de datos e intentos de conexiones con SSH y Telnet
06/04 00:05	179	4	Intento de exfiltración de datos con el protocolo SNMP
06/04 00:24	80	1	Escaneo y intento ataque VNC
06/04 17:28	245	1	Posible ataque al servidor de DNS
06/04 17:44-45	11	4	Configuración de SNAT del router
06/04 18:30	622	1	Intento de exfiltración de datos por FTP
06/04 21:44-46	2	1	Colisión ARP

REFERENCIAS

[1] C. Systems, "White paper: The Zettabyte Era: Trends and Analysis," Cisco, Tech. Rep., 2010.

[2] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, "Intrusion detection by machine learning: A review," *Expert Systems with Applications*, vol. 36, no. 10, pp. 11 994 – 12 000, 2009.

[3] N. Ye, S. M. Emran, Q. Chen, and S. Vilbert, "Multivariate statistical analysis of audit trails for host-based intrusion detection," *IEEE Transactions on Computers*, vol. 51, no. 7, pp. 810–820, Jul 2002.

[4] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, and G. Maciá-Fernández, "PCA-based multivariate statistical network monitoring for anomaly detection," *Computers and Security*, vol. 59, pp. 118–137, jun 2016.

[5] J. Camacho, J. M. García-Giménez, G. Maciá-Fernández, N. M. Fuentes-García, and P. García-Teodoro, "Multivariate Big Data Analysis for Cybersecurity: 5 steps from the haystack to the needle," *Computers & Security*, 2018, submitted.

[6] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, p. 219, 2004.

[7] J. Camacho, G. Maciá-Fernández, J. Díaz-Verdejo, and P. García-Teodoro, "Tackling the big data 4 vs for anomaly detection," in *2014 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, April 2014, pp. 500–505.

[8] "X-pack," https://www.elastic.co/products/x-pack, [Online; accessed 15-Mar-2018].

[9] J. Camacho, "Observation-based missing data methods for exploratory data analysis to unveil the connection between observations and variables in latent subspace models," *Journal of Chemometrics*, vol. 25, no. 11, pp. 592–600, 2011.

[10] M. Fuentes-García, G. Maciá-Fernández, and J. Camacho, "Evaluation of diagnosis methods in pca-based multivariate statistical process control," *Chemometrics and Intelligent Laboratory Systems*, vol. 172, pp. 194 – 210, 2018.

[11] J. M. García-Giménez, A. Pérez-Villegas, and J. Camacho, "Fcparser," https://github.com/josecamachop/FCParser, 2017.

[12] J. Camacho, A. Pérez-Villegas, R. A. Rodríguez-Gómez, and E. Jiménez-Mañas, "Multivariate Exploratory Data Analysis (MEDA) Toolbox for Matlab," *Chemometrics and Intelligent Laboratory Systems*, vol. 143, pp. 49–57, apr 2015.

[13] J. M. García-Giménez and J. Camacho, "Maquina virtual: Experimento metodología 5 pasos," 2018, disponible en: https://nesg.ugr.es/veritas/index.php/demostrador-mbda.

[14] "Visual analytics community," retrieved from: http://www.vacommunity.org/VAST+Challenge+2017.

[15] "Vast challenge 2012," http://www.vacommunity.org/VAST+Challenge+2012.

[16] V. A. Community, "Vast 2012 mini-challenge 2 solution description for reviewers and committee," February 2012, http://www.cs.umd.edu/hcil/varepository/benchmarks.php#VAST2012.

[17] J. F. MacGregor and T. Kourti, "Statistical process control of multivariate processes," *Control Engineering Practice*, vol. 3, no. 3, pp. 403–414, 1995.

[18] J. E. Jackson and G. S. Mudholkar, "Control procedures for residuals associated with principal component analysis," *Technometrics*, vol. 21, no. 3, pp. 341–349, 1979.

Desarrollo de una metodología para la caracterización de *logs* heterogéneos e identificación de anomalías en sistemas de información

Gonzalo de la Torre-Abaitua
Instituto Nacional de Ciberseguridad
INCIBE
gonzalo.torre@incibe.es

Luis F. Lago-Fernández
Escuela Politécnica Superior
Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
luis.lago@uam.es

David Arroyo
Escuela Politécnica Superior
Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
david.arroyo@uam.es

***Resumen*—Los *logs* de sistemas son el principal mecanismo para identificar problemas y para detectar usos inapropiados de los recursos en una determinada infraestructura de información. Una característica de los actuales sistemas de información es la inclusión de una variedad cada vez más amplia de dispositivos y de modos de acceso a la información. Este factor deriva en un aumento de la heterogeneidad y complejidad de los eventos de seguridad que deben registrarse en los *logs*, lo cual puede constituir un serio obstáculo para el correcto análisis y categorización de la información contenida en tales registros de actividad. Es por ello que es necesario desarrollar metodologías que permitan la integración de *logs* con distinta codificación, y que habiliten la identificación de comportamientos anómalos con precisión y eficiencia. Esta comunicación resume los primeros resultados de nuestro trabajo en esta dirección, y presenta las líneas que se van a seguir para completarlo. En concreto, se discute una primera propuesta basada en la detección de anomalías mediante técnicas de compresión de información. En efecto, las distancias estadísticas basadas en la compresión de información permiten capturar la normalidad de un sistema sin necesidad de seleccionar parámetros más allá de los del algoritmo de clasificación utilizado. Gracias a esta premisa, hemos definido un mecanismo basado en la distancia de compresión normalizada (*Normalized Compression Distance - NCD*) y en máquinas de soporte vectorial (*Support Vectorial Machine - SVM*) que permite clasificar eventos de *logs* HTTP en normales o anómalos con la particularidad de que solo es necesario fijar los parámetros de *SVM*.**

***Index Terms*—IDS, Detección de anomalías, heterogeneidad de logs, NCD, Deep Learning**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

La movilidad y la diversidad de los medios de almacenamiento y de acceso a la información son rasgos inequívocos de la transformación digital que las Tecnologías de la Información y de las Comunicaciones (TIC) están experimentando. En efecto, estamos cada vez más habituados a usar la nube y, además, lo hacemos preferentemente a través de nuestro dispositivo móvil[1]. A esto hay que añadir la irrupción de la denominada *Internet de las cosas* (*Internet of Things* - IoT-). Así, el número de dispositivos de muy diversa índole que se encuentran conectados a Internet es muy elevado y se encuentra en aumento[2]. Esta creciente diversidad y complejidad en los modos de almacenamiento de la información genera un problema a la hora de gestionar su seguridad, ya que el número de *logs* crece a un ritmo similar o superior. Esto provoca que haya que procesar un gran número de estos *logs* de forma eficiente, tanto en el ámbito general de las TIC como en su aplicación en el contexto industrial [8]. Históricamente, este problema ha sido tratado mediante mecanismos de detección de intrusiones (*Intrusion Detection Systems - IDS*) [9]. No obstante, las técnicas inherentes a los IDS suelen necesitar un gran preprocesado de los datos y un entrenamiento previo para conseguir fijar los parámetros e hiperparámetros necesarios para ejecutar con éxito los algoritmos involucrados. Así mismo, la heterogeneidad en los datos almacenados en los diferentes *logs* genera la necesidad de definir mecanismos que permitan tratar cualquier tipo de datos de forma ágil y sencilla.

En este escenario, nuestro trabajo tiene por objeto la definición de un *framework* para capturar la heterogeneidad de los datos tratados en los *logs*, y para ayudar a los analistas a detectar anomalías. El primer paso de este proyecto consiste en el desarrollo de un marco para el tratamiento centralizado de información de relevancia en sistemas de gestión de la seguridad de la información. Esta fase ha sido parcialmente cumplimentada mediante el estudio de técnicas estadísticas basadas en compresión para la detección de anomalías en *logs* de tráfico web. Esta primera contribución ha de ser extendida con la comparación con otras técnicas aplicadas al mismo problema, el estudio de modelos de *Deep Learning (DL)* y la aplicación de ambos modelos a fuentes de datos heterogéneas. Así mismo, aparte de recolectar datos públicos, se pretende generar conjuntos de datos sintéticos para aplicarlos en los modelos desarrollados. Finalmente, se concretará un marco de análisis de modelos de DL en el contexto de la ciberseguridad analizando el uso de técnicas de evasión *Adversarial Machine Learning* (AML) [10] y sobre el análisis de ecuanimidad y objetividad algorítmica *Algorithmic Fairness* (AF) [11].

Todo este trabajo se engloba en una tesis doctoral con tres

[1]https://tinyurl.com/y84ejen5 (Servicios en la nube, INE). Último Acceso 08/03/2018.

[2]https://www.visioncritical.com/internet-of-things-stats/. Último Acceso 08/03/2018.

principales bloques que son detallados en la sección siguiente.

II. Metodología para la caracterización de logs heterogéneos e identificación de anomalías en sistemas de información

Para poder llevar a cabo la detección de anomalías en fuentes heterogéneas de datos, se va a partir del estado del arte actual mediante el empleo de técnicas estadísticas para, posteriormente, extenderlo con la aplicación de técnicas de DL y, por último, evaluarlo mediante técnicas de auditoría. Así, la investigación estará centrada en los tres grandes bloques que se detallan a continuación.

II-A. Marco para el tratamiento centralizado de información de relevancia para los sistemas de gestión de la seguridad de la información

Para el desarrollo de este marco se ha partido del estado del arte de los IDS y se ha propuesto un mecanismo para la detección de anomalías mediante distancias estadísticas basado en compresión de la información gracias a las propiedades y ventajas de la NCD [1]. Dicho mecanismo se basa en la información condicional comprimida y normalizada dada por la ecuación:

$$D(x,y) = \frac{C(y|x)}{C(y)} = \frac{C(xy) - C(x)}{C(y)} \quad (1)$$

Donde x e y son strings o ficheros, *C(x)* representa la longitud de x una vez comprimido y *C(xy)* representa la longitud, una vez comprimida, de la concatenación de x e y. El compresor utilizado ha sido gzip ya que se ha observado que obtiene mejor rendimiento en términos de velocidad[3] y mantiene las propiedades de simetría [2]. Mediante la ecuación (1) se calcula la diferencia entre una cadena de texto y un conjunto de cadenas de texto. En [4], nuestro objetivo ha sido aplicar la NCD al diseño de cortafuegos de nivel de aplicación en tráfico web. La intuición seguida es que la entropía de información variará al añadir una cadena de texto diferente al conjunto de datos considerado normal. La validación de esta hipótesis se ha efectuado considerando el dataset del CSIC[4], obteniéndose como mejor resultado un área bajo la curva ROC (AUC) de 0,975 y una tasa de acierto (*accuracy*) de 0,95.

Aunque estos resultados son similares a los de otros mecanismos propuestos en la literatura [3], el siguiente paso de nuestro proyecto será la comparación del método propuesto con otras técnicas previamente aplicadas al mismo problema. Además, se aplicará el modelo construido a otras fuentes de datos (e.g., información extraída de fuentes abiertas o en repositorios de vulnerabilidades). De esta forma, se persigue conseguir una primera aproximación a un IDS que opere con fuentes de datos heterogéneas.

II-B. Metodología para el ajuste de modelos de DL en contextos de análisis de datos de ciberseguridad

En este punto se pretende extender el estado del arte actual y el modelo de la sección II-A mediante la utilización de arquitecturas *DL*. Para ello, se va a estudiar el empleo de redes recurrentes [12], ya que tienen una gran utilidad en el análisis de secuencias y han obtenido resultados positivos en tareas de procesamiento de lenguaje natural. Así mismo, tienen unas propiedades inherentes que permiten una codificación automática de características y aprovechar las dependencias estructurales y espaciales existentes en los datos para conseguir construir modelos fiables. También se analizará el uso de otras técnicas como *Varational Autoencoders* o *Generative Adversarial Networks*. Con todo esto, se pretende inferir variables latentes en el volumen de datos de actividad en una red u organización compleja.

II-C. Marco de análisis de modelos de DL en el contexto de la ciberseguridad

Todo esquema integral para el tratamiento de eventos e incidencias de seguridad debe estar apoyado en una adecuada articulación de su ciclo de vida. Por ello, en el último estadio de nuestro proyecto se llevará a cabo una fase de verificación de los modelos desarrollados, tomando como base los estudios recientes sobre técnicas de AML frente a sistemas adaptativos de identificación de amenazas a la ciberseguridad [5][6] y las propuestas sobre AF [7]. Estos modelos son de especial relevancia hoy en día, dada la alta dependencia respecto a software de terceros y a tenor de las exigencias normativas de la regulación de protección de datos (General Data Protection Regulation - GDPR[5]).

[3]http://tukaani.org/lzma/benchmarks.html, Último Acceso 08/03/2018.

[4]http://www.isi.csic.es/dataset/, Último Acceso 08/03/2018.

[5]https://www.eugdpr.org/, Último Acceso 08/03/2018.

Agradecimientos

Este trabajo está adscrito a los proyectos CIBERDINE (S2013/ICE-3095) -Comunidad Autónoma de Madrid- y MINECO TIN2014-54580-R (Gobierno de España).

Referencias

[1] Vitányi, Paul M. B., Balbach, Frank J., Cilibrasi, Rudi L., Li Ming: "Normalized Information Distance", en *Information Theory and Statistical Learning*, pp. 45-82, 2009.

[2] Cilibrasi, R., Vitányi, P.M.B.: "Clustering by compression", en *IEEE Transactions on Information Theory* Vol. 51(4), pp 1523–1545, 2005.

[3] Nguyen, H.T., Torrano-Gimenez, C., Alvarez, G., Petrovic, S. y Franke, K.: "Application of the generic feature selection measure in detection of web attacks", en *Computational Intelligence in Security for Information Systems* vol. 6694, pp. 25–32, 2011.

[4] de la Torre-Abaitua, G., Lago-Fernández, L.F., Arroyo, D.: "A Parameter-Free Method for the Detection of Web Attacks", en *Advances in Intelligent Systems and Computing*, vol 649, pp 661-671, 2017.

[5] F. Tramér, N. Papernot, I. Goodfellow, D. Boneh, P. Mcdaniel: "The Space of Transferable Adversarial Examples", 2017.

[6] F. Tramér, A. Kurakin, N. Papernot, D. Boneh, P. Mcdaniel: "Ensemble Adversarial Training: Attacks and Defenses", en *ICLR*, 2018.

[7] B. Lepri, N. Oliver, E. Letouzé, A. Pentland, P. Vinck: "Fair, Transparent, and Accountable Algorithmic Decision-making Processes The Premise, the Proposed Solutions, and the Open Challenges, Philosophy & Technology", en *Philosophy & Technology*, 2017.

[8] Hahn, Adam: "Operational Technology and Information Technology in Industrial Control Systems", en *Cyber-security of SCADA and Other Industrial Control Systems*, pp 51-68, 2016.

[9] Kruegel, C., Valeur F., Vigna, G.: "Intrusion Detection and Correlation", en *Advances in Information Security*, vol 14, 2005

[10] Shoshitaishvili, Y., Weissbacher, M., Dresel, L., Salls, C., Wang, R., Kruegel, C., Vigna, G.: "Rise of the HaCRS: Augmenting Autonomous Cyber Reasoning Systems with Human Assistance" en *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp 347-362, 2017.

[11] Patki, N., Wedge, R., Veeramachaneni, K.: "The Synthetic Data Vault" en *DSAA*, 2016

[12] Hochreiter S., Schmidhuber, J.: "Long short-term memory", en *Neural Comput.*, vol. 9, pp. 1735–1780, 1997.

La intención hace el agravio: técnicas de clustering conceptual para la generalización y especialización de intencionalidades en el spear phishing

Iñaki Vélez de Mendizábal, Enaitz Ezpeleta, Urko Zurutuza
Mondragon Unibertsitatea
Loramendi 4, 20500, Mondragon
{ivelez, eezpeleta, uzurutuza}@mondragon.edu

David Ruano-Ordás
University of Vigo
Department of Computer Science
Campus As Lagoas, 32004, Ourense
drordas@uvigo.es

***Resumen*—La difusión de contenidos spam a través de los distintos servicios de Internet, es un problema que afecta a millones de usuarios al día y al que hay que añadir el spear phishing o spam dirigido.**

Este trabajo de investigación propone la creación de una plataforma configurable para el filtrado de spam basado en contenido, haciendo uso de información semántica (como las temáticas, la polaridad ó el análisis de sentimientos).

Se propone la identificación de nuevas técnicas para extraer información de polaridad y personalidad de los contenidos, además del diseño y aplicación de técnicas de clustering conceptual para la generalización y especialización de intencionalidades.

***Index Terms*—spam, spear phising, Procesamiento del Lenguaje Natural, PLN**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

La lucha contra el spam ha supuesto un reto para administraciones públicas, empresas e investigadores que han abordado el problema de distintas formas: (i) las legislaciones específicas, (ii) los filtros basados en características específicas del servicio de comunicación (distintas del contenido) y/o (iii) los filtros basados en contenido. Como conceptualmente el spam engloba a determinadas temáticas o intencionalidades (información semántica) que resultan de nulo interés para los usuarios, se considera que el uso del filtrado basado en contenido es la forma más efectiva de combatirlo. Actualmente la mayoría de filtros spam basados en contenido usan técnicas de aprendizaje automático empleando como entrada, características sobre la presencia o frecuencia de palabras concretas en los textos. El equipo de investigadores de este trabajo comparte la idea de que estos filtros pueden mejorarse añadiendo información semántica en todas las fases de la clasificación. Este trabajo de investigación en desarrollo pretende ahondar en la detección de la intencionalidad y/o las temáticas de los contenidos que se usarán como entrada en las técnicas de aprendizaje automático.

II. Estado del arte

En un escenario de conectividad permanente a través de Internet, cada internauta puede escoger la forma más adecuada para comunicarse. Sin embargo, las ventajas ofrecidas por estos servicios pueden ser empleadas con propósitos no éticos, como el envío de información publicitaria molesta sin el consentimiento de los destinatarios. En efecto, algunos ejemplos de estos usos inadecuados son el Spam 2.0 [1], el webspam [2], el spam por correo electrónico [3] o incluso el spam en SMS [4].

Un estudio reciente [5] muestra hechos concretos que permiten interpretar el alcance del problema, como el gran número de amenazas persistentes avanzadas (*advanced persistent threat*) detectadas, el alto porcentaje de spam (56,63 % en 2017[1]) o el aumento del ransomware distribuído a través de correo electrónico o las redes sociales.

A pesar de que existen alternativas concretas de filtrado de spam basadas en contenido como el uso de expresiones regulares o patrones, análisis estadísticos (en email [6], webspam [2], SMS [7], y spam 2.0 [8]), la mayoría de las propuestas en este ámbito se basan en el uso de algoritmos de aprendizaje automático (Machine Learning, ML) [9], [10] junto con mecanismos de reducción de la dimensionalidad (como la selección de características) [11].

La configuración de un filtro basado en contenido con algoritmos de aprendizaje automático comprende cuatro etapas distintas durante la fase de entrenamiento: (i) La extracción de información (habitualmente tokenización) de contenidos previamente etiquetados para entrenar el filtro, (ii) la selección de las propiedades (tokens) más relevantes para distinguir entre spam y legítimo usando un mecanismo de selección de características (habitualmente filtros, debido a su rapidez [12]), (iii) la representación de la información de los contenidos en forma de vectores de características de acuerdo con las propiedades seleccionadas en la etapa anterior y (iv) la construcción de un modelo de clasificación usando la técnica de ML seleccionada.

Los mecanismos de selección de características no han evolucionado significativamente y especialmente, no se contempla el uso de información semántica en ellos sino en la dependencia estadística que hay entre los valores de las propiedades y la clase real del contenido a clasificar (variable objetivo) [12]. Además, dado que la definición de spam tiene que ver con determinadas temáticas su identificación (y agrupación de los contenidos según éstas) es un factor determinante para el éxito en el filtrado de contenidos. De hecho, la selección de características debería estar guiada por la detección de temáticas y complementada con información de correlación con la variable objetivo. En esta línea surge

[1] https://securelist.lat/spam-and-phishing-in-2017/85992/

la necesidad de un método de aprendizaje no supervisado [13] para agrupar las palabras de una comunicación con el objetivo de buscar un significado global o contextual. Más concretamente parece adecuado aplicar un método de agrupamiento aglomerado jerárquico [14], [15] denominado clustering conceptual [16] para realizar la generalización de las palabras buscando un sentido semántico de más alto nivel. Existen evidencias de que algunos tipos de información semántica complementaria (como la polaridad o el análisis de sentimientos) puede resultar significativamente útil en los procesos de clasificación [17], por lo que se debe seguir profundizando en esta línea.

III. Plan de trabajo

El trabajo de investigación explorará las posibilidades de aprovechar la información semántica de los contenidos con el objetivo de detectar la intencionalidad de los mensajes, y así diseñar, desarrollar y evaluar un sistema de detección de spam que obtenga mejores resultados que los sistemas actuales, y que sea configurable para el usuario final.

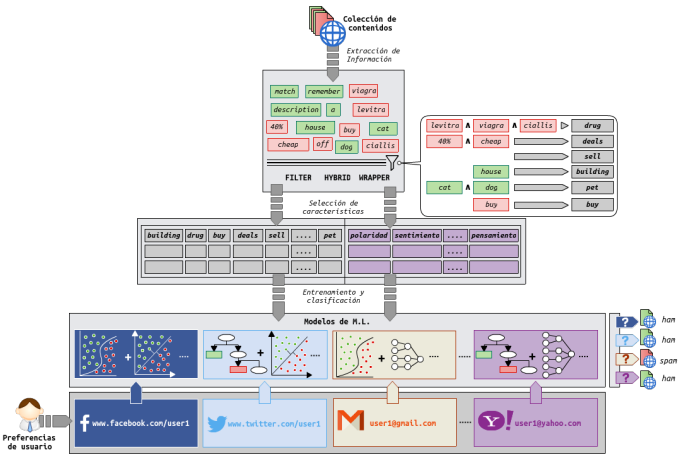


Figura 1. Arquitectura del sistema propuesto

En la Figura 1 se describe el proceso de clasificación que se pretende implementar en este proyecto. Se parte de una colección de documentos clasificados para extraer sus palabras, las cuales son analizadas semánticamente para extraer la intencionalidad del contenido. Parece adecuado aplicar el algoritmo llamado Attribute Oriented Induction (AOI) como un método de clustering conceptual con un lenguaje de descripción de conceptos especializados, para realizar la generalización de las palabras de un mensaje buscando un sentido semántico de más alto nivel. Gracias a esta generalización, los datos previamente distintos se vuelven idénticos y pueden fusionarse. Así por ejemplo, tomando como referencia la base de datos léxica Wordnet, se puede observar que palabras como ”viagra”, ”cialis” o ”levitra” encontrados en documentos spam se pueden generalizar a través de sus hiperónimos, en la característica ”drug” siendo necesario decidir el nivel de generalización (”sexual_drug”, ”drug”, ”chemical_compound”, etc). De esta forma el proceso de extracción de información de un contenido tiene que ver con la identificación de temáticas y no con el cálculo de correlaciones entre las propiedades y la variable objetivo. De cara a la mejora de la precisión de los clasificadores, a este grupo de características se le añadirá la polaridad y el análisis de sentimientos. Como se muestra en la Figura 1, además de la información semántica, los modelos de ML recibirán las preferencias del usuario para realizar el filtrado de los contenidos con mayor precisión.

Resumiendo, esta propuesta contempla el uso de técnicas avanzadas de información semántica, y pretende la creación de un sistema capaz de reconocer la/las temáticas principales de los contenidos para ser capaz de realizar su clasificación en distintos perfiles de comunicación configurables por el usuario. Así, de acuerdo con la Fig. 1, un usuario podría decidir que no está dispuesto a recibir promociones a su cuenta de correo personal, pero sí a través de otros medios.

Agradecimientos

Este trabajo ha sido financiado parcialmente por MINECO/AEI/FEDER, a traves de los proyectos TIN2017-84658-C2-1-R y TIN2017-84658-C2-2-R.

Referencias

[1] Chandra, A.; Suaib, M. ”A survey on web spam and spam 2.0”. *Int. Journal of Advanced Computer Research*, 4(2):634-644. 2014.

[2] Fdez-Glez, J.; Ruano-Ordás, D.; Laza, R.; Méndez, J. R.; Pavón, R.; Fdez-Riverola, F. ”WSF2: A Novel Framework for Filtering Web Spam”. *Scientific Programming*. Available at https://www.hindawi.com/journals/sp/2016/6091385/. DOI: 10.1155/2016/6091385. 2016.

[3] Pérez-Díaz, N; Ruano-Ordás, D.; Fdez-Riverola, F.; and Méndez, J. R. ”SDAI: An integral evaluation methodology for content-based spam filtering models”. *Expert Syst. Appl.*, 39(16):12487-12500. DOI: 10.1016/j.eswa.2012.04.064. 2012

[4] Gómez-Hidalgo, J. M.; Bringas, G. C.; Sánz, E. P.; García, F. C. ”Content based SMS spam filtering”. *In Proceedings of the 2006 ACM symposium on Document engineering*, (pp. 107-114). ACM. 2006.

[5] Gudkova, D.; Vergelis, M, Demidova, N; Shcherbakova, T. ”Spam and phishing in Q2 2016”. Available at https://securelist.com/analysis/quarterly-spam-reports/75764/spam-and-phishing-in-q2-2016/. 2016.

[6] Conrad, E. ”Detecting Spam with Genetic Regular Expressions”. (Accessed 05.04.17) https://www.sans.org/reading-room/whitepapers/email/detecting-spam-genetic-regular-expressions-2006. 2007.

[7] Giyanani, R.; Desai, M. ”Spam detection using natural language processing”. *Int. J. Comput. Sci. Res. Technol*, 1: 55-58. 2013.

[8] Perveen, N.; Missen, M. M. S.; Rasool, Q.; Akhtar, N. ”Sentiment Based Twitter Spam Detection”. *International Journal of Advanced Computer Science and Applications*, 7(7): 568-573. 2016.

[9] Blanzieri, E.; Bryl, A. ”A survey of learning-based techniques of email spam filtering”. *Artificial Intelligence Review*, 29(1):63-92. 2008.

[10] Geng, G. G.; Jin, X.-B.; Zhang, X.-C.; Zhang, D. X. ”Evaluating web content quality via multi-scale features”. *International Journal of Computing Research Repository (CoRR)*. arXiv:1304.6181v1. 2013.

[11] Liu H.; Yu, L. ”Toward integrating feature selection algorithms for classification and clustering”. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):491-502. 2005.

[12] Forman, G. ”An extensive empirical study of feature selection metrics for text classification”. *Journal of Machine Learning Research, Special Issue on Variable and Feature Selection*, pp. 1289-1305. 2003.

[13] Berkhin, P. ”Survey of clustering data mining techniques”. *Technical report, Accrue Software*, San Jose, CA. URL citeseer.ist.psu.edu/berkhin02survey.html. 2002.

[14] Jain, A.K.; Dubes, R.C. ”Algorithms for Clustering Data”. *Prentice-Hall, Inc*. 1988.

[15] Kaufman, L.; Rousseeuw, P.J. ”Finding Groups in Data: An Introduction to Cluster Analysis”. *Series in Applied Probability and Statistics. Wiley-Interscience, John Wiley & Sons, Inc*, New York, USA. 1990.

[16] Pitt, P.; Reinke, R.E. ”Criteria for polynomial-time (conceptual) clustering”. *Machine Learning*, 2(4):371-396. ISSN 0885-6125. 1988.

[17] Ezpeleta, E.; Zurutuza, U.; Gómez Hidalgo, J.M. ”Does sentiment analysis help in bayesian spam filtering?”. *Hybrid Artificial Intelligent Systems Volume 9648 of the series Lecture Notes in Computer Science*, pp 79-90, Springer International Publishing. 2016.

cAPTor - a multi-language Tor analysis tool for APT groups activities

Viatcheslav Zhilin, Jose M. de Fuentes, Lorena Gonzalez-Manzano
Computer Security Lab (COSEC). Universidad Carlos III de Madrid.
Avenida de la Universidad 30, 28911 Leganés, Madrid
100356363@alumnos.uc3m.es, {jfuentes,lgmanzan}@inf.uc3m.es

***Abstract*—Advanced Persistent Threats (APTs) are attracting attention due to their persistence and eventual sophistication. Identifying the groups behind and discovering how they organise seems a promising alternative to achieve an early reaction. However, this task is challenging due to the numerous tools that hackers can use to stay anonymous, being Tor one of them. Previous works have attempted to analyze Tor's network distribution and expose its contents to the clear web, but without any focus on intelligence gathering. In this paper, we propose cAPTor, a tool to collect and analyse information from the Tor network related to threat actors such as APT groups. cAPTor features multi-language support, a keyword search that can be updated on-the-fly, and basic deanonymization techniques. To illustrate its application, cAPTor is applied over APT 29 group. While no relevant conclusions have been found related to the group, cAPTor is released to foster further research in this direction.**

***Index Terms*—Crawling, Tor, OSINT, APT.**

Tipo de contribución: *Investigación original*

I. Introduction

With the advent of networks and the explosion of connected devices, Internet has attracted a vast amount of users and resources. Along with the huge set of services that are offered based on this network, a significant amount of risks have been developed. Indeed, many forms of online threats have appeared, such as ransomware [1], phishing [2] or data theft [3], to name a few.

This novel cybercrime economy has motivated the appearance of organised groups of cybercriminals. In particular, since the appearance of Stuxnet a novel type of online threat has been identified – Advanced Persistent Threats (APTs) [4]. According to the US Democratic Policy and Communications Center, APTs have been defined as "A group, such as a foreign government, with both the capability and intent to continually and effectively target a specific entity, often to conduct espionage or attack operations"[1]. Thus, one of the main goals of APT groups is to gather intelligence [5], which can be considered as Internet-enabled espionage.

In order to counter APTs, as it happens with traditional malware or IT threats, there are two main approaches, namely *a posteriori* and *a priori*. In the former, mitigation techniques come into play to minimize the impact and identify the root cause and (potentially) its author. However, in this case the damage is already caused and the effect can be severe if suitable countermeasures were not properly in place. Moreover, attribution is usually hard [6]. On the contrary, a priori approaches take place before the attack has even occurred, thus enabling predictive responses. Thus, in this paper we focus on this particular approach.

One of the main goals of a priori approaches is to identify who is the attacker behind. This would lead to understand its motivation and its internal organisation. For instance, it is common for APT groups to recruit their members from the cyberspace, governmental[2] or not, by simply posting job offers [7]. This can lead to uncovering their attack vectors or the technologies they use. Here is where operations security (OPSEC)[3] is primordial.

Despite their promised benefits, a priori approaches have to deal with existing anonymisation techniques [8]. Indeed, anonymity is a double-sword edge – it may allow regular users to avoid censorship for honest purposes, or allow cybercriminals to carry out their actions while remaining unliable. One of the main technologies in this regard is the Tor network [9].

There have been numerous efforts regarding the analysis of Tor anonymizing network. Some of them include a deep analysis of the network [10], analysis of BitTorrent traffic through Tor [10], the network distribution [11], what search engines are available and what are they capable of [11], the exposition of content to the clear web [12] [13] [14] and topic exploration [15]. Apart from these research works, there are multiple tools that have been developed for specific purposes to help analyze the Tor network. Despite all these efforts, to the best of authors' knowledge no previous work has addressed how to perform a targeted search about APT groups. This can be considered a particular case of Open Source Intelligence (OSINT).

To overcome this limitation, in this paper we propose cAPTor, a Tor crawling tool that helps gathering intelligence from specific targets (such as APT groups). cAPTor features multi-language support and uses a keyword list that can be updated in real time. Moreover, the tool offers basic deanonymization strategies. To illustrate its application, we apply cAPTor on many Tor sites looking for traces regarding APT 29 group. This group has attracted attention due to its alleged participation in the 2016 US election[4].

The structure of the paper is the following. Section II presents the background. Section III introduces the the pro-

[1] https://www.newamerica.org/cyber-global/compilation-of-existing-cybersecurity-and-information-security-related-definitions/, last access March 2018

[2] https://www.theregister.co.uk/2017/08/24/accused_yahoo_hacker_pleads_not_guilty/?mt=1503538172146

[3] http://conference.hitb.org/hitbsecconf2012kul/materials/D1T3%20-%20The%20Grugq%20-%20OPSEC%20-%20Because%20jail%20is%20for%20wuftpd.pdf

[4] https://www.washingtonpost.com/news/politics/wp/2017/07/06/heres-the-public-evidence-that-supports-the-idea-that-russia-interfered-in-the-2016-election/, last access March 2018

posal of the paper. The design of our proposed tool is explained in Section IV. The evaluation is presented in Section V. In Section VI we describe the related work. Finally Section VII outlines conclusions and future work.

II. Background

In this section we will give a brief introduction to Tor, discussing how the onion routing and hidden service protocol work and how an onion name is obtained. Afterwards, a brief description of APT 29 is provided.

A. The Onion Router

The Onion Router (Tor)[9] is a free and open source project which provides a circuit-based anonymous communication service. This is achieved by directing the traffic through the Tor network, which is an overlay network consisting of volunteer relays which conceal the users location from anyone trying to perform basic network surveillance or traffic analysis. However, this doesn't prevent determining if the access if being performed though Tor by the online service being accessed[16].

Remarkably, Tor counts on an open source implementation supported by a public specification. It is self authenticated, provides with end-to-end encryption and has over two million daily users[17].

In a nutshell, when a client desires to communicate through Tor he establishes a circuit which allows a bidirectional communication. This circuit consists of three repeaters or hops. Repeaters are Tor instances which accept and replicate traffic from another one. Thus, they can be seen as transparent proxies which route traffic to the next hop of the circuit. Once the client has chosen the circuit, it requests the public key of each repeater to encrypt the packets. This encryption is carried out in a layered manner – first the public key of the exit node is applied and afterwards that of the remaining nodes in reverse order. Thus, repeaters only need to apply their own private keys to discover which is the next step in the circuit. Indeed, this is all information they get from each packet.

Leveraging the Tor network, many services (e.g., HTTP, FTP, SMB, etc.) can be offered. They are usually referred to as *hidden services*. To be available for clients, the hidden service will have to register its information through a Hidden Service Descriptor – a file containing the onion address, the public key and the list of its entry points.

B. Onion name generation

There are 1.2 septillion possible names for Tor hidden services, which were meant to be self-authenticating[18]. The procedure to generate all of them is described in the following.

First of all, a 1024 bit RSA key pair is generated. Then the public key of the key pair is converted using ed25519[19], which is an elliptic curve cryptography (ECC)[5]. The first half of the generated result is then encoded with Base32. The result is a character string, formed by characters (from *a* to *z*) and digits (from *2* to *7*). This will be the name of the hidden service followed by ".*onion*".

In order to explore all possible names, several frameworks have been developed. The most notorious ones are *Scallion*[6] and *Shallot*[7]. As an interesting fact about *Shallot*, on a 1.5GHz processor, with fourteen desired starting characters of the onion name (from the total length of sixteen characters), it would take about 2.6 million years to find the necessary RSA key pair.

C. Advanced Persistent Threat 29 (APT29)

According to the cybersecurity firm *Crowdstrike*, APT29 group is closely tied to Russian intelligence entity Federal Security Service (FSB) and Foreign Intelligence Service (SVR)[20].

Among this group objectives we can identify some commercial entities, but mostly government organizations from countries like USA, Germany, South Korea, Turkey, Poland and Norway.

Their attack vector usually includes three stages (see Figure 1): intrusion, propagation and data leakage.

The intrusion stage begins with an initial compromise in the form of spear-fishing, with which the attackers establish an initial foothold in their target using their malware toolsets.

This is followed by the propagation stage which consists of an internal reconnaissance to obtain the high priority objectives in which an escalation of privileges is carried out using tools like Mimikatz[8] or procdump[9]. Then they perform lateral movements to keep growing inside the target and try to achieve persistence within the environment.

Finally, in the data leakage stage, the sensitive information is exfiltrated using innovative and unorthodox methods like applying steganography techniques[21] or using social networks as tunnels[22].

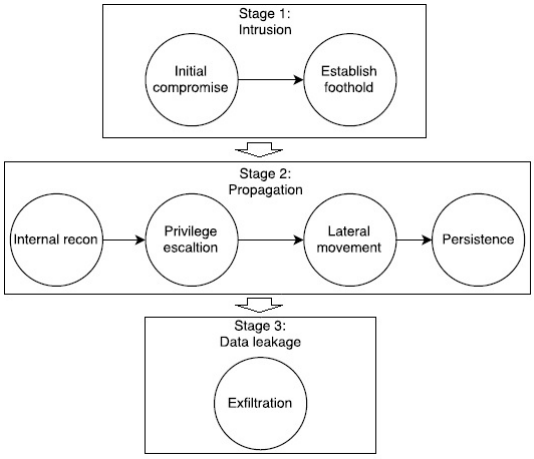


Figure 1: APT29 attack vector

The known toolsets attributed to this group are: PinchDuke, GeminiDuke, CosmicDuke, MiniDuke, CozyDuke, OnionDuke, SeaDuke, HammerDuke and CloudDuke. We point the reader to [5] for a detailed explanation on these artifacts.

There have been several notorious campaign launched by this group. In July of 2014 the *Office Monkeys* spear-phishing campaign took place. These emails contained an e-fax arrival

[5]https://www.johannes-bauer.com/compsci/ecc/
[6]https://github.com/lachesis/scallion
[7]https://github.com/katmagic/Shallot
[8]https://github.com/gentilkiwi/mimikatz
[9]https://docs.microsoft.com/en-us/sysinternals/downloads/procdump

notification which infected the victims with CozyDuke malware. In one instance the email had a zip file attached which contained a Flash video file.

During the CloudDuke campaign in 2015, this toolset was analyzed and received a lot of attention but APT29 continued with their operation. This proved that they prioritized the continuity of the operation over stealthiness.

In June 2016 in the Democratic National Committee's servers appeared to have several presences, which were unaware of each other. According to existing reports, this seems a coordinated action between APT29 and APT28 (also known as *Fancy Bear*)[23].

III. Proposal overview

In this section, the outline of the proposal is presented (Section III-A). Afterwards, the goal to achieve and the working assumptions are presented in Sections III-B and III-C, respectively.

A. Outline

This work aims to help on spotting APT groups which are using Tor sites to organise themselves. For this purpose, there are two main actions to be carried out. On the one hand, traces about the group activity have to be discovered. Once they are found, the next step is to deanonymise the actor at stake.

In this work, we propose a tool, called cAPTor, to address both steps. With respect to finding traces, cAPTor receives a series of keywords and Tor sites that have to be crawled. Once these traces are found, cAPTor can look for additional evidences (e.g., Shodan results) that may be helpful to deanonymise the author behind.

Using the method above, cAPTor aims to collect information based on the user's OPSEC mistakes, which has been identified as a means to break anonymity [17]. In particular, cAPTor keywords will be related to posts about specific subjects, email accounts, snippets of code or anything that could be related to the APT group at stake.

B. Goals

The objective of this project is to develop a tool that is able to obtain information from hidden services which could be analyzed and offer relevant information regarding it. The intended use case is to locate any asset related to the internal activity of APT groups. It should be able to be configured with a starting seed if needed and store a big amount of information. This general goal can be translated into specific features as follows:

- **Tor crawling.** The proposed tool must be able to crawl the Tor network based on a starting seed.
- **Multi-language support.** The proposed tool must be able to analyze websites written in different languages.
- **Keyword search.** The tool must be able to tell pages apart depending on the existence of a pre-defined set of keywords.
- **Deanonymization.** If the information is available, the tool should perform a basic deanonymization on the specific target.

C. Working assumptions

The tool is intended to work under the following assumptions. First, the analyst should have clearly defined the objective and collected some potential hints (keywords) beforehand. Second, we aim to look for information in accesible sites, thus leaving restricted areas out of the scope. Third, the keywords should be comprehensive enough to cope with evolving types of communication, such as *leetspeak*[24] or a specific slang[25].

IV. cAPTor design

In a nutshell, cAPTor will crawl some specific websites in order to collect keyword-based pieces of information. Once this data is retrieved, an in-depth analysis will be carried out to extract additional elements that may be of the interest for the analyst. Therefore, in the following we introduce the two main steps carried out by cAPTor – crawling (Section IV-A) and post-analysis (Section IV-B).

A. Crawling module

On input of a set of keywords and a set of initial domains to analyse, the *crawler module* (Figure 2) performs two main tasks: requests web pages from the hidden services and connects through SSH to these services. All this is done through the Tor network which allows crawling both the hidden services and the clear web.

The inner workings are of simple web crawler/spider. It visits a page, extracts links and visits these links (prioritizing onion links over clear web links), skipping the already visited pages. While visiting a new domain it also tries to connect through SSH to the hidden service to obtain the SSH client fingerprint. The web responses are stored for the post-processing module.

B. Post-processing module

The *post processing module* (Figure 3) performs various tasks. The previously stored SSH fingerprint is queried on Shodan[10] to see if this service is aware of an IP address with the same fingerprint. In that case, this result leads us to conclude that (with some probability), that address is the one of the hidden service machine. Moreover, several additional evidences are extracted from the studied resource. On the one hand, this module tries to detect the language in which the resource is written. This helps to strength the link of a resource to a given speaking region. Moreover, indirect identifiers such as e-mail or Bitcoin addresses are gathered as well. This can be helpful to enable future attribution of APT-related incidents. Last but not least, links to other protocols (e.g., FTP, IRC or XMPP, etc.) are extracted. These elements could prove to be useful for a posterior in-depth inspection.

The vast majority of the information flows from the crawler module to the processing module, which is then filtered and offered to the analyst. In case it were necessary, since all the information from both modules is stored in a database, it could be queried with other parameters to perform other analysis.

[10]https://www.shodan.io/,lastaccessMarch2018

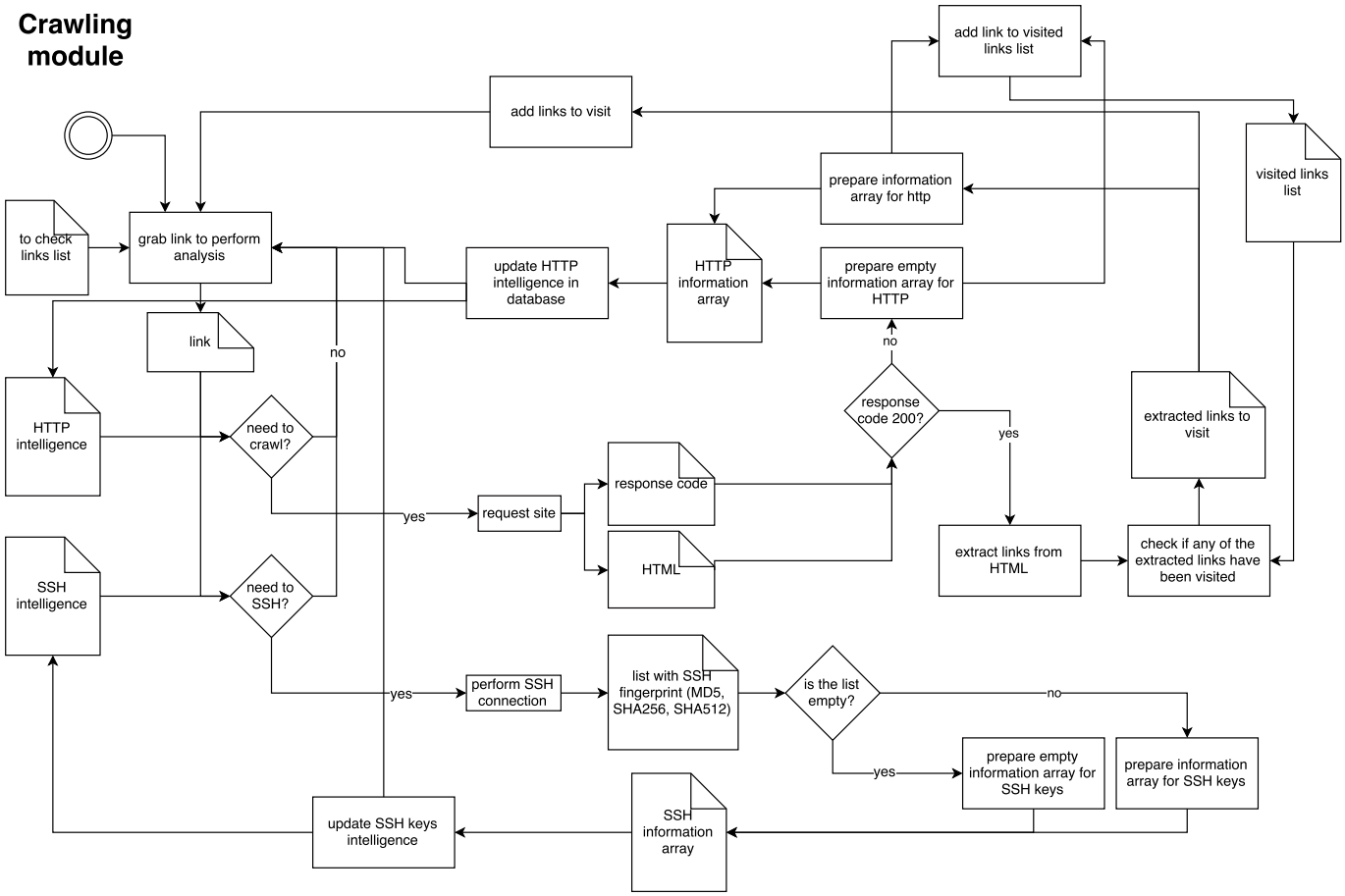


Figure 2: Crawling module diagram

V. Evaluation

In this section, the proposed approach is assessed. For this purpose, we first analyse whether the proposed goals are met (Section V-A). Moreover, we experimentally show the application of cAPTor in a concrete use case. In particular, we attempt to find evidences about the internal organisation of APT29. We describe the experimental setup and the obtained results in Sections V-B and V-C, respectively. To foster further research in this area, cAPTor has been released in GitHub[11].

A. Goal assessment

cAPTor meets all intended goals. In particular, several design decisions ensure that the system meets these requirements. Thus, cAPTor is able to crawl both clear web and Tor-based sites. Moreover, it has been coded in such a way that multiple character sets are allowed, thus being able to process latin and non-latin based (e.g. cyrillic, greek, etc.) resources.

Concerning keywords, cAPTor uses them as one of the input parameters. Remarkably, this keyword list can be updated in real-time, thus enabling the refinement of results during a crawling operation.

Last but not least, cAPTor is able to reveal IP addresses that are associated to a given hidden service. This feature, along with other indirect identifiers such as e-mail addresses, can serve as a basic means for deanonymization.

[11]https://github.com/viatchezhilin/cAPTor

B. Experimental setup

In order to apply cAPTor, a Virtual Private Server (VPS) with *Ubuntu 16.04.3 LTS*, 8 GB of RAM and 40 GB of hard disk space was used. By using a VPS the possibility of a power surge or network instability is minimized. This is also used as a security layer in case the crawler visits a malicious website (e.g. drive-by).

As a seed for the crawler a list with 9.102 unidentified unique onion links was used. This list comes from a GitHub public project[12].

As for the terms that the crawler is searching for a dictionary with several hacker terms, in English and Russian, known malware toolsets, malware campaigns and known hacker group names and denominations. Thus, the list contains terms such as uroboros, уязвимости, MiniDuke or эксплойт, to name a few.

C. Results

First we analyzed the nine thousand links from the seed with Onionscan (which will be discussed in Section VI) to perform a preliminary analysis.

Figure 4 represents the connections between domains of our seed that Onionscan was able to obtain. Purple nodes are onion domains and orange nodes are clear web domains. It is clear that the majority of the nodes are from onion addresses

[12]https://github.com/automatingosint/osint_public/blob/master/onionrunner/onion_master_list.txt

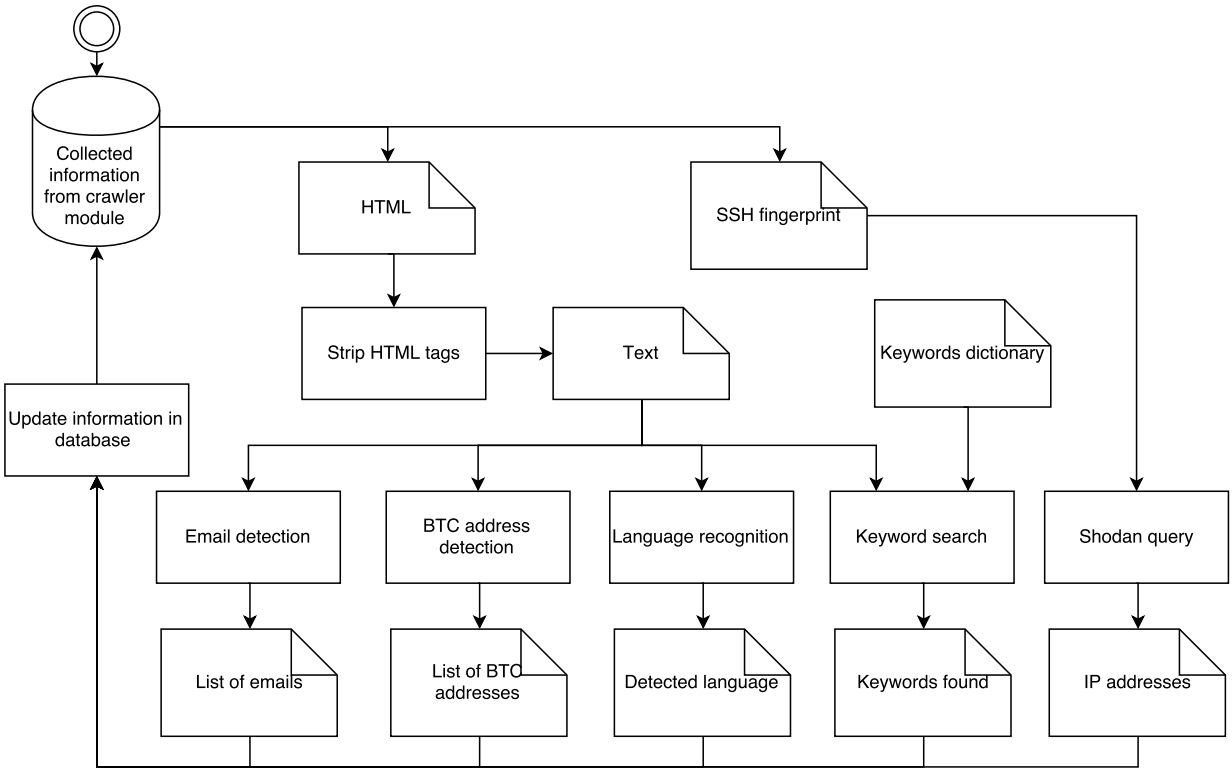


Figure 3: Post processing module diagram

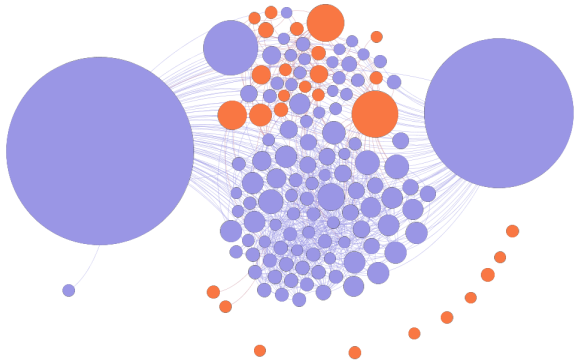


Figure 4: Filtered graph with nodes with degree 50 from Onionscan analysis of 9.000 onion links

that lead to other onion addresses, but there are several nodes that are connected to the clear web.

The two biggest nodes correspond to *"7cbqhjnlkivmigxf.onion"* and *"gotchafjkmcqdz2x.onion"* (first one being the biggest node on the left and the other one being the biggest node on the right, respectively).

"7cbqhjnlkivmigxf.onion" is *"The onion crate - Tor hidden service index"*, which is just compilation of several hidden site links. And *"gotchafjkmcqdz2x.onion"* is *"GOTCHA! A list of LEAKED ONION Websites"*, which is currently offline, but from the title of the page we can deduct that it is another page with a list of hidden services sites.

Then we used cAPTor with the same seed and obtained the results shown in Table I. The total amount of links that our tool visited is 96.258.

From the total amount of links visited, about 23.27% were offline. This is a typical behavior with hidden services since the administrators tend to migrate their pages from place to place. This sometimes has another reason, when a site goes down permanently, which can occur because of *exit scams*. When a marketplace has an ESCROW service[26][27] (usually run by the owners of the marketplace) and there is a lot of currency in transit, the administrators which act as a middleman between the vendors and buyers may decide to run with the money.

Apart from the links visited, there have been a total of 10.363 attempts to establish an SSH connection to a unique domain. We specified in the configuration file of our tool to try to establish the connection to port 22, which is the default port for SSH.

From all these tries, only 24 SSH fingerprints could be obtained. This indicates several possibilities. It can be that the SSH client of the server is not running on port 22, which is a good security habit. The other possibility is that SSH is not used during the configuration of the server and other tools are used for managing the site, such as control panels offered by hosting providers (e.g. OneHostCloud[13]).

From the twenty four SSH fingerprints we could deanonymize four hosts with the technique mentioned earlier in Section IV. By searching the fingerprint in Shodan there were a few hosts which returned several IPs for a single fingerprint.

Analyzing the links from the available web pages, the average number of links per page is 129.15. Moreover, 95% of the collected links from the visited sites were onion links. This result corresponds to the results from the first analysis

[13]https://onehostcloud.hosting/tor-hidden-service/

Links visited	Response code 200	Response code not 200	Total SSH requests	SSH with fingerprint	Emails
96.258	73.852	22.406	10.363	24	453.443
Deanonymized SSH	**Average number of links per page**	**% of onion links**	**% of clear web links**	**Links with keywords**	**BTC addresses**
4	129,15	95,21%	4,79%	7480	61

Table I: Results from cAPTor with a given seed

from Onionscan, where the majority of domains were hidden services.

Inspecting the contents of the visited pages, about 7,500 pages contained a keyword from our dictionary. The distribution of the found keywords can be seen in Figure 5, where the keyword with the most coincidences was *forum*, followed by the keywords *exploit* and *hacker*.

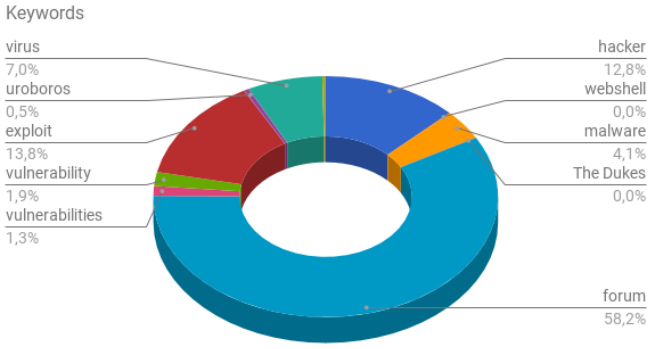


Figure 5: Keywords distribution

By using Natural Language Toolkit[14] (NLTK) we tried to detect the language in which the collected pages were written. The results are shown in Figure 6, where the most used language is English. This is an expected result since English is the most used language in the Internet. It was unexpected not to find any Russian written sites. This can be due to the fact that some individuals communicate in Russian but write with the Latin alphabet and the NLTK library does not consider such scenarios.

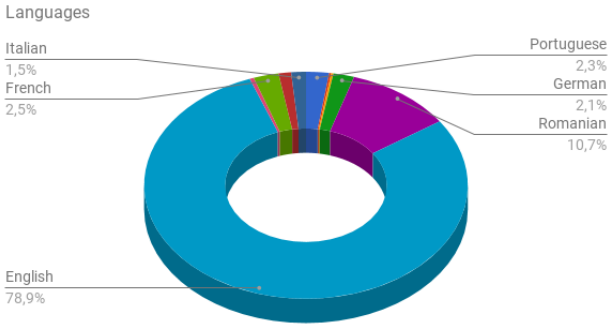


Figure 6: Languages distribution

Last but not least, cAPTor extracted 453,443 e-mail addresses and 61 Bitcoin addresses. The distribution of the email domains can be seen in Figure 7. Interestingly, the domains that stand out are *mit.edu* (1,727 matches) and *ibm.net* (2,231 matches).

VI. Related work

There have been many previous attempts related to the analysis of contents in the Tor network or the deep web in

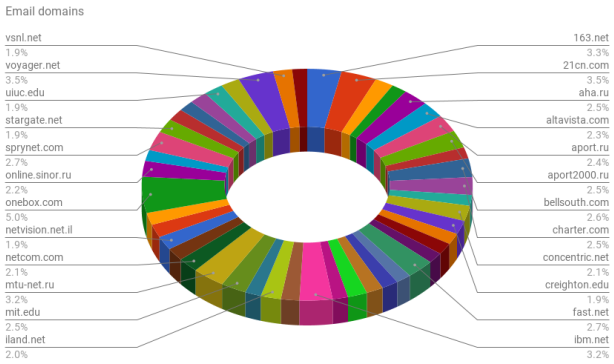


Figure 7: Filtered email domains distribution (domains with more than 1.000 occurrences)

general. In the following, we describe them and compare them against our approach. Table II summarizes the results of the comparison.

In [10] an analysis of the Tor anonymizing network was performed, with emphasis on Bittorent traffic flowing through Tor and how it could affect it.

To explore the entrances, scale, structure, distribution and search engines of the Deep Web a research[11] took place with interesting results. For example, the rapid expansion of the Deep Web between the year 2000 and 2004 or the dominant area which is ”e-commerce”.

Several research works [12][13][14][28] have as their main goal the exposition of content from the Deep Web which is hidden behind HTTP forms. Thanks to this, this information could be accessed from the clear web and could prove to be a vast amount of data that could be accessed, for example, from a search engine.

Estimation about the size of the Deep Web, as well as characterization of the content was performed in [28], apart from automating the process of exposure for search engines.

As for topic exploration, in [15] a tool was developed to be able to harness the Deep Web. Some notable features of their tool, *Kosmix*, are: federated search approach, caching and query transformation. Some of the features were important during the design and development to ensure a reasonable performance of the tool.

There are several developed frameworks that offer a solution to crawling the dark web, and in particular Tor. Some of them are:

- *Onionscan*[15]. This is a framework with two main goals: help administrators of hidden services with operational security of their services and help researchers to monitor and track hidden services. They publish a monthly report with snapshots of the dark web focusing on the correlation between services. It’s written in Go.

[14]http://www.nltk.org/

[15]https://onionscan.org/

	Tor crawling	Multi-language support	Keyword search	Security assessment	Deanonymization
Onionscan	Yes	No	No	Yes	Yes
DreamStorm	Yes	No	No	No	No
FreshOnions	Yes	No	Yes	No	No
Hunchly	Yes	No	No	No	No
Kosmix[15]	Yes	No	Yes	No	No
Accessing the Deep Web: A Survey[11]	Yes	No	No	No	No
Google's Deep-Web Crawl[12]	No	Yes	Yes	No	No
Harnessing the Deep Web: Present and Future[13]	No	Yes	Yes	No	No
ViDE[14]	No	Yes	Yes	No	No
The Deep Web: Surfacing Hidden Value[28]	No	Yes	Yes	No	No
cAPTor	Yes	Yes	Yes	No	Yes

Table II: Related work analysis

- *DreamStorm*[16]. A simple Python crawler library.
- *FreshOnions*[17]. A compilation of hidden services which crawls twice a day and returns a list of validated links.
- *Harry 71's Onion Spider Robot*[18]. A spider which compiled a list of available hidden services. Currently it is offline.
- *Hunchly Daily Hidden Services Report*[19]. A daily mailing list which notifies of new discovered hidden services. A distributed crawler is used to offer this kind of information.

VII. Conclusion

Cybercrime has become a major worldwide problem with the increase of hacking and malware campaigns. In order to counter these threats, a priori techniques can enable an early reaction. To contribute in this direction, in this work cAPTor has been proposed. This tool carries out multi-language crawling for Tor sites. cAPTor outperforms existing tools in that it collects specific information that can be useful in the process of identification and attribution. To motivate further works, cAPTor has been publicly released as open source software. Although at the moment cAPTor has not found evidences of APT groups operating in Tor, it opens the door for further in-depth analysis in this direction.

As a future work, we plan to perform URL pattern recognition which then will be used to perform URL fuzzing. On the other hand, we aim to improve the detection of HTML mirrors (i.e., if two domains are replicated at different addresses).

VIII. Acknowledgement

This work has been partially supported by MINECO grant TIN2016-79095-C2-2-R (SMOG-DEV) and CAM grant S2013/ICE-3095 (CIBERDINE), co-funded with European FEDER funds.

References

[1] S. Mohurle and M. Patil, "A brief study of wannacry threat: Ransomware attack 2017," *International Journal*, vol. 8, no. 5, 2017.
[2] T. Dakpa and P. Augustine, "Study of phishing attacks and preventions," *International Journal of Computer Applications*, vol. 163, no. 2, 2017.
[3] C. van Der Walt, "The impact of nation-state hacking on commercial cyber-security," *Computer Fraud & Security*, vol. 2017, no. 4, pp. 5–10, 2017.
[4] M. K. Daly, "Advanced persistent threat," *Usenix, Nov*, vol. 4, no. 4, pp. 2013–2016, 2009.
[5] A. Lehtiö, "The dukes: 7 years of russian cyber-espionage," 2015.
[6] D. P. Fidler, "The us election hacks, cybersecurity, and international law," *American Journal of International Law*, vol. 110, pp. 337–342, 2016.
[7] R. Stoyanov, "Russian financial cybercrime: How it works," *Kaspersky Lab Cybercrime Underground Report*, 2015.
[8] A. Pfitzmann and M. Köhntopp, "Anonymity, unobservability, and pseudonymity—a proposal for terminology," in *Designing privacy enhancing technologies*. Springer, 2001, pp. 1–9.
[9] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC, Tech. Rep., 2004.
[10] A. Chaabane, P. Manils, and M. A. Kaafar, "Digging into anonymous traffic: A deep analysis of the tor anonymizing network," in *Network and System Security (NSS), 2010 4th International Conference on*. IEEE, 2010, pp. 167–174.
[11] B. He, M. Patel, Z. Zhang, and K. C.-C. Chang, "Accessing the deep web," *Communications of the ACM*, vol. 50, no. 5, pp. 94–101, 2007.
[12] J. Madhavan, D. Ko, Ł. Kot, V. Ganapathy, A. Rasmussen, and A. Halevy, "Google's deep web crawl," *Proceedings of the VLDB Endowment*, vol. 1, no. 2, pp. 1241–1252, 2008.
[13] J. Madhavan, L. Afanasiev, L. Antova, and A. Halevy, "Harnessing the deep web: Present and future," *arXiv preprint arXiv:0909.1785*, 2009.
[14] W. Liu, X. Meng, and W. Meng, "Vide: A vision-based approach for deep web data extraction," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 3, pp. 447–460, 2010.
[15] A. Rajaraman, "Kosmix: high-performance topic exploration using the deep web," *Proceedings of the VLDB Endowment*, vol. 2, no. 2, pp. 1524–1529, 2009.
[16] R. Singh, R. Nithyanand, S. Afroz, P. Pearce, M. C. Tschantz, P. Gill, and V. Paxson, "Characterizing the nature and dynamics of tor exit blocking," 2017.
[17] R. Dingledine, "Next generation tor onion services," 2017.
[18] P. Syverson and G. Boyce, "Genuine onion: Simple, fast, flexible, and cheap website authentication," *arXiv preprint arXiv:1506.04115*, 2015.
[19] D. J. Bernstein, S. Josefsson, T. Lange, P. Schwabe, and B.-Y. Yang, "Eddsa for more curves." *IACR Cryptology ePrint Archive*, vol. 2015, p. 677, 2015.
[20] D. Alperovitch, "Bears in the midst: Intrusion into the democratic national committee," *From The Front Line, update*, 2016.
[21] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal processing*, vol. 90, no. 3, pp. 727–752, 2010.
[22] F. E. T. Intelligence, "Hammertoss: Stealthy tactics define a russian cyber threat group," *Milpitas, CA: FireEye, Inc*, 2015.
[23] P. Paganini, "Apt28: Fireeye uncovered a russian cyber espionage campaign," *Security Affairs*, vol. 29, 2014.
[24] J. M. Tellería Gelabert, "English and leetspeak: A step towards global nerdism?" 2012.
[25] E. S. Raymond *et al.*, *This is Jargon File*. Wiretap, 2000.
[26] X. Hu, Z. Lin, A. B. Whinston, and H. Zhang, "Hope or hype: On the viability of escrow services as trusted third parties in online auction environments," *Information Systems Research*, vol. 15, no. 3, pp. 236–249, 2004.
[27] S. Antony, Z. Lin, and B. Xu, "Determinants of escrow service adoption in consumer-to-consumer online auction market: an experimental study," *Decision Support Systems*, vol. 42, no. 3, pp. 1889–1900, 2006.
[28] L. BrightPlanet, "The deep web: Surfacing hidden value," 2000.

[16] https://github.com/OAlienO/DreamStorm
[17] http://zlal32teyptf4tvi.onion/
[18] http://skunksworkedp2cg.onion/
[19] https://darkweb.hunch.ly/

CyberSecurity Challenge: Detección de talento en ciberseguridad mediante una competición virtual de Capture the Flag

José Carlos Sancho Núñez
Cátedra ViewNext-UEx
Escuela Politécnica
Av. Universidad s/n - Cáceres
jcsanchon@unex.es

[1]Andrés Caro Lindo
[2]Laura Martín Sánchez
Universidad de Extremadura
Escuela Politécnica
Av. Universidad s/n - Cáceres
[1,2]{andresc, laurams}@unex.es

José Andrés Félix de Sande
ViewNext S.A.
CENIT Cáceres (PCTEx)
Av. Universidad s/n - Cáceres
jafelix@viewnext.com

Resumen- **La identificación de talento en materia de ciberseguridad es una tarea compleja, tanto en la educación superior como en los procesos de selección de personal en las empresas del sector tecnológico. Los planes de estudio de las universidades españolas prestan, aún, poca atención a la formación en ciberseguridad, pese a que se ha convertido en una de las disciplinas de la informática mayor demandada en la actualidad. Este trabajo describe una iniciativa conjunta llevada a cabo entre la Universidad de Extremadura (UEx) y la empresa tecnológica Viewnext. La actividad, del tipo Capture the Flag, ha reunido virtualmente a 132 participantes del ámbito nacional e internacional. Se detalla el diseño, los recursos tecnológicos y humanos utilizados para el desarrollo y, finalmente, se plasman una serie de resultados obtenidos tras su celebración.**

***Index Terms*- talento, educación, ciberseguridad, innovación, formación, Capture The Flag (CTF).**

Tipo de contribución: *Formación innovación*

I. Introducción

La detección de talento en una disciplina de la informática tan especializada como es la ciberseguridad es un desafío, debido a que los planes de estudios de las universidades españolas no cubren adecuadamente esta temática. Estas carencias vienen cubriéndose, como mucho, mediante cursos y certificaciones adicionales. Además, hoy en día, poder demostrar habilidades y destrezas reales en el ámbito de la ciberseguridad se augura complicado. Una forma de demostrar estos conocimientos es realizando acciones, en la mayoría de los casos cuasi-ilícitas, siendo poco aconsejables para el alumnado.

Así mismo, la formación a través de competencias en la educación superior es una tarea compleja de llevar a cabo, incluso aplicando técnicas de gamificación [1]. Por este motivo y por el interés generado en temas de ciberseguridad durante los últimos años en nuestra comunidad autónoma [2], la Universidad de Extremadura decidió organizar el *CyberSecurity Challenge Viewnext-UEx*. El objetivo de esta iniciativa es detectar el talento en ciberseguridad en jóvenes promesas del panorama nacional. Está pensada para complementar el evento *ForoCIBER*, único evento divulgativo que trata las temáticas de la ciberseguridad y el derecho tecnológico en la Comunidad Autónoma de Extremadura.

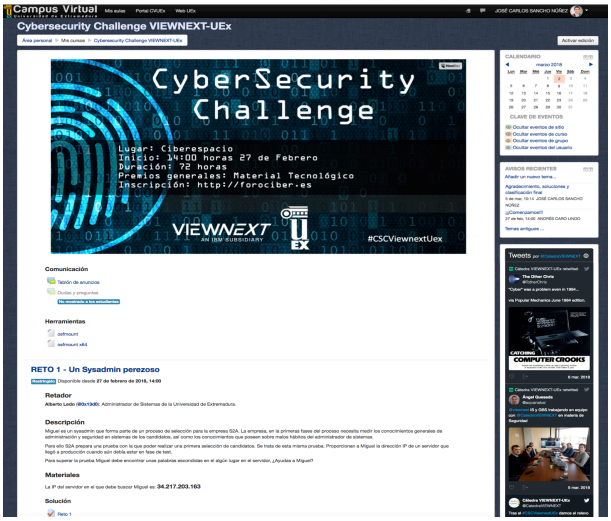


Fig. 1. Curso de Moodle utilizado para la competición.

II. Diseño

El *CyberSecurity Challenge* es una llamativa competición del tipo Capture the Flag (CTF), al igual que otras iniciativas que se realizan a nivel nacional [3] [4]. Celebrada por primera vez en Extremadura y desarrollada en 72 horas de manera completamente virtual, a través de la plataforma Moodle del Campus Virtual de la Universidad de Extremadura, como muestra *Fig.1*. La competición comenzó el día 27 de febrero a las 14:00 horas finalizando el día 2 de marzo a la misma hora, siendo accesible para cualquier persona residente en España y no superase los 35 años. Esta limitación en la edad pretende evitar que profesionales con mucha experiencia puedan desvirtualizar el objetivo principal de esta iniciativa, la detección de talento en jóvenes promesas.

Los retos, concretamente 5, se diseñaron teniendo en cuenta distintos niveles de dificultad y abordando diversas disciplinas de la ciberseguridad como la ingeniería inversa, el exploiting, el hacking web, el análisis forense o la esteganografía. *Fig.2* muestra los retos.

La puntación de cada reto se fijó en función de su dificultad, contemplando los valores 10, 15, 20, 25 y 30 puntos respectivamente, con una suma global de 100 puntos.

Expertos profesionales y colaboradores asiduos en acciones formativas con la Universidad de Extremadura se encargaron de configurar los retos. Todos ellos se mencionan en los agradecimientos de este trabajo.

Organizativamente, a cada persona encargada de proponer un reto se le facilitó una ficha a rellenar, que contemplaba aspectos como: nombre del reto, breve historia descriptiva que permita una contextualización atractiva y motivadora, categoría donde se encuadra el reto, nivel de dificultad, diversas pistas para guiar a los usuarios y soluciones metodológicas de los mismos.

Fig. 2. Tipos de retos y persona encargada de proponerlo.

Seguidamente se detalla, para cada prueba, el enunciado, el material facilitado a los participantes y la solución metodológica a seguir para conseguir la flag escondida.

A. *Reto 1: Un sysadmin perezoso…* – 10 puntos

a) Enunciado: "Miguel es un sysadmin que forma parte de un proceso de selección para la empresa S2A. Para ello S2A prepara una prueba con la que poder realizar una primera selección de candidatos. Se trata de esta misma prueba. Para superar la prueba Miguel debe encontrar unas palabras escondidas en el algún lugar en el servidor, ¿Le ayudas?"

b) Material: Se facilita la IP del servidor.

c) Solución: En primer lugar, se debe realizar un escaneo de los puertos abiertos del servidor: uno relativo a la base de datos es accesible. Seguidamente, hay que averiguar usuario y contraseña y extraer la flag guardada en una tabla.

B. *Reto 2 y 3: Ataca mi web server…* – 15 y 20 puntos

a) Enunciado: "Hola Hacker, tenemos una nueva misión para ti, nos han encargado obtener una información que está contenida en un servidor. Para acceder a la información necesitamos el nombre de usuario y contraseña, dicho terminal se encuentra en: http://www.pentestlab.es/sh.php".

b) Material: Se aporta la dirección web de un servidor.

c) Solución: Inicialmente, hay que obtener el usuario y contraseña del servidor facilitado mediante una inyección SQL a una base de datos. A continuación, se accede a una consola que contiene ficheros con información cifrada en ROT-13, un sencillo cifrado César que descubre las dos siguientes flags.

C. *Reto 4: Un día como analista forense* – 25 puntos

a) Enunciado: Este es un reto real (las evidencias no), está basado completamente en un caso real que he tenido que resolver como analista forense de la Policía Nacional.

b) Material: Se proporciona una imagen de disco fragmentada a los participantes con información básica.

c) Solución: Se debe reconstruir la imagen de disco facilitada preservando la integridad de la misma. Uno de los ficheros indica la utilización de un conocido diccionario para craquear la contraseña de un .zip que permite obtener un archivo .doc. Por último, se comprueba su extensión viendo que no es la adecuada, siendo una imagen que contiene entre sus metadatos la preciada flag.

D. *Reto 5: Lo que ves puede no ser lo que hay...* – 30 puntos

a) Enunciado: "Los límites de la esteganografía están allí donde estén los límites de tu imaginación por lo que deberás encontrar *la palabra mágica*".

c) Material: Se proporciona al usuario una imagen de disco que contiene la información y herramientas necesarias.

c) Solución: Tras recuperar los ficheros borrados de la imagen se encuentran las pistas para descifrar la imagen mediante técnicas esteganográficas y obtener la última flag.

III. Resultados

En la primera edición de esta iniciativa han participado un total de 132 usuarios, de los cuales 11 han finalizado exitosamente la totalidad de las pruebas propuestas y 41 consiguieron superar alguno de ellos.

Sobre el total de participantes indicar que el 57% (75) han sido trabajadores cualificados, el 41% (54) estudiantes y el 2% (3) personas en desempleo. Destacar que entre estos participantes se encuentran varios miembros de Fuerzas y Cuerpos de Seguridad del Estado.

IV. Conclusiones

Se evidencian dos conclusiones fundamentales bien diferenciadas tras la ejecución de esta peculiar actividad. La primera de ellas, es la eficaz trasmisión de conocimiento y, por ende, el aprendizaje adquirido por los usuarios. La segunda, es la obtención de un listado de usuarios que han demostrado altos conocimientos, habilidades y competencias en las diversas disciplinas que conforman la ciberseguridad.

Podemos afirmar que la clave principal de la alta aceptación de esta actividad universitaria ha sido realizar todo el proceso de forma online, desde la inscripción y la realización de las pruebas hasta el seguimiento de la competición.

Tras esta experiencia los autores se proponen llevar estas dinámicas y actividades a los Grados de Ingeniería Informática de la Universidad de Extremadura. Se espera acercar contextos de simulación reales al alumnado, para trasmitir el conocimiento de manera eficaz.

Agradecimientos

Los autores agradecen la implicación en la preparación de los retos a: Alberto Ledo, administrador de sistemas de la UEx; Francisco Moraga, analista de inteligencia; José Aurelio García, perito informático; y Manuel López Guerra, analista forense de la Unidad de Tecnológica de la Policía Nacional.

Referencias

[1] G. J. Conti, T. Babbitt, and J. Nelson, "Hacking competitions and their untapped potential for security education," IEEE Security & Privacy, vol. 9, no. 3, pp. 56–59, 2011.

[2] Andrés Caro: "Una apuesta por la educación en ciberseguridad desde el ámbito universitario", en Jornadas Universitarias de Investigación en Ciberseguridad (JNIC 2015), pp. 198-202, 2015.

[3] "Defcon hacking conference - CTF history," https://www.defcon.org/html/links/dc-ctf.html.

[4] "CTF Cybercamp" https://cybercamp.es/competiciones/CTF_individual.

“Crypto Go”: criptografía simétrica en tapete verde

Ana I. González-Tablas
COSEC Lab – Universidad Carlos III de Madrid
Avda. de la Universidad 30
28911 Leganés, Madrid
aigonzal@inf.uc3m.es

María Isabel González Vasco
MACIMTE – Universidad Rey Juan Carlos
C/ Tulipán, S/N
28933 Móstoles, Madrid
mariaisabel.vasco@urjc.es

Resumen- **En este documento describimos el diseño preliminar de un juego de mesa, “Crypto Go” cuyo planteamiento mimetiza el del conocido juego de cartas “Sushi Go”. El fin de nuestra propuesta es familiarizar al alumno de una manera lúdica con las principales herramientas de clave simétrica. Así, el objetivo de cada partida es llegar a construcciones robustas para conseguir los objetivos de confidencialidad, integridad y autenticación en la transmisión de mensajes. En esta aproximación inicial obviamos numerosos aspectos que pueden incorporarse para completar nuestra propuesta, como la consideración de tamaños de clave o la generación pseudoaleatoria de calidad. Nuestro diseño inicial, sin embargo, es suficiente para conseguir que el alumno afiance los conceptos básicos más relevantes adquiridos en un curso elemental de criptografía simétrica, conozca un gran número de herramientas de amplio uso en la actualidad y sepa identificar errores de planteamiento en construcciones reales.**

Index Terms- gamificación, criptografía simétrica, educación.

Tipo de contribución: *Formación e innovación educativa*

I. Introducción

En la actualidad existe una necesidad urgente de capacitar en ciberseguridad a un gran número de profesionales, así como de concienciar y formar de forma adecuada en esta área a los desarrolladores de sistemas que procesen información. Una de las disciplinas en las que se apoya la ciberseguridad es la criptografía. En este trabajo se propone un juego de mesa con el objetivo de afianzar los conceptos adquiridos en un curso básico de criptografía simétrica. Está, por tanto, enfocado a alumnos de grado.

En los últimos años, se han propuesto un número no desdeñable de juegos educativos que abordan alguna temática en ciberseguridad, arrojando resultados positivos en las evaluaciones a pequeña escala realizadas [1]. La tipología es variada, desde complejos desarrollos software que simulan mundos virtuales (e.g., *CyberCIEGE*) hasta sencillos juegos de cartas (e.g., *Elevation of Privilege*).

En este trabajo se propone un juego de cartas sencillo, de mecánica rápida y sistema de puntuaciones simple. El objetivo didáctico es que los jugadores sean capaces de reconocer cuáles son las primitivas y los esquemas criptográficos adecuados para cubrir ciertos objetivos de seguridad (siguiendo como referencia esencial las recomendaciones publicadas en [2]).

II. Diseño Preliminar de Crypto Go

Crypto Go se plantea como un juego de cartas, sin tablero, en el que los usuarios parten con una mano inicial y pretenden terminar con una selección de cartas que contenga lo que llamamos *Crypto-Kit.* Un *Crypto-Kit* contendrá suficientes herramientas criptográficas para implementar un sistema seguro de criptografía simétrica con garantías de confidencialidad, integridad y autenticación.

Fig. 1. Ejemplo de cartas: 2 anversos (SHA-3 y HMAC) y reverso.

A. *Cartas criptográficas*

Las cartas de la baraja Crypto Go representan herramientas criptográficas susceptibles de ser utilizadas para realizar un esquema de transmisión de información con ciertas garantías (ver sección B). Contemplamos, en esta versión inicial, la inclusión de dos tipos de cartas:

Cartas de primitivas. Cada una representa a una primitiva criptográfica fundamental para cualquier diseño simétrico. En concreto, se incluyen tres modalidades:

1. **BC** – Cifradores de bloque (AES, 3DES, DES,…).
2. **H** – Funciones resumen (MD5, SHA2, SHA3,…).
3. **SC** – Cifradores de flujo (HC-128, SALSA, RC4,…).

Cartas para construcciones combinadas. Representan distintos esquemas que pueden determinar una implementación efectiva de las herramientas representadas por las cartas de primitivas (aisladas o en combinación). Contemplamos, en principio, las siguientes modalidades:

1. **OM –** Modos de operación (EME, FFX, OFB, CTR, CBC, ECB,…), para ser combinados con un cifrador de bloque explicitado por una carta BC.
2. **AE** – Métodos de cifrado autenticado (OCB, EAX, CCM, CWC…), que se construyen a partir de un cifrador de bloque. En su mayoría, estos métodos son implementados mediante la combinación de un modo de operación con un MAC que verifique ciertos requisitos, aunque en el juego simplemente se explicita que dependen de un cifrador en bloque.
3. **MAC** – Códigos de autenticación de mensaje (CMAC, EMAC, AMAC, HMAC,…), construidos con distintas primitivas (cifradores de bloque o funciones hash).

En la *Fig. 1* se ilustra el diseño de las cartas correspondientes a la función resumen SHA-3 y la función de generación de códigos de autenticación de mensajes HMAC. El anverso de cada carta identificará la herramienta criptográfica a la que representa (e.g., SHA-3 y HMAC), identificándose cada tipo de herramienta con un color diferente

(e.g., naranja para las cartas H y azul claro para las MAC). Además, se incluirá una breve reseña de la herramienta representada, sin desvelar su nivel de seguridad. El reverso de todas las cartas ha de ser idéntico para no revelar ni su tipología ni la herramienta concreta que representan.

B. Objetivo del juego

Crypto-Sets y Crypto-Kits. Comenzamos por definir los siguientes conjuntos especiales de cartas o *Crypto-Sets*:

- **Crypto-Set CS1** (Confidencialidad): a partir de una carta OM combinada con una carta BC, o con una carta SC.
- **Crypto-Set CS2** (Integridad + Autenticación): a partir de una carta MAC combinada con una carta H o BC, dependiendo del tipo de MAC.
- **Crypto-Set CS3** (Confidencialidad + Integridad + Autenticación): a partir de una carta AE combinada con una carta BC.

El objetivo del juego es conseguir un conjunto de cartas que permitan cubrir los tres objetivos de seguridad considerados, denominando dicho conjunto de cartas como *Crypto-Kit*. Con los *Crypto-Sets* recién definidos hay dos maneras de alcanzar dicho objetivo:

- **Crypto-Kit CK1**: Combinando dos *Crypto-Sets*, uno del tipo CS1 y otro del tipo CS2.
- **Crypto-Kit CK2**: Consiguiendo un *Crypto-Set* CS3.

Nótese que éste es un escenario muy simplificado, pues en esta versión inicial tomaremos dos hipótesis que no se corresponden al cien por cien con la práctica criptográfica real. Concretamente, consideraremos que:

- cualquier cifrador en bloque es válido para construir cualquiera de los *Crypto-Set* de tipo CS3.
- las cartas tipo MAC pueden combinarse con cualquier cifrador en bloque, a excepción de la carta HMAC que deberá combinarse con una carta de tipo H para poder completar un *Crypto-Set* de tipo CS2.

Completar un Crypto-Kit robusto. El objetivo de cada jugador será conseguir un *Crypto-Kit* robusto, es decir, una combinación de cartas que posibilite la construcción de un diseño criptográfico seguro, según las premisas descritas en las instrucciones del juego. Dicho *Crypto-Kit* puede formarse con distinto tipo y número de cartas, y no todos los diseños, siendo válidos, darán al jugador la misma puntuación.

Nivel de seguridad de las herramientas. El juego incluirá una tabla que asignará a cada herramienta criptográfica un color indicando su robustez. Así, las cartas a las que se les asigne el color verde serán las que representen herramientas cuya seguridad se considera elevada, el naranja será para las que alcanzan una seguridad media y reservaremos el rojo para aquellas cuya seguridad es altamente cuestionable. La asignación de colores a cada herramienta se realizará a partir de las recomendaciones de [2], identificando con color verde las herramientas etiquetadas con nivel de seguridad "Future" y con naranja las etiquetadas como "Legacy". Las cartas rojas serán aquellas cuya seguridad se considera en entredicho de manera aplastante, como la función resumen MD5, o el cifrador de flujo RC4. El uso de esta tabla en el juego permitirá que los jugadores interioricen el nivel de seguridad de las herramientas involucradas.

C. Mecánica del juego

El juego se articulará en rondas (considerando en un principio partidas de tres rondas) y ganará aquel jugador que haya obtenido mayor puntuación acumulando los puntos de cada ronda. Planteamos el juego con un máximo de 8 jugadores y una baraja de 108 cartas. De esta forma puede garantizarse la variabilidad en el juego, así como la aparición en cada partida de un amplio abanico de cartas distintas (véase Tabla I).

Tabla I: Número de copias de cada carta

Tipo de carta	BC	H	SC	OM	AE	MAC
Número de primitivas o construcciones en [2]	7	10	15	8	7	7
Copias de carta por primitiva o construcción en baraja	4-5	1-2	1	1-2	1	4
Número de cartas por tipo en baraja	32	13	15	13	7	28

Inicio de partida: Se baraja el mazo y se reparten a cada jugador 6 cartas. El resto se disponen boca abajo en un mazo en el centro.

Ronda: En cada turno normal cada jugador debe elegir una carta de su mano y la deja boca abajo en la mesa. A continuación, todos revelan la carta elegida simultáneamente y la colocan enfrente de cada jugador con el anverso a la vista (junto con las otras cartas seleccionadas). A continuación, cada jugador pasa al jugador sentado a su izquierda las cartas restantes de su mano boca abajo. De este modo, en el siguiente turno cada jugador empieza con una carta menos y una mano nueva. Este proceso se repite hasta que cada jugador recibe un mazo con una sola carta.

Entonces cada jugador tomará 2 cartas del mazo central y los incorporará a la mano. En este momento, además de seleccionar una carta como en los turnos normales, el jugador podrá sustituir hasta 2 de sus cartas ya jugadas, descartando las que sustituya en un mazo de descartes. Antes de pasar el mazo al siguiente jugador, por cada carta sustituida, se tomará otra carta del mazo central. Cuando se han jugado todas las cartas, la ronda se da por finalizada y se procede a calcular públicamente las puntuaciones de cada jugador.

Puntuando una ronda: Solo puntúan las cartas que conforman un *Crypto-Kit*. Por cada *Crypto-Kit*, el jugador acumula 10 puntos de los que se restan 1 punto por cada carta con clasificación naranja y 2 puntos por cada carta con clasificación roja de las incluidas en sus *Crypto-Kits*.

Fin de la partida y ganador(es) del juego: Gana el jugador con mayor puntuación tras tres rondas.

III. Trabajos Futuros y Conclusiones

Este diseño inicial de Crypto Go es muy útil para ayudar al alumno a afianzar los conocimientos básicos de criptografía simétrica impartidos en un curso de grado. Por supuesto, un primer paso a dar de cara a la validación de esta idea es hacer un prototipo sencillo y analizar la experiencia de varios grupos de alumnos al jugar. Entre nuestros planes está además el realizar una versión extendida en la que se puedan añadir cartas para, por un lado, incorporar nociones y destrezas relacionadas con la correcta gestión de claves en este ámbito y, por otro, afianzar conocimientos directamente relacionados con ataques concretos a las herramientas consideradas. Nuestra idea es que esta versión extendida, enfocada a alumnos de postgrado, sea compatible con la aquí presentada.

IV. Referencias

[1] Hendrix, M., Al-Sherbaz, A. & Victoria, B.: "Game based cyber security training: are serious games suitable for cyber security training?", en *International Journal of Serious Games*, vol. 3, n. 1, pp. 53-61, 2016.

[2] ECRYPT CSA: "D5.2 algorithms, key size and protocols report", *informe del proyecto 645421 (H2020-ICT-2014)*, 2016.

Plataforma de gestión de escenarios de ciberseguridad para aprendizaje y entrenamiento

Francisco Barea, Irene Romero, José Ignacio Rojo, Víctor A. Villagrá y Julio Berrocal
Departamento de Ingeniería de Sistemas Telemáticos, Universidad Politécnica de Madrid
Avda. Complutense, 30. 28040 Madrid.
Email: jfbarea@dit.upm.es, iromero@dit.upm.es, jirojo2@gmail.com, villagra@dit.upm.es, berrocal@dit.upm.es

***Resumen*—Estando el sector de la ciberseguridad en pleno auge, con una demanda de puestos de trabajo muy superior al número de profesionales cualificados en la actualidad, queda de manifiesto la importancia de la docencia de los nuevos profesionales del sector. Este artículo presenta una propuesta para proveer a los centros de enseñanza de una plataforma que posibilite la realización de prácticas docentes de laboratorio con escenarios complejos que cuentan con infraestructura dedicada al alumno para la resolución de cada ciberejercicio de forma aislada, optimizando el coste en infraestructura y operaciones necesario para su puesta en marcha. Para aislar los escenarios, se utilizarán contenedores Docker, con redes virtualizadas para dar conectividad a los diferentes contenedores que conforman el escenario. Cada equipo dispondrá de un entorno dedicado al equipo y aislado del resto de equipos para realizar el escenario. La plataforma deberá ser capaz de gestionar estos entornos y de asegurar su correcto aislamiento, minimizando la utilización de recursos en todo momento para facilitar la escalabilidad del sistema. Asimismo, este documento también incluye una propuesta de ejercicios a ser desplegados con la plataforma en formato de prácticas docentes de ciberseguridad donde el alumno se enfrentará a los conceptos fundamentales del hacking ético y las diversas categorías que debe dominar todo profesional del sector de la ciberseguridad.**

***Index Terms*—Formación, Competición, Contenedores, Docker, Laboratorios docentes, Entrenamiento, Ciberseguridad**

Tipo de contribución: Formación e innovación educativa (límite 8 páginas)

I. Introducción

I-A. Ámbito

Dada la necesidad de protección de los sistemas de información y redes de telecomunicación, el gran aumento de amenazas y ataques a estos entornos, y el aumento de la concienciación de seguridad de todos los actores involucrados, este documento se centra en el ámbito de la ciberseguridad, en concreto en metodologías alternativas de formación en este campo mediante el uso de los avances tecnológicos más recientes, primando la usabilidad y la escalabilidad, así como la optimización de recursos necesarios para llevar a cabo esta propuesta formativa en un laboratorio docente.

La ciberseguridad está cobrando especial importancia en las empresas del sector TIC, ampliando el mercado laboral de profesionales expertos tanto en hacking ético, análisis forense, bastionado de servicios, o consultoría en general.

La amplia demanda de estos días de profesionales del sector de la ciberseguridad y del hacking ético en particular ha revolucionado los métodos de formación tradicionales. La fuerte componente creativa, así como la amplia base técnica y tecnológica necesarias para el correcto desempeño de esta profesión penaliza los planes tradicionales y requiere un enfoque más práctico y directo.

Por esto mismo han cobrado especial importancia las certificaciones profesionales en este sector en particular. Estas certificaciones se suelen basar en un laboratorio práctico, con un examen final sobre los contenidos practicados en el laboratorio. Las certificaciones mejor valoradas difieren en contenidos, pero todas tienen en común el enfoque puramente práctico, siempre fomentando el pensamiento lateral de los profesionales, que en el fondo es la cualidad más importante de los mismos para este sector en particular. Los laboratorios prácticos asociados a las certificaciones profesionales cuentan con una infraestructura compleja, poco escalable y de muy alto coste, ya sea por la propia infraestructura o por el contenido docente que busca emular escenarios de ataque reales y que por tanto requieren de un profundo análisis técnico.

I-B. Objetivo

Aunque en ciertos escenarios puede permitirse asumir el alto coste de los laboratorios típicos de las empresas especializadas en certificación profesional, replicar dichos entornos con un fin educativo en una universidad o en un laboratorio interno de una empresa u organización resulta a día de hoy lejos de ser óptimo. Por esta misma razón esta propuesta busca la innovación en la infraestructura necesaria para realizar sesiones docentes de hacking ético mediante la utilización de nuevas tecnologías que se irán definiendo en los siguientes puntos de este documento, siempre con el fin de optimizar la escalabilidad y los costes necesarios para la realización de sesiones de entrenamiento en un laboratorio docente. El objetivo de este documento es fomentar la educación práctica en ciberseguridad, proporcionando una plataforma de entrenamiento escalable y con capacidad y flexibilidad suficientes para el diseño de ciberejercicios complejos que pongan a prueba las habilidades y competencias de los alumnos en un ámbito de laboratorio. La plataforma provee de ciberejercicios basados en escenarios que pondrán a prueba a los alumnos y facilitará su aprendizaje con un enfoque práctico. Asimismo, la plataforma facilita el seguimiento de los progresos de los alumnos por parte de los docentes, tanto con un sistema de puntuación general, fomentando la competitividad entre los participantes, como con un sistema de calificaciones personalizado a disposición del docente. Todo este sistema de puntuación se encuadra en el marco de la gamificación, enfoque que se ha probado eficaz a la hora de potenciar el compromiso del alumno en el aprendizaje[1]. Y no menos importante, se busca un diseño modular del contenido, en este caso los ciberejercicios a realizar por los alumnos. En este aspecto la plataforma busca la facilidad

de incorporación de nuevos ciberejercicios en sus diferentes modalidades, pudiendo ser diseñados por los propios docentes, o incluso propuestos por los estudiantes, fruto de sus estudios en la materia, para ser resueltos por sus compañeros en un modelo colaborativo.

I-C. Estructura del artículo

En esta memoria se define una propuesta diseño y desarrollo de una plataforma de gestión de escenarios de ciberseguridad para aprendizaje y entrenamiento.

La segunda sección describe el estado actual de la formación en materia de ciberseguridad, tanto títulos universitarios de grado o máster, ya sean oficiales o propios, certificaciones profesionales o competiciones prácticas relacionadas.

En la tercera sección se detalla la arquitectura de la plataforma que dotará de infraestructura a los ciberejercicios y servirá de interfaz de usuario.

En la cuarta, se definen los requisitos de los ciberejercicios para que se puedan albergar en la plataforma, así como recomendaciones y buenas prácticas a la hora de diseñar los ciberejercicios y los escenarios que los componen. Se detallan los tipos de entorno que pueden conformar los escenarios y alternativas en el caso de querer expandir las características fuera del entorno de virtualización que propone la plataforma.

La quinta sección contiene una propuesta de casos de uso.

Por último, la sexta y séptima secciones cierran el artículo comentando posibles líneas de continuación del trabajo y las conclusiones finales del mismo respectivamente.

II. Estado de la formación en ciberseguridad

El mapa actual de formación en ciberseguridad cuenta con una oferta muy diversificada, desde asignaturas de grado hasta másteres especializados, pasando por certificaciones profesionales, talleres y competiciones.

II-A. Formación universitaria

En cuanto al material impartido en las asignaturas de grado oficiales se pueden encontrar asignaturas de especialidad que conglomeran los principios básicos de la ciberseguridad, con el objetivo de conocer y aplicar las tecnologías que proporcionan seguridad a los sistemas y redes TIC, conociendo los fundamentos organizativos y criptográficos en los que se basan las tecnologías de seguridad y su aplicación a la programación y desarrollo de software seguro.

En las asignaturas de grado relativas a la ciberseguridad en España se trabajan temarios relativos a criptografía, firma electrónica, programación segura, amenazas de internet, servicios de control de acceso y servicios de protección de la comunicación.

A nivel de máster universitario, existen tanto títulos oficiales como títulos propios o bien elaborados por la propia universidad.

Los másteres oficiales especializados en ciberseguridad profundizan los conceptos introducidos en las asignaturas de nivel de grado. Cuentan con asignaturas de amenazas y contexto de la ciberseguridad, servicios de control de acceso, servicios de seguridad en red, protección de sistemas y servicios, protección de la información, ingeniería inversa y análisis de malware, gestión de riesgos y operaciones, privacidad, evidencias forenses, seguridad en el desarrollo de software, diseño de estrategias corporativas relativas a la ciberseguridad, sistemas de gestión de seguridad de la información y seguridad física. Adicionalmente, introducen una experiencia más práctica familiarizando al alumno con los procedimientos de auditoría y se complementan con prácticas en empresas del sector.

Las asignaturas tanto de grado como de máster cuentan, o sería idóneo que contaran, con componentes prácticos en formato de prácticas de laboratorio docente.

II-B. Certificaciones profesionales

Por otro lado, las certificaciones profesionales varían en ámbito y dificultad. Su reconocimiento profesional también varía en función de la región, siendo CEH la certificación más reconocida a nivel Europeo y Africano, y OSCP/OSCE las certificaciones más requeridas a nivel Americano. Las certificaciones profesionales están gestionadas por organizaciones de acreditación independientes, como ComTIA, EC Council, GIAC, ISACA, (ISC)2 y Offensive Security.

Típicamente, las certificaciones se dividen en tres segmentos: nivel de entrada, nivel intermedio y nivel experto, y todas ellas constan típicamente de un periodo de entrenamiento culminado en un examen final. Las certificaciones de nivel de entrada están orientadas a exponer al alumno a los conceptos fundamentales, mejores prácticas, herramientas esenciales, últimas tecnologías y metodologías.

Algunas de estas certificaciones, como es el ejemplo del OSCP y el OSCE, centran su evaluación en un componente fundamentalmente práctico, dando acceso durante un periodo de entrenamiento a un laboratorio con una serie de escenarios donde el alumno puede practicar y obtener la técnica propuesta en ese ejercicio.

II-C. Competición

Esta práctica se asemeja al formato de competición de tipo CTF o capturar la bandera por sus siglas en inglés. Capture the Flag (CTF) es un tipo de competición de ciberseguridad que cuenta con dos modalidades principales: Jeopardy y Ataque-Defensa, pudiendo considerarse una opción híbrida entre ambas.

Jeopardy en este caso se trata de un formato en el que hay una serie de retos o ejercicios agrupados por categorías, y los participantes obtienen puntos por la resolución de estos retos. Los participantes pueden organizarse por equipos si la organización así lo permite.

Para la modalidad de Ataque y Defensa se organizan los participantes por equipos, donde cada equipo dispone de una infraestructura dedicada (servidores, red, etc) con una serie de activos vulnerables. El objetivo de cada equipo es doble: arreglar sus vulnerabilidades y atacar al resto de equipos, ya que los puntos se dividen en las categorías de ataque (se obtienen puntos por atacar otros sistemas) y defensa (obteniendo puntos por no ser atacado, ya sea por no ser vulnerable o por no haber sido objetivo de ningún ataque)

Para la realización de prácticas docentes de laboratorio, el formato Jeopardy de competición CTF resulta particularmente atractivo dada su facilidad de despliegue y operación, y va a ser este mismo formato el propuesto en este documento. Una posible contraposición sería el coste y la complejidad de la infraestructura técnica para desplegar los escenarios que

componen los ejercicios, por lo que será el factor que se trata de optimizar en la plataforma propuesta.

II-D. Plataformas disponibles

Por un lado existen una considerable cantidad de soluciones gestionar infraestructuras que soporten laboratorios docentes sobre ciberseguridad, en [3], [4], [5], [**?**] se muestran varios ejemplos. El problema que acarrean estas soluciones es que están orientadas a virtualización, lo cual exige un alto coste en infraestructura que se pretende subsanar con la solución propuesta. Por un lado existen diversas plataformas para competición que no proporcionan una infraestructura sobre la que ejecutarlas, a continuación se listan unos ejemplos de dicho tipo de plataformas.

- **Mellivora:** escrita en PHP y publicada en Github [7]. Esta solución provee de una plataforma web que permite la gestión de los retos, la presentación de los mismos a través de un enunciado y un tablón de puntuaciones y la validación de los resultados.
- **CTFd:** también disponible en Github [8] y escrito en Python. La capa de presentación cumple las expectativas que se esperan de este tipo de plataformas, pero tampoco tiene una capa de gestión de infraestructura, que será la diferencia que propone la plataforma diseñada en este documento.
- **FBCTF:** también disponible en Github [9], con una plataforma web que se basa en un mapa mundial donde cada reto representa un país. De nuevo la capa de presentación cumple las expectativas, pero no cuenta con una gestión de la infraestructura.

En cuanto a las soluciones existentes en el mercado que cubren cuentan gestión de la infraestructura se venden típicamente como servicio. Existen empresas como HackingLab o Indra, que cuentan con plataformas y servicios de organización de eventos formativos o de competición. Sus soluciones se basan en tecnologías de virtualización, que suponen un requisito importante de hardware.

II-E. Solución propuesta

En definitiva, la solución propuesta en este artículo cuenta con una plataforma web capaz de orquestar y presentar los ejercicios a los participantes, incluida la infraestructura necesaria para desplegar los escenarios, así como dotar de herramientas de seguimiento y control al personal docente. La infraestructura de los diferentes escenarios que componen los ejercicios que se ofertarán a los alumnos en el laboratorio viene dada también por la plataforma, a través de tecnologías de virtualización y contenerización. Poder contar con infraestructura para los escenarios, permite un diseño más complejo de los ejercicios, eliminando los problemas de compatibilidades y de requerimientos para los participantes y permite aislar en caso de análisis de muestras de malware. En definitiva, la solución propuesta en este documento serviría para poder ofertar prácticas docentes de ciberseguridad para cubrir el itinerario expuesto en este documento, pudiendo cubrir prácticas de criptografía, certificados digitales, configuración y prueba de funcionamiento para software de defensa perimetral, análisis de muestras de malware, auditoría, ingeniería inversa, análisis forense, etc. En los siguientes puntos se entrará en el detalle de la arquitectura del sistema propuesto, y del formato de los ejercicios y escenarios que pueden ofrecerse con esta solución.

III. Arquitectura del sistema

La arquitectura que va a seguir el sistema prima la escalabilidad a la hora de gestionar recursos virtualizados para poder lanzar los retos, respetando el nivel de aislamiento que requieren.

III-A. Stack tecnológico

Las tecnologías utilizadas para desarrollar la plataforma propuesta se resumen en la figura 1

Figura 1. Tecnologías que componen las solución.

Para dotar de infraestructura al proyecto, se utilizan contenedores docker de diferentes tipos. Para un entorno sencillo se puede utilizar un único nodo docker con varios contenedores, pero para un despliegue productivo se recomienda una configuración en modo swarm (clúster de docker) con al menos tres nodos, facilitando así el consenso en caso de fallo simple.

Los servicios fundamentales de la plataforma se reducen a un servicio de base de datos, con un contenedor basado en una imagen Debian con MongoDB, y a un servicio web también desplegado en un contenedor basado en una imagen Debian con NodeJS que albergará una API REST y dos aplicaciones frontales basadas en AngularJS.

El framework sobre el que se basa la plataforma es NodeJS, un entorno en tiempo de ejecución multiplataforma y de naturaleza asíncrona. Se asocia con conceptos de alta escalabilidad, ya que su arquitectura asíncrona basada en eventos permite exprimir al máximo el rendimiento en procesos típicos de servicios de API en el que prima la métrica de peticiones atendidas por segundo.

La capa servidor pues, implementa una API REST que alimenta dos aplicaciones web basadas en AngularJS y se nutre de una fuente de datos documental MongoDB.

La implementación base para las funcionalidades HTTP elegida es ExpressJS, que dota al sistema del protocolo HTTP y el modelo REST, así como un potente enrutador basado en el concepto de middleware [10].

Para el acceso a datos se utiliza el ODM (Object Document Mapper) Mongoose, que dota de una capa de abstracción sobre el motor a bajo nivel que ofrece MongoDB.

La capa frontal o frontend estará compuesta por dos aplicaciones SPA (single page application) escritas con el framework AngularJS. Ambas aplicaciones se alimentan de la API REST que se ha visto en el comienzo de esta sección.

La primera aplicación gestiona toda la interfaz de usuario de la plataforma de entrenamiento, mientras que la segunda gestiona la parte del panel de administración de la plataforma.

III-B. Arquitectura Lógica

La arquitectura lógica seguiría el diagrama de la figura 2, con dos tipos de usuario bien diferenciados: los participantes o alumnos, que irán contra la aplicación AngularJS principal, y los admin o docentes, que utilizarán el panel de administración, que es una aplicación AngularJS diferente aunque ambas se alimenten del mismo servicio de API.

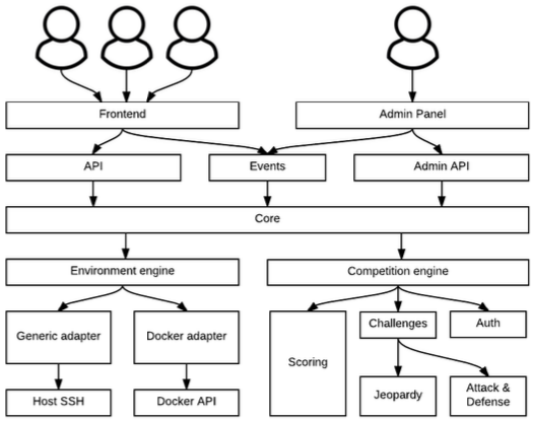


Figura 2. Arquitectura lógica de la plataforma

La arquitectura física se ilustrará con un ejemplo concreto ya que la solución propuesta tiene entre sus objetivos la flexibilidad de adaptarse al entorno del cliente final.

El sistema, por tanto, se compondrá de varias partes bien diferenciadas:

- Un conjunto de LANs para los usuarios. El escenario más común será aquel en el que los usuarios de la plataforma se dividan en equipos. Por poner números de ejemplo, se supone un total de diez equipos, cada equipo compuesto por diez integrantes, todos ellos situados en un laboratorio docente. En este caso particular se contaría con diez sendas LANs, aisladas entre sí y enrutando el tráfico hacia el punto de entrada a la plataforma, pero nunca hacia las LANs de los otros equipos.
- Después de estas LANs de equipos, estaría la LAN de la plataforma. Lo primero que se encuentra en este segmento de red será un balanceador de carga (ya sea nginx, haproxy o traefik) que repartirá las peticiones entre los servidores de la plataforma correspondientes. Después, estarían los servidores de la plataforma (dos, para este ejemplo), que sirven tanto el frontend (artefactos compilados de la aplicación SPA escrita en AngularJS) como la API REST y los Websockets que componen el servicio de la plataforma.
- Además de los servidores web, se disponen nodos Docker configurados en modo swarm (cluster de docker) para la virtualización de los retos. En este escenario concreto no se hace uso del adaptador genérico de entornos virtualizados descrito en la figura de arquitectura, sino que se usará únicamente Docker. Ya que el motor de virtualización de escenarios se comunica con el cluster docker mediante su API REST, no es relevante el número de nodos docker aprovisionados (aunque para el ejemplo expuesto se supondrá que hay un total de 5 nodos para servir a los 100 participantes)
- Finalmente se cuenta con un nodo de base de datos MongoDB, escalable a un cluster compuesto por varios nodos de ser necesario, para almacenar todos los datos relacionados al backend del sistema.

III-C. Gestión de la infraestructura de los escenarios

Para la gestión de la infraestructura se pretende utilizar un método de virtualización lo más ligero posible, con el fin de simplificar al máximo esta funcionalidad y contener los costes de infraestructura.

Aunque la plataforma permitirá diferentes soluciones para proveer la infraestructura, este artículo se centra en un método de virtualización ligera basada en contenedores Docker [2].

Docker utiliza características de aislamiento de recursos del kernel de Linux, tales como cgroups y espacios de nombres (namespaces) para permitir que ”contenedores” independientes se ejecuten dentro de una sola instancia de Linux, evitando la sobrecarga de iniciar y mantener máquinas virtuales.

Otras soluciones disponibles en el mercado utilizan virtualización convencional mediante un hypervisor ya sea nivel 1 o nivel 2, lo cual supone asignar recursos fijos desde el inicio para todos los participantes y los escenarios, y cada instancia consumirá muchos más recursos que la contrapartida basada en contenedores, ya que cada instancia tendrá que disponer de su propio kernel y de su propio sistema operativo completo.

La ventaja principal de Docker es su ligereza, y dado que los contenedores que se utilizarán en los ciberejercicios están diseñados para ejecutar con un usuario distinto de root, se previenen posibles fugas del entorno aislado o sandbox. Adicionalmente, Docker permite la virtualización de redes y conectividad entre contenedores, pudiendo diseñar entornos con varios contenedores, totalmente aislados.

IV. Diseño de la plataforma

Como ya se ha expuesto en la sección anterior, la plataforma de entrenamiento se divide en dos aplicaciones funcionales o interfaces de usuario:

- Un **panel de control** donde definir los equipos, participantes, ciberejercicios, sesiones, reglamento, etc., y donde también se puede llevar a cabo un seguimiento de la sesión en curso (y anteriores) y generar reportes personalizados.
- La propia **plataforma de entrenamiento**, que proveerá a los participantes de un medio donde obtener y validar los ciberejercicios.

La modalidad de los ciberejercicios será CTF, donde cada ciberejercicio tendrá como objetivo la obtención de un token o bandera que la plataforma valida como solución para obtener los puntos correspondientes y dar por finalizado el ciberejercicio.

Cada ciberejercicio se compondrá de los siguientes campos:

- Un título o nombre del mismo, junto con una categoría que agrupe los retos de la misma tipología y una descripción detallada del ciberejercicio conteniendo su enunciado, guiando al participante durante la resolución del mismo.

- Una solución del reto o flag, que no será visible para el participante más allá del formulario correspondiente de validación para resolver el ciberejercicio.
- Un entorno virtualizado, si procede, asociado al ciberejercicio. El propósito y los diferentes tipos de entornos virtualizados se explicarán en detalle a continuación.
- Archivos adjuntos al enunciado a ser gestionados por la plataforma, o bien incluir enlaces a sistemas de distribución de ficheros externos en el propio texto del enunciado.

Todo ciberejercicio tendrá definida una puntuación bruta del reto. A esta puntuación se le podrán descontar el valor en puntos de una serie de pistas, canjeables por los participantes a cambio de información extra para la resolución del reto. Cada ciberejercicio puede tener cero o varias pistas, por un valor combinado de puntos menor o igual a la puntuación bruta de puntos del mismo.

Las categorías disponibles en la plataforma dependen de la definición realizada por el equipo docente que haya preparado el paquete de ciberejercicios, aunque se sugieren las siguientes:

- Análisis web
- Análisis forense
- Criptografía
- Ingeniería inversa
- Exploiting

IV-A. Entorno virtualizado

Cada ciberejercicio puede tener asociado un entorno virtualizado, que proporcionará al participante una infraestructura específica para la resolución del mismo.

La plataforma está preparada para dar soporte a varios tipos de entornos virtualizados, pero este artículo se va centrar en Docker, que ya se ha introducido en la sección III.

Se diferencian los tipos de entornos virtualizados mediante Docker en dos tipos principales: Entorno simple o entorno complejo.

IV-A1. Entorno simple: Los entornos simples tienen un único contenedor y por tanto no requieren de virtualización de redes ni de orquestación de contenedores aislados. En este caso implementan un Dockerfile que, a partir de una imagen base, construye un servicio vulnerable para que el equipo participante lo comprometa y obtenga el token o bandera objetivo, para su posterior validación en la plataforma de entrenamiento descrita en este documento.

En el diagrama de la figura 3 se puede ver cómo el usuario obtiene los detalles de conexión al entorno virtualizado del escenario simple del ciberejercicio. En este caso el usuario se conectará al puerto asignado para ese entorno virtualizado en concreto, puerto 20501 en este ejemplo, que la plataforma conectará internamente al puerto 1337 del contenedor que albergará el entorno del reto. La conectividad interna es transparente para el usuario y está orquestada por la plataforma de ciberejercicios, proporcionando al usuario la información necesaria para que pueda conectar con su instancia del entorno virtualizado. Algunas instancias requieren de autenticación, como puede ser un servicio ssh o un servicio git, que serían provistas también por la plataforma al usuario.

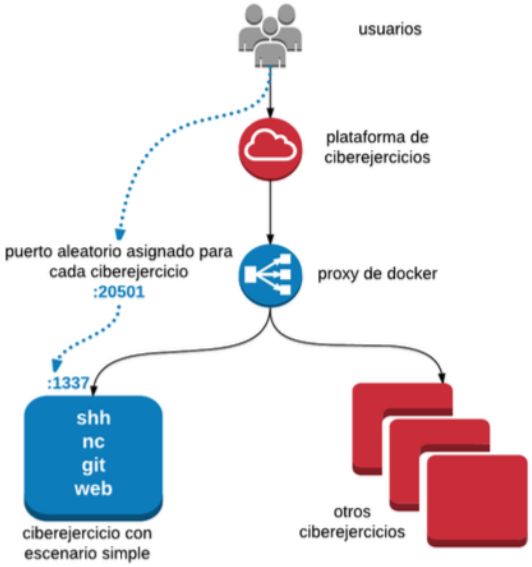


Figura 3. Diagrama de ejemplo de un entorno simple

IV-A2. Entorno complejo: Los entornos complejos aprovechan las características de virtualización de redes que brinda Docker. El entorno expondrá un servicio desde un punto de entrada o entrypoint. Este punto de entrada estará albergado en el contenedor principal del escenario, y tendrá conexión con la red general de Docker, por la que el participante accederá al servicio.

Como se ilustra en el diagrama de la figura4, Tras el contenedor principal, el escenario albergará más contenedores, conectados entre sí por, al menos, una red interna de Docker. El objetivo a la hora de diseñar estos escenarios es poder añadir complejidad a los mismos y simular saltos de redes diseñadas en capas.

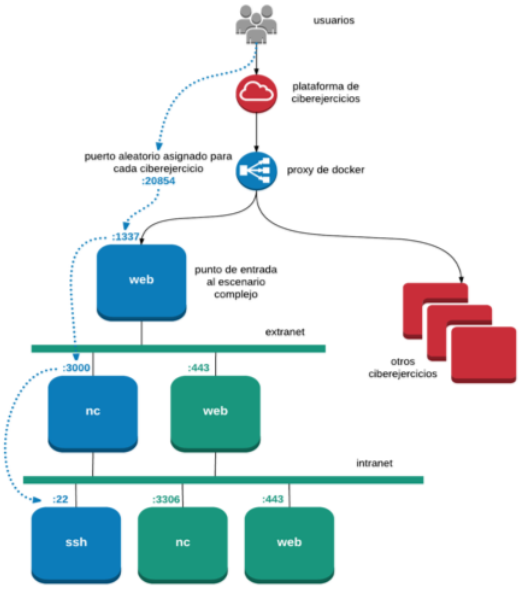


Figura 4. Diagrama de ejemplo de un entorno complejo

V. Diseño de los ciberejercicios

Los ciberejercicios deberán diseñarse con una serie de reglas para asegurar la compatibilidad con la plataforma. Estas reglas afectan especialmente a aquellos ciberejercicios que requieren de un entorno virtual para funcionar.

Aunque la plataforma está diseñada para dar soporte a otros métodos de virtualización de entornos, esta propuesta se centra en las especificaciones del entorno Docker. Por lo tanto, los ciberejercicios propuestos en este en la sección VI, estarán todos ellos adaptados para ser acoplados a la plataforma con el entorno Docker.

V-1. Especificaciones del proveedor de entornos virtualizados basado en Docker: Estas especificaciones son las reglas mencionadas al principio de esta sección y son:

1. Todo entorno virtualizado con Docker requiere exponer un único servicio al exterior. Además, el puerto de dicho servicio expuesto, deberá ser internamente a nivel contenedor el puerto 1337. La plataforma se encargará de enrutar dicho puerto y de exponer al usuario el puerto correspondiente en función a las estrategias de enrutado. Para el resto de servicios que pueda albergar el entorno virtual para ese ciberejercicio no se aplicaría esta restricción, ya que se encuentran en los segmentos internos de red del ciberejercicio en cuestión, que están totalmente aislados del resto de instancias.
2. Utilizar la misma imagen base para todos los retos, siempre que sea posible. Esta especificación no es de obligado cumplimiento, pero ayuda en gran medida a optimizar el consumo de recursos de la plataforma en su conjunto. Para el desarrollo de este documento se ha elegido la imagen base **debian:9** y derivados, primando siempre este tipo de imagen. En definitiva, es preferible evitar utilizar una imagen base diferente para cada reto, ya que esto supone almacenar todas esas imágenes base en cada nodo del swarm de docker.

La definición de los escenarios, o ciberejercicios complejos, se realizará mediante el panel de control de la plataforma. Además, debido a limitaciones diseño de la propia plataforma no se pueden utilizar herramientas como docker-compose ni stacks ni servicios de docker.

VI. Casos de uso

En esta sección se expondrá una serie de casos de uso de esta plataforma en forma de propuestas de ciberejercicios básicos para la modalidad de Jeopardy de varias categorías clave del *hacking* ético.

El objetivo de estos ciberejercicios es educar a los participantes en los aspectos básicos de cada categoría, fomentando la participación y el trabajo en equipo para superar una serie de retos complejos. Cada uno de ellos tiene asociada una puntuación que está integrada en un sistema de puntos que engloba toda la plataforma. Mediante el conteo de estos puntos, se puede organizar una competición entre los distintos puntos de usuario.

VI-A. Criptografía

Este tipo de retos son simples problemas de criptografía propuestos para ser resueltos por los usuarios de la plataforma. Un ejemplo de este tipo de ejercicios es el que se representa en la figura 5. En dicha figura se presenta una captura de la interfaz de la plataforma propuesta en este documento. En este caso se trata de un reto bastante simple que consta de un enunciado escrito y un campo con la respuesta pertinente. Resulta evidente que para este tipo de ejercicios no se necesita de la infraestructura de Docker para su ejecución.

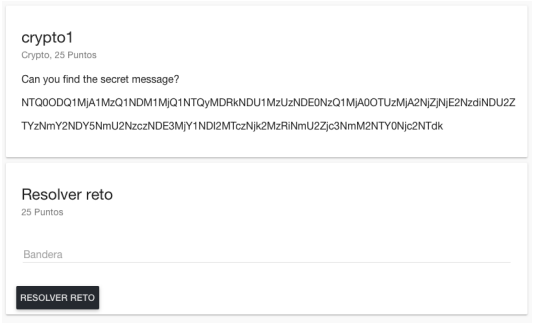


Figura 5. Ejemplo de reto criptográfico

VI-B. Exploiting

En la figura 6 se muestra un ejemplo de ejercicio sobre *exploiting*. Como se puede ver, este ejercicio sí que requiere un entorno virtual, de modo que sí se hace uso de la infraestructura de Docker.

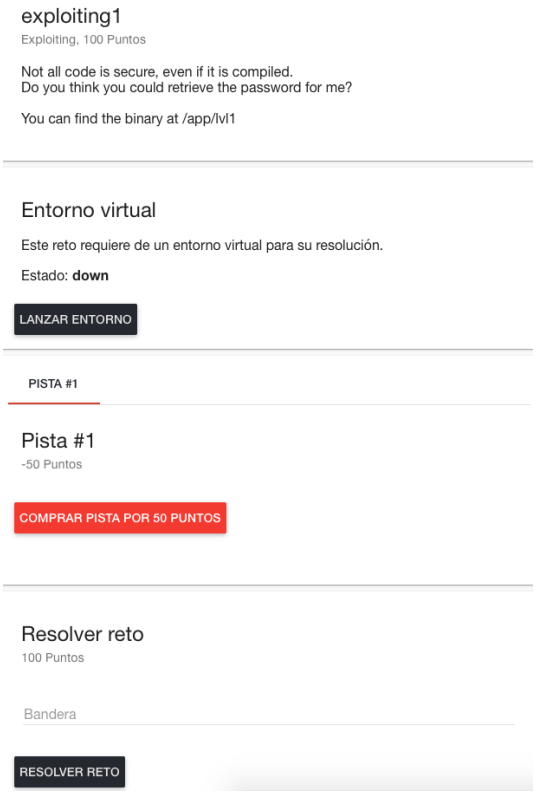


Figura 6. Ejemplo de reto sobre exploiting

Una vez lanzado el entorno virtual, la plataforma proporciona los datos de acceso al mismo como se muestra en la figura 7

Figura 7. Datos de acceso para entorno virtual

El objetivo de este ejercicio es obtener nociones básicas de las diferentes herramientas que existen para analizar datos binarios, obteniendo información a partir de las cadenas de texto potencialmente reconocibles en un bloque binario utilizando la herramienta *strings*.

Como se puede ver en la figura 6, es posible obtener pistas para resolver el ejercicio a costa de disminuir la puntuación conseguida en el ejercicio. Esto aporta otro matiz más de gamificación a la plataforma.

VI-C. Análisis Web

Dentro de la disciplina del análisis web, este ejemplo versa sobre inyección SQL. Como se puede ver en la figura 8, en este caso, cuando se lanza el entorno virtual, se despliega un contenedor que aloja un servidor web que el participante tendrá que atacar mediante una inyección SQL. Como se muestra en la figura, la plataforma proporciona una URL que apunta hacia la aplicación web objetivo.

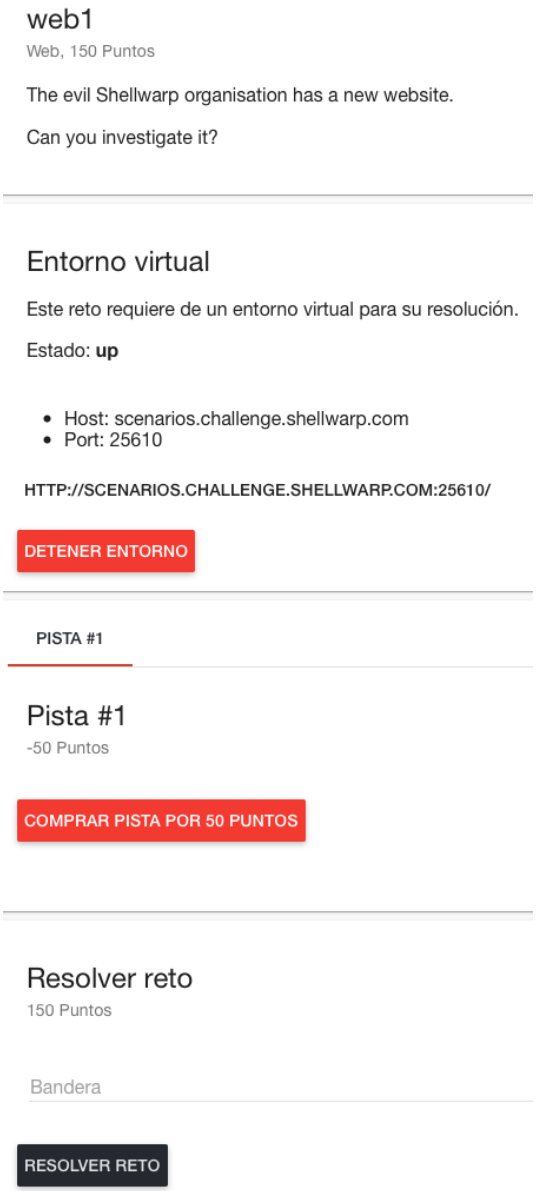


Figura 8. Ejemplo de reto sobre inyección SQL

VI-D. Pivoting

Usando la plataforma propuesta también es posible desplegar escenarios complejos como el de la figura 9.

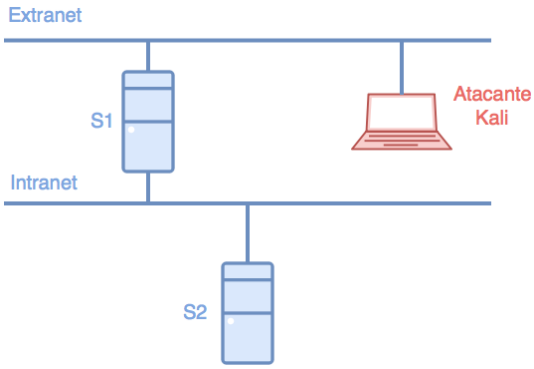


Figura 9. Escenario para ejercicio de Pivoting

Este ejercicio consiste en acceder al servidor S2 desde la máquina atacante usando el servidor S1 como pivote. Una vez

más, la plataforma desplegaría este escenario con dos redes y tres máquinas, proporcionaría los credenciales de acceso de la máquina atacante y esperaría un flag que se encontrase en la máquina objetivo (en este caso S2)

VII. Trabajo futuro

Uno de los trabajos futuros relacionados con esta plataforma sería incluirlo en un proyecto de innovación educativa donde se implante como herramienta de trabajo en una asignatura relacionada con la ciberseguridad de titulación oficial y medir el aumento de la productividad y satisfacción de los alumnos con respecto a las herramientas usadas en la actualidad en este tipo de asignaturas.

Por supuesto, otra línea de trabajo interesante sería aumentar la colección de ciberejercicios contenidos en la plataforma e integrarlo en los temarios de distintas asignaturas que se impartan en las diversas titulaciones relacionadas de alguna manera con la ciberseguridad.

Actualmente, la aplicación web está dividida en dos SPAs independientes (una para los ”administradores” y otra para los ”estudiantes”).Una continuación del trabajo relacionado con esta plataforma podría ser integrar estas dos aplicaciones en una sola con los roles mencionados bien diferenciados.

Otra mejora a la herramienta propuesta sería integrarla con una interfaz gráfica que permita la visualización de escenarios complejos. Un ejemplo de dicha interfaz se puede ver en [11]

Además de Docker, se podrían desarrollar adaptadores a otros proveedores de virtualización o compartimentalización, o simplemente a entornos hardware dedicados, ya sean pequeños arduinos, drones, dispositivos fabricados a medida o grandes sistemas SCADA. Las principales casuísticas serían de adaptación de dispositivos *IoT* controlados por radio, ya sea *WiFi*, *ZigBee*, *BlueTooth* u otras tecnologías comunes del mercado.

VIII. Conclusiones

Una de las conclusiones finales de este documento es que la concienciación es fundamental a hora de combatir las amenazas y riesgos de internet, pero que no sólo basta con concienciar a la población, sino que es necesario un entrenamiento específico de los profesionales que se dedicarán al sector de la ciberseguridad.

Para facilitar el entrenamiento, o en su caso el aprendizaje básico, los escenarios virtualizados resultan especialmente útiles y prácticos, aunque repercutirán directamente en el coste de infraestructura y operaciones de la solución tecnológica que se decida utilizar para la realización de los ejercicios. Es por esto que la propuesta presentada en este documento se ha focalizado en utilizar nuevas tecnologías para optimizar dicho coste reduciendo drásticamente los requisitos de hardware y de operaciones respecto a otras soluciones que puedan estar actualmente en el mercado.

Referencias

[1] Muntean, Cristina Ioana.”Raising engagement in e-learning through gamification.” en Proc. 6th International Conference on Virtual Learning ICVL. vol. 1. sn, 2011.

[2] Merkel, Dirk. ”Docker: lightweight linux containers for consistent development and deployment” en Linux Journal. vol. 2014. n. 239., 2014

[3] Nake, Niraj B., and Pushpanjali Chouragade. ”Review on Virtual Laboratory in Network Security Education.” en IJCA Proc. on National Conference on Recent Trends in Computer Science and Engineering, MEDHA 2015, n.1, pp. 40-28, 2015.

[4] Willems, Christian, and Christoph Meinel. ”Practical network security teaching in an online virtual laboratory.” en Proc. 2011 Intl. Conference on Security and Management (SAM 2011). 2011.

[5] Kunnath, Tony B., and Ashok Babu. ”Cloud Based SDN-Lab for Network Security Education.” en International Journal of Computer and Mathematical Sciences vol. 4, n. 12, 2015.

[6] xu, Le; huang, Dijiang; tsai, Wei-Tek. ”Cloud-based virtual laboratory for network security education”. IEEE Transactions on Education, 2014, vol. 57, no 3, p. 145-150, 2014

[7] Nakiami/mellivora. GitHub. 2018. Available at: https://github.com/Nakiami/mellivora. Accessed March 7, 2018.

[8] CTFd/CTFd. GitHub. 2018. Available at: https://github.com/CTFd/CTFd. Accessed March 7, 2018.

[9] facebook/fbctf. GitHub. 2018. Available at: https://github.com/facebook/fbctf. Accessed March 7, 2018.

[10] Bernstein, Philip A., en ”Middleware: A Model for Distributed System Services”,vol. 39, n. 2, pp. 86-98, 1996.

[11] Ruiz, Ricardo; Barea, J. Francisco; Álvarez-Campana, Manuel; Vázquez, Enrique ”Herramienta gráfica de configuración de escenarios virtuales para ejercicios de ciberdefensa” en III Jornadas Nacionales de Investigación en Ciberseguridad (pp 134-141), 2017

Third best student paper award "Study on privacy of parental control applications"

Álvaro Feal Fajardo
IMDEA Networks / IMDEA Software
Madrid, Spain
alvaro.feal@imdea.org

***Abstract*—Parental control applications are one kind of mobile software programs used by parents to monitor and control the use that their kids make of their cellphone. In this study we aim to learn about the privacy of these applications. We combine manual analysis of the Android marketplace with dynamic analysis of the application itself to learn about the ecosystem of these apps and their privacy problems. We studied 17 different applications of which 10 where fully analyzed using dynamic analysis. We found 11 privacy issues in 8 of these 10 apps. We also found that 17% of the permissions requested by these apps are not clearly mapped to any app functionality and that 50% of the applications do not clearly explain their communication model to users.**

***Index Terms*—Privacy, Android, Parental control applications**

Contribution type *Third best student paper award (extended abstract)*

I. Parental Control Apps

Parental control apps are used by parents to keep track of their child's activities on the internet. Usually, parents install the application in the kid's phone and can later configure a set of rules to dictate what their child can and cannot do. The privacy threats on these type of applications are huge, not only because they gather all kinds of private data, but also because these data are sent over the Internet towards third party servers. Also, the number of downloads on Google Play for the apps in our study shows that these types of apps are widely used. Nevertheless, to the bests of our knowledge there has not been a previous study on this type of mobile applications.

A. Threat model

Due to this communication model, there are several points where privacy problems may occur and lead to private data leaks. When sending information from the phone to the Internet, communications must be secure. Otherwise, an observer on the network could intercept the communication and gain access to private data. Therefore, all communications that include private data must be encrypted using TLS. The other way privacy leaks may occur is by the mishandling of these data by the controller of the software. If they do not treat data correctly in any of the points were these data is handled, then privacy of the user is compromised.

B. Understanding the ecosystem

We study a shortlist of 17 apps using different measurements:

Google Play Store data allows us to learn that these apps have a number of installs that revolves around 100.000 to 500.000 unique downloads. The mean user rating is around 3.4 out of 5.

License and pricing wise, there are three main business models used by these apps. Free, meaning that the whole app is free to use; Premium, where all of the features are pay to use; and Freemium, where there is a core of free features but users that pay get access to more features.

Parental control features allow us to separate these apps in blocking and monitoring apps. The first type only does active blocking of some of the phone's features, while the latter also keeps track of phone activity in order to show this info to parents.

Our shortlist consist of *14 monitoring apps* which we will further analyze, plus *3 blocking apps* for completeness.

II. Communication model

This type of applications are installed on the children's phone. Nevertheless, parents can later access data logs and blocking settings from other devices, such as their own phone or any device with a web browser. This behavior tells us that this type of applications must send private data to Internet server, where it will be stored, processed and later accessed by parents. The problem with this model is that when the app description does not clearly explain this model, non expert users such as parents will not know that their child's data is leaving their phone and going through the Internet. We study the descriptions provided in Google Play for the 14 monitoring apps in our shortlist. For this, we manually check if the developers clearly explain this behavior to parents, by searching in the text description for words such as "server", "Internet communication" or "sending of information". Our results show that only 50% of the studied applications clearly explain that private data will leave the phone.

III. Permissions study

Overprivilege in Android applications is an issue that has been broadly studied. [1][2][3]. Applications tend to request more permissions than those necessary for the correct functioning of the software. In most of the cases this is due to lack of understanding of the Android permission, reusing code found in Internet tutorials, unfinished features and requesting permissions for related methods to the ones that are being used. It is safe to say than in most cases, an unnecessary permission request means a developer error and not a malicious attempt to get access to phone resources.

Table I
Percentage of permissions that can be mapped to the functionality of the app

Necessary	Maybe necessary	Unnecessary
83%	3.5%	13.5%

We study the permissions requested by the 14 monitoring applications and manually match this to their advertised features and to the runtime behavior. When there is a clear correlation between a requested permission and one functionality, we mark this as "Clear". When these correlation is not clear or we can't be sure if the permissions is necessary we tag this as "Maybe necessary". Finally, when a permission is clearly unmatched with any feature, we mark it as "Unnecessary". Our results show that 11 of the 14 apps request permissions that are not necessary. Table I shows the results of our study, where out of all the permissions requested for the applications, only 83% are clearly necessary.

IV. Privacy threats

To learn about the privacy problems present in these apps, we dynamically analyze them using *TaintDroid*. This is a tweaked version of Android that identifies private information during runtime and tags it. This tag is attached to the data during the flow of the execution, and if it ever reaches a data sink (i.e., network interface and phone storage), TaintDroid shows that the tagged data has been leaked. We run TaintDroid in an Android emulator, which leads to only 7 of the 14 monitoring apps to work. Therefore, this analysis has been applied to 7 monitoring apps and 3 blocking apps.

A. Private data mishandling

To study these apps, we individually install each one in the TaintDroid emulator and follow a set of actions that aim to trigger data leakage. These actions are: installation and setup, Internet browsing (whitelisted and blacklisted sites), app usage (blocked and allowed apps), calling a phone number, sending a SMS and setting a fake location for the phone. Whenever we find that data is sent to a domain, we study this domain to know if it belongs to a third party service or to the developers. We also save the IP address where the info is sent to analyze network traffic (see Subsection IV-B). After testing the 7 monitoring apps we found that all of them presented at least one of these problems:

Leakage of information before policy acceptance: Users must read and accept the privacy policy before the app can gather any type of information. Nevertheless we found one app that never shows the privacy policy to the user and two apps that gather information before the user has even created an account (which means implicit acceptance of the policy).

Leakage of information prior to permission grant: When an user gives an application permission to access certain information, the info collection should start at that moment. We found two applications that request access to the SMS and call logs and send a complete history, including data previous to the permission being granted. We believe that this is a problem in the Android permission API as well, because such an action should be blocked.

Sending of sensitive information to third parties: We consider that data that allows a party to learn sensitive information about the user (e.g, SMS and call logs, browsing history and location data) should not be shared with third parties. We found two applications that sent such data to third parties, one of them sends every URL visited by the child while the other one shares location data.

B. Insecure communications

We inspect the network traffic between each one of the apps and the controller servers, as well as any communication that might happen with third party services. To do so, we gather the TCP dumps of the phone's network communications and use the tool *Wireshark* to check if the communications are encrypted. We are able to filter the incumbent communications by using the IPs gathered in the previous subsection.

We found that 3 of the 7 monitoring apps send private information in the clear. Even more important is that two of those are actually sending every URL visited by the child in the clear, making it possible to infer the complete browser history. In the case of blocking apps, we don't expect them to send information over the Internet, because the parent manages the blocking from the child's phone, making the communication model different (and safer). One of the studied apps sends the IMEI code (unique identifier of a phone) unencrypted.

Table II shows a summary of the results found in the ten analyzed applications.

Table II
Number of apps where each of the privacy threats were found

Privacy threat	N^o of apps
Private data sent to 3rd party	2
Leakage of info prior to permission grant	2
Sending of private data before policy acceptance	3
Sending of private information without encryption	4

V. Ethical considerations and future work

We want to address that we have not harvested real data from users at any point of our experiments. Instead, we have created fake accounts with traffic generated by ourselves to avoid the gathering of data from real users. We also want to address that given the limitations spotted in this work, we are currently working on extending our method to be able to be more robust on finding dissemination of private data. We will do a responsible disclosure before publishing our final work.

VI. Conclusions

To the best of our knowledge, this is the first privacy study on parental control applications. These set of apps are highly intrusive making their study an important issue.

We analyzed 14 monitoring applications (7 of them partially) and 3 blocking apps and found that 7 out of 14 monitoring apps do not clearly explain their communication model, 17% of the requested permissions for 7 of the monitoring apps are not clearly necessary and that there are 11 total privacy issues between one of the blocking apps and the 7 monitoring apps.

References

[1] Felt, Adrienne Porter et al. "Android permissions demystified." ACM *Conference on Computer and Communications Security*, 2011.

[2] Felt, Adrienne Porter et al. "Android permissions: user attention, comprehension, and behavior." *Symposium on Usable Privacy and Security*, 2012.

[3] Backes, Michael et al. "On Demystifying the Android Application Framework: Re-Visiting Android Permission Specification Analysis." *USENIX Security Symposium*, 2016.

[4] Enck, William and Gilbert et al. "TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones." *USENIX OSDI*,2010

Second best student paper award: Towards Protecting Users' Sensitive Information in Context-Aware Solutions

Alberto Huertas Celdrán

Faculty of Computer Science, University of Murcia, 30100 Murcia, Spain
Email: alberto.huertas@um.es

***Abstract*—The evolution experienced by software and hardware technologies is increasing the vast amount of heterogeneous and sensitive information handled by information management systems. Users' location, their mobility, as well as the contextual information considered by context-aware services are also being affected due to the complexity when protecting the privacy of the users' information, which should be an essential requirement in information management systems. Addressing this challenge requires automatic mechanisms that allow users to control their information in real-time. To this end, we present several context-aware solutions that allow users to manage the privacy of their information in intra- and inter-context scenarios. The privacy policies managed by our solutions let users decide what, where, when, how, to whom, and at which level of precision they want to reveal their personal information.**

***Index Terms*—Context-awareness, multi-context, privacy policies, location-based services**

I. Introduction

The early days of Information Management Systems (IMS) were focused on managing and storing business information in companies and public organisms. The evolution experienced by computation paradigms and hardware technologies changed the focus and functionality of IMS by bringing them to the users' daily life. Nowadays, we use these systems with heterogeneous purposes such as social networks, on-line banking, or electronic commerce. This evolution has increased not only the amount of data to manage, but also the necessity of protecting some sensitive pieces of them. Otherwise, private pieces of information such as the users' locations, identities, activities, and contextual information could be obtained by malicious users or service providers and misuse it.

Controlling the privacy of sensitive information managed by IMS, and specifically by context-aware solutions, is an open challenge that needs more efforts to address it. An important number of context-aware solutions protect the users' information through static privacy policies defined at set-up time; however, they do not provide easy mechanisms to allow users to manage their privacy. Furthermore, static policies are not suitable for context-aware solutions when these users are changing of environment constantly. In this sense, context-aware solutions should:

- allow users to share their sensitive information with other users and services in a privacy-preserving way;
- control the access to the users' information in real-time and depending on their context and situation;
- enable the definition and management of context-aware policies to control what information users want to release, who can access them, and in which contexts and situations they want to disclose such information;
- represent in a formal way the information considered by contexts and privacy policies; and
- recommend suitable privacy policies according to the context or environment in which users are located.

In order to address the previous open challenges, during the Ph.D. Thesis we proposed several context-aware and privacy-preserving solutions that allow developing applications and services interested in protecting and preserving users' privacy. The proposed solutions and frameworks allow users to manage their information by privacy policies that consider the context or environment in which they are located. These policies allow users to share their information to the right users, at the right granularity, at the right place, and at the right time.

II. Context-aware solutions preserving privacy

Users' mobility has brought an evolution from proposals preserving users' information in certain contexts (intra-context solutions) to multiple and independent contexts (multi-context solutions). This section presents the contributions made during the Ph.D. Thesis in both environments.

A. Intra-context solutions

Among solutions protecting the information in intra-context scenarios, we proposed the framework *SeCoMan* (Semantic web-based Context Management) [1] that provided support for developing context-aware applications aimed at preserving users' privacy and their information. Using SeCoMan, users can define privacy policies to hide their locations to others; mask (cloacking) their position with fictitious ones; establish the granularity level at which they want to be located; and define the closeness level accepted to be located. Fig. 1 depicts the SeCoMan framework to offer these privacy features.

In SeCoMan, as well as in the following publications, we developed a set of ontologies to formally define the location and element items considered by contexts and privacy policies. Fig. 2 illustrates an example of ontology modeling the space, context information, and relationships among them.

PRECISE (Privacy-aware REcommender based on Context Information for cloud Service Environments) [2] is an evolution of SeCoMan that allows users to define privacy policies regarding services. It proposes a privacy-preserving and context-aware system by providing Location-Based Services (LBS) in the Mobile Cloud Computing (MCC) paradigm, this being located at its Platform as a Service (PaaS) layer.

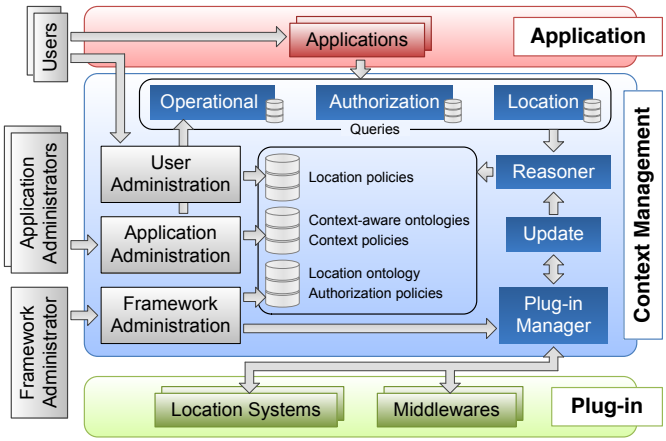


Fig. 1. SeCoMan framework to protect user's privacy

B. Multi-context solutions

These solutions consider intra- and inter-context scenarios when protecting users' information even though they move between independent contexts or environments. In that sense, *CAPRIS* (Context-Aware PRIvacy-preserving system Supervised by users) [3] was our first approximation to inter-context solutions. This is in charge of protecting users' information in the context where they are. Using CAPRIS, users are able to decide where, when, how, and to whom they want to reveal the *space* in which they are located, their *personal information*, *activity*, and other *contextual* information.

As evolution of CAPRIS, *MASTERY* (Multicontext-Aware System That prEserves the useRs privacY) [4] was presented to protect the privacy of users' information in multi-context scenarios, but incorporating the user consent to reveal his/her information to protect. To this end, MASTERY suggests to users several sets of privacy-preserving and context-aware policies, called *profiles*, and they just have to choose the most suitable profile according to their interests in the context where they are. Moreover, users who use MASTERY are able to modify the profiles by adding, deleting, or modifying some of the policies that form such profiles. Finally, the *h-MAS* (health-related Multicontext-Aware System) [5] solution is an important evolution of MASTERY oriented toward the management of the users' privacy in health scenarios, using policies that are both intra- and inter-context.

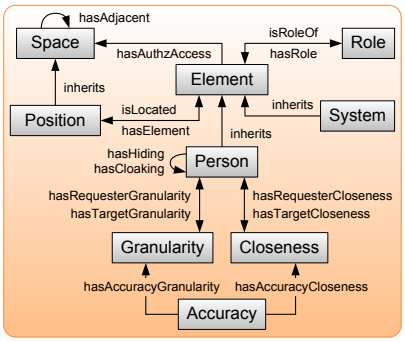


Fig. 2. Ontology to formally define location and elements to apply privacy-preserving policies

The policies making up the privacy-preserving and context-aware profiles of the previous solutions can be categorized into two different groups, intra- and inter-policies. The former preserve the users' information within specific contexts, while the latter provide such protection between different contexts. To protect the information, the intra- and inter-policies' groups are made up of *disclosure* policies, to indicate what information of the users can be shared, and *reveal* policies to indicate where, when, and how the information can be shared.

The main characteristics of the solutions presented above are illustrated in TABLE I.

TABLE I
PRIVACY-PRESERVING SYSTEMS MAKING UP THE PH.D. THESIS

	[1]	[2]	[3]	[4]	[5]
Intra-context privacy	√	√	√	√	√
Preserve users' location	√	√	√	√	√
Preserve users' identity		√	√	√	√
Preserve users' context			√	√	√
Policies defined by users	√	√	√	√	√
Policies set by the system			√	√	√
Inter-context privacy			√	√	√
Multi-context solutions				√	√
Health privacy-oriented					√

III. Conclusions

We have presented in this paper several solutions in charge of protecting the privacy of users' information within context-aware environments. The privacy-preserving policies provided by our systems consider the context in which users are located to manage their information to protect. Users of our solutions can modify policies at will by adding, deleting, and modifying information to control what, where, when, how, and to whom the users want to reveal their information. All the work herein presented is the result of the Ph.D. Thesis by Huertas Celdrán, under the supervision by the rest of co-authors of the paper.

Acknowledgment

This work has been supported by a Séneca Foundation grant within the Human Resources Researching Training Program 2014 (FEDER/ERDF) and a Séneca Foundation grant within the Human Resources Researching Postdoctoral Program 2017.

References

[1] A. Huertas Celdrán, F. J. García Clemente, M. Gil Pérez, G. Martínez Pérez: "SeCoMan: A Semantic-Aware Policy Framework for Developing Privacy-Preserving and Context-Aware Smart Applications," EEE Systems Journal, vol. 10, no. 3, pp. 1111–1124, September 2016.

[2] A. Huertas Celdrán, M. Gil Pérez, F. J. García Clemente, G. Martínez Pérez: "PRECISE: Privacy-Aware Recommender Based on Context Information for Cloud Service Environments," IEEE Communications Magazine, vol. 52, no. 8, pp. 90–96, August 2014.

[3] A. Huertas Celdrán, M. Gil Pérez, F. J. García Clemente, G. Martínez Pérez: "What Private Information are You Disclosing? A Privacy-Preserving System Supervised by Yourself," 6th International Symposium on Cyberspace Safety and Security, pp. 1221–1228, August 2014.

[4] A. Huertas Celdrán, M. Gil Pérez, F. J. García Clemente, G. Martínez Pérez: "MASTERY: A Multicontext-Aware System that Preserves the Users' Privacy," 2016 IEEE/IFIP Network Operations and Management Symposium, pp. 523–528, April 2016.

[5] A. Huertas Celdrán, M. Gil Pérez, F. J. García Clemente, G. Martínez Pérez: "Preserving Patients' Privacy in Health Scenarios Through a Multicontext-Aware System," Annals of Telecommunications, vol. 72, no. 9-10, pp. 577–587, October 2017.

Primer premio al mejor trabajo de estudiante: Detección de anomalías en redes industriales guiada por datos

Mikel Iturbe
Mondragon Unibertsitatea
Goiru 2, Arrasate-Mondragón
miturbe@mondragon.edu

***Resumen*—Este artículo resume el trabajo realizado en una tesis doctoral en el campo de la ciberseguridad industrial. Más concretamente, el trabajo realizado se ha centrado en la detección de anomalías guiada por datos en redes industriales. Las redes industriales –entornos interconectados por sistemas dedicados a automatizar, monitorizar y controlar procesos físicos– han evolucionado enormemente, mientras que los mecanismos de seguridad aplicables no han evolucionado al mismo paso, bien porque no escalan correctamente, bien porque no han tenido en cuenta las particularidades de este tipo de redes. Esta tesis doctoral se centra en desarrollar sistemas de detección de anomalías (SDAs) que utilizan los datos intrínsecamente creados en este tipo de redes (mediciones de campo, tráfico de red, registros...) para detectar eventos de seguridad.**

***Index Terms*—redes industriales, detección de anomalías, análisis de datos**

Tipo de contribución: *Premio al mejor trabajo de estudiante*

I. Motivación

Desde el desarrollo de los primeros Controladores Lógicos Programables (PLC) en la década de 1960, los sistemas de control industrial (SCI) han evolucionado considerablemente. Desde las primitivas instalaciones aisladas, los SCI están más conectados entre sí, hasta formar los entornos interconectados complejos conocidos como redes industriales (RIs) de hoy en día. Los SCI son responsables de un gran número de procesos físicos, desde plantas industriales de fabricación hasta la generación de energía, incluyendo procesos que pertenecen a infraestructuras críticas (ICs). Por ello, el correcto funcionamiento de las redes industriales es vital para preservar la actividad cotidiana de sociedades modernas, ya que gran parte del tejido ecónomico y del bienestar de éstas se sustentan en este tipo de redes.

En el campo de la ciberseguridad, los sistemas de detección de anomalías (SDAs) han tenido un rol referencial en la protección de este tipo de redes. Estos sistemas monitorizan el comportamiento de las RI y/o SCI para detectar eventos o comportamientos que se alejan del funcionamiento normal del entorno. Generalmente, estas propuestas se han basado en el aprendizaje automático o el análisis de datos para cumplir su función. Sin embargo, la creciente complejidad de las RIs ha derivado en los que los datos intrínsecamente creados en las RIs han crecido en volumen (más capacidad de almacenamiento de datos históricos), variedad (más variables monitorizadas) y velocidad (más lecturas) convirtiéndose así en un problema *Big Data*. No obstante, los SDAs diseñados para trabajar en RIs no han evolucionado igualmente, y las propuestas recientes no han sido diseñadas para abordar esta complejidad de los datos, ya que no escalan correctamente o no analizan la mayoría de los tipos de datos creados en RIs. Además, en una revisión de la literatura en el campo de los SDAs a gran escala se comprobó que éstos no eran adecuados para su uso en RIs [1].

Esta tesis doctoral aspira a llenar ese vacío mediante dos propuestas principales: (i) un sistema visual de monitorización de red y (ii) un SDA multivariante, capaz de trabajar a gran escala y con datos heterogéneos (a nivel de red y proceso).

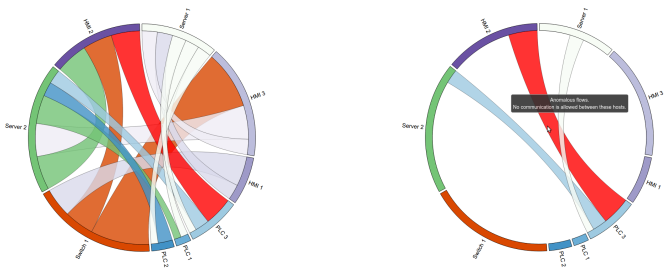
II. Contribuciones

Las contribuciones de esta tesis doctoral siguen un orden cronólogico: la construcción de un entorno de pruebas, la detección de anomalías a nivel de red y la detección de anomalías a nivel de red y de campo utilizando un modelo unificado escalable.

II-A. Banco de pruebas híbrido

En la investigación en ciberseguridad, es importante disponer de un entorno de pruebas en el que desarrollar y evaluar las diferentes propuestas. Los bancos de pruebas deben ser fieles al entorno que se está intentando replicar, y seguro, en el sentido que la realización de pruebas no afecte a activos fuera del banco de pruebas. Este último aspecto es aún más importante en el campo industrial, ya que la realización de pruebas puede comprometer la disponibilidad de los sistemas o tener un impacto físico no deseado en el entorno. Por ello, como paso preliminar a las principales contribuciones de la tesis doctoral, se ha construido un banco de pruebas en el que desarrollar y evaluar mecanismos de seguridad para redes industriales [2].

Este banco de pruebas es híbrido ya que utiliza diferentes técnicas de implementación: implementación en hardware, emulación, simulación y virtualización. Para ello se vale del software Emulab y del proceso simulado Tennessee-Eastman (TE). Emulab permite la emulación dinámica de diferentes topologías y escenarios de red, con la inclusión de nodos virtuales o físicos en diferentes configuraciones, mientras que el proceso TE, un proceso industrial de referencia, permite estudiar los efectos de diferentes situaciones en el plano físico de forma segura. La combinación de ambos, y la posibilidad de incluir diferentes tipos de procesos industriales hace que



(a) Flujo prohibido entre PLC 3 y HMI 2.

(b) Detalle del flujo anómalo

Figura 1: Diagrama de cuerdas de flujos de red con flujo de red anómalo.

el banco de pruebas desarrollado sea una plataforma versátil para la investigación en ciberseguridad para redes industriales.

II-B. Sistema visual de monitorización de seguridad de flujos de red industriales

Después de un análisis preliminar en el que se ha constatado la falta de visualizaciones de seguridad específicas para entornos industriales, se ha propuesto un sistema de monitorización visual de monitorización diseñado para facilitar la operación de la red y detectar anomalías de flujo [3]. Esta propuesta está basada en listas blancas, y diagramas de cuerdas. En una fase preliminar, el sistema construye las listas blancas en base al tráfico de red existente.

Una vez formadas las listas blancas, el sistema monitoriza tráfico real y lo visualiza en diagramas de cuerdas que muestran las diferentes relaciones entre los nodos de la red. En el caso de encontrar una conexión no contemplada en las listas blancas, ese flujo de red se muestra destacado sobre el resto, tal y como muestra la Figura 1. La escalabilidad de esta propuesta viene dada por la utilización de un servidor de búsqueda para el almacenamiento de las trazas de red y la generación de registros y de alertas, para los casos en los que la inspección visual no es viable. La propuesta ha sido validada con tráfico de una red industrial real.

II-C. Sistema de detección de anomalías multinivel

Las redes industriales se dividen en dos níveles lógicos principales. Por un lado, está el nivel de campo, compuesto por la realidad física que es monitorizada y los dispositivos (sensores y actuadores) que interactúan con ella. Por otro lado, está la capa de red, también conocida como la capa *cíber*, que engloba los dispositivos supervisores (servidores de control, interfaces humano-máquina...) de las redes indutriales. En términos generales, es el controlador industrial el que hace de puente entre ambos niveles, conectando la realidad física con los dispositivos supervisores.

La mayoría de las propuestas en la literatura solo se centran en una de las dos capas para detectar anomalías. En esta tesis, se ha desarrollado un marco en el que unificar ambos niveles en un único modelo para detectar anomalías. Para ello, utilizamos el Control Estadístico Multivariante de Procesos (CEMP, también conocido como MSPC, por sus siglas en inglés). CEMP ha sido utilizado previamente como método de detección de anomalías para redes IT [4]. Sus principales ventajas:

1. Basado en análisis de componentes principales, CEMP proporciona un modelo robusto de detección de anomalías y es capaz de definir en dicho modelo la información de un gran número de variables, teniendo en cuenta todas las variables existentes en una red industrial.
2. Mediante lo que se conoce como gráficos de contribución, CEMP permite analizar la contribución que han tenido las diferentes variables en la causa de una anomalía.

Utilizando CEMP, ya hemos demostrado que es posible obtener información acerca de la causa de una anomalía, pudiendo discernir en ciertas condiciones de si la causa de una anomalía se debe a una falla de proceso o la actuación de un atacante externo, solamente analizando las variables correspondientes al proceso industrial, es decir, magnitudes físicas [5]. Sin embargo, para obtener más información acerca de la causa del ataque e identificar los vectores de ataque utilizados, no basta con analizar la realidad física del proceso industrial; es indispensable añadir la información contenida en el nivel cíber o de red en el modelo CEMP para obtener esa información.

Para ello, se ha implementado un método basado en el conteo de ciertos eventos de red (registros, flujos activos...) en un lapso de tiempo dado, propuesto por [4] y se han añadido estas variables a las variables de proceso para la construcción del modelo CEMP. Para asegurar la escalabilidad de la propuesta, se ha desarrollado sobre Apache Spark.

La validación de este SDA multinivel se ha realizado utilizando el banco de pruebas descrito en la sección II-A, con tráfico de red real, el proceso TE y los registros creados por el SDA descrito en la sección II-B. Los resultados muestran que el SDA multinivel es capaz de detectar anomalías que afectan a uno o ambos niveles de la red industrial.

III. Transferencia tecnológica

El SDA visual se ha realizado en un proyecto de colaboración con MSIGrupo, transfiriendo el SDA visual a la gama de productos ofrecidos por la empresa.

Referencias

[1] M. Iturbe, I. Garitano, U. Zurutuza, and R. Uribeetxeberria, "Towards large-scale, heterogeneous, anomaly detection systems in industrial networks: A survey of current trends," *Security and Communication Networks*, vol. 2017, 2017.

[2] M. Iturbe, U. Izagirre, I. Garitano, I. Arenaza-Nuño, U. Zurutuza, and R. Uribeetxeberria, "Diseño de un banco de pruebas híbrido para la investigación de seguridad y resiliencia en redes industriales," in *Actas de II Jornadas Nacionales de Investigación en Ciberseguridad (JNIC 2016)*, 2016, pp. 3–10.

[3] M. Iturbe, I. Garitano, U. Zurutuza, and R. Uribeetxeberria, "Visualizing Network Flows and Related Anomalies in Industrial Networks using Chord Diagrams and Whitelisting," in *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 2, 2016, pp. 99–106.

[4] J. Camacho, A. Pérez Villegas, P. García Teodoro, and G. Maciá Fernández, "PCA-based multivariate statistical network monitoring for anomaly detection," *Computers & Security*, vol. 59, pp. 118–137, 2016.

[5] M. Iturbe, J. Camacho, I. Garitano, U. Zurutuza, and R. Uribeetxeberria, "On the feasibility of distinguishing between process disturbances and intrusions in process control systems using multivariate statistical process control," in *2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W)*, 2016, pp. 155–160.

An Analysis of Pre-installed Android Software

Julien Gamba[†◇], Mohammed Rashed[◇], Abbas Razaghpanah[‡], Narseo Vallina-Rodriguez[†∗], Juan Tapiador[◇]
† IMDEA Networks Institute, ◇ Universidad Carlos III de Madrid, ‡ Stony Brook University, ∗ ICSI

***Abstract*—Thanks to the openness of the Android ecosystem, mobile device vendors can build and sell smart phones and other mobile devices using their own custom versions of Android. Most of these custom versions deviate significantly from Google's official Android Open Source Project (AOSP): in addition to various visual and functional changes to the base OS, vendors add proprietary applications (apps hereafter) to their firmware, and sometimes even add custom (often unknown) certificates to the system's root certificate store. In fact, recent anecdotal evidence has revealed that pre-installed apps can put, intentionally or not, user's privacy and security at risk. This is especially concerning for lesser-known brands producing lower-end devices for whom preserving user privacy might not be high on the priority list. In this extended abstract, we present our methodology to explore the complex and diverse ecosystem of Android pre-installed apps as well as our preliminary results.**

***Index Terms*—Android; security; privacy; measurements**

Tipo de contribución: *Investigación en desarrollo*

I. Introduction

A modern Android phone comes with multiple apps and services pre-installed. The openness of the Android source code makes it possible for any manufacturer to ship a custom version of Android, along with some proprietary pre-installed apps on the system partition. These apps can be useful to users (*e.g.*, a browser or a calculator) but can also be unwanted. Moreover, pre-installed apps, specifically those installed in `/system/priv-app/` or signed with the platform signing key, may have access to privileged system permissions which are not available to user-installed apps. In many cases these apps run in the background, tracking user activities without their explicit consent, and often without their knowledge.

Some vendors have recently come under scrutiny by the media for these practices. For instance, it has been reported that OnePlus devices contain software that allows a remote controller to root the phone and perform other high-privilege operations that are reserved for the manufacturer [1]. Such modifications are typically introduced by manufacturers, but may also be done by network operators and phones resellers. So far, no research study has systematically studied the privacy and security risks of Android OS modifications beyond the addition of certificates in the trusted root store [2]. Consequently, pre-installed apps have remained a largely unknown area. It is unclear whether vendors use these apps only to harvest personal data from consumers, or provide APIs to affiliate apps and partners to access privileged resources, as in the case of the Samsung Knox API. While it is possible to avoid these potential abuses and vulnerabilities by installing more widely-trusted and open-source alternatives to the stock firmware (*e.g.*, LineageOS), it should be noted that it is far from an ideal solution as rooting devices exposes users to further security risks, it is neither easy nor accessible for most users to try, and that a third-party firmware can reduce the functionality of the device due to missing drivers and a slew of other issues.

This project, which is still in its early stages, seeks to shed light on the presence of pre-installed Android apps across vendors and devices, studying them in depth to answer the following questions:

- What is the ecosystem of pre-installed apps, including all actors in the supply chain?
- Do pre-installed apps disseminate or leak personally identifiable information (PII)? If so, with whom do they share this information and for what purpose?
- Do such apps present security vulnerabilities?
- What are the relationships between vendors and the potential app developers for their pre-installed apps?

To that end, we are currently gathering a large corpus of pre-installed apps by crowd-sourcing them from real user devices. Once this is done, we will apply both static and dynamic analysis techniques to these binaries. Finally, we aim to identify the origins of those apps, hoping to attribute their development to third party app developers using signature matching techniques. In this extended abstract, we will first detail our methodology (Section II) and then present some preliminary results (Section III).

II. Datasets and methodology

Most of the pre-installed apps cannot be found on traditional app stores; instead, they must be extracted from the system partition from real phones. We called for volunteers and extracted over 1,000 unique pre-installed APKs out of 15 devices, covering 8 different manufacturers–including both high-end and low-end vendors like Samsung and Wiko, respectively. None of these APKs is detected by the Androzoo project [3]. We obtained written consent from all users before we harvested anything from their phone, even though no personal data was obtained.

Our second dataset comes from the Lumen Privacy Monitor app [4]. Lumen is a home-built Android app, publicly available on Google Play, that aims to promote mobile transparency and enable user control over their personal data and traffic. Lumen leverages the Android VPN permission to intercept and analyze all Android traffic on user-space and in-situ, even if encrypted. For this study, we use over 15M anonymized traffic logs provided by over 13,000 users from over 120 countries. This dataset covers 567 pre-installed apps[1] found in 140 different Android vendors. We reference the reader to Lumen's previous work [4] to get a better understanding of its capabilities and its mechanisms to generate accurate traffic fingerprints on an per-application basis.

III. Preliminary results

A. Static Analysis

We applied basic static analysis techniques to the apps in our dataset to analyze permission usage, access to privileged resources and privacy leaks.

Personal Data Dissemination: We decompiled and manually investigated APKs from our dataset to look for personal data dissemination. We found some apps leaking the IMEI of

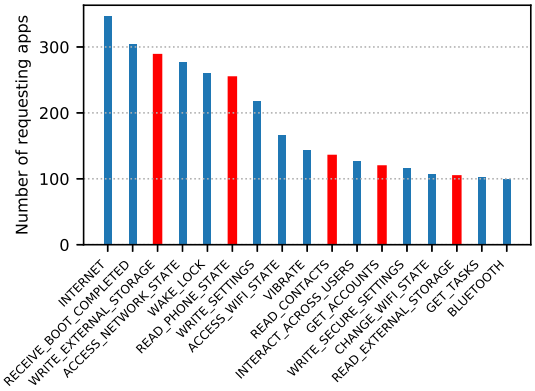


Figure 1: Most popular permissions among pre-installed apps. Red bars flag those permissions considered as "Dangerous" by Google [5].

Vendor	Custom Permissions
Samsung	188
Sony	138
Google	37
Meizu	15
HTC	13

Table I: Number of custom permissions by vendor

the phone through SMS along with other items of private information. We also found apps geo-targeting users from specific countries: first the pre-installed service checks the geographical location of the user and then, it leaks sensitive personal information if the user is located in a given country.

Android Permissions: The total number of permissions requested by pre-installed apps in our dataset is 1,064. Figure 1 shows the most requested permissions. The plot only shows the most popular permissions, *i.e.*, those requested by at least 100 apps. The red bars are the permissions flagged as "Dangerous" by Android [5] (*e.g.*, Location). As we can see, most pre-installed apps require Internet access, and a substantial amount of them can read the phone state, the list of contact and the accounts.

However, more than 99% of the permissions identified are custom permissions, defined by the app developer. Table I shows the number of custom permissions for some vendors in our dataset. These permissions are used for authentication mechanism, for billing or to control IoT devices. Samsung is the vendor with a largest number of custom permissions, including those associated with the Knox API. Google also offers additional permissions besides basic Android ones. For instance, `com.google.android.googleapps.permission.GOOGLE_AUTH` is used by Google services to authenticate the user with Google servers.

B. Lumen Analysis

Lumen detected personal data dissemination to cloud services emanating from 407 pre-installed apps, including 77 apps that where extracted from volunteers' phones. Note that not all of the apps from volunteers were put under Lumen's scrutiny: we expect to find more personal data dissemination by inspecting every app in our dataset.

[1]For this study, we consider a Lumen-analyzed app as pre-installed if it is not detected by the Androzoo project [3]. We are currently exploring more accurate methods to distinguish pre-installed software.

Domain	Percentage of PII leaks
data.flurry.com	1.24%
android.clients.google.com	1.18%
www.google.com	0.88%
pagead2.googlesyndication.com	0.87%
googleads.g.doubleclick.net	0.86%
settings.crashlytics.com	0.85%

Table II: Top third-party domains used by pre-installed apps

Traffic Analysis: We found 7,613 unique domains that receive data from pre-installed apps. Up to 79% of these communications are done over encrypted channels. Among these domains we found domains that are used to serve ads to the user (*e.g.*, Google's DoubleClick), and also other tracking and analytics services (*e.g.*, Crashlytics and Verizon's Flurry).

Personal Data Dissemination: We found that 63.2% of the flows containing personal data reach first-party domains, and the remaining third-party domains. Table II shows the most popular third-party domains among the pre-installed apps in our dataset. Further, 45.3% of these flows containing personal data are sent to third-party domains. We have observed that pre-installed services may upload sensitive data without encryption, hence constuting a serious privacy risk for users.

IV. Conclusion and Ongoing Work

To the best of our knowledge, our work is the first study of pre-installed apps and services in Android devices. The size and disparity of the Android ecosystem makes a thorough study challenging but crucial for users' security and right to privacy. Even with a limited size dataset, we managed to find numerous PII leaks and bad practices in pre-installed apps.

We are extending our dataset by scaling up our crowdsourcing campaign. Then, we'll leverage FlowDroid [6]–a static taint analysis tool for Android apps — to study the behaviour of our apps more deeply. We are also complement our analysis using the mobile cyber-intelligence dataset provided by Eleven Paths' Tacyt which contains millions of APKs [7] to identify relationships between pre-installed services and publicly available apps. Tacyt will allow us to cross-match, for instance, a given URL or custom permission across millions of apps. In addition to that, we are investigating methods to dynamically analyze pre-installed apps in virtualized environments, using Lumen to inspect closely their traffic.

References

[1] "OnePlus Secret Backdoor," https://www.theregister.co.uk/2017/11/14/oneplus_backdoor/, [Online; accessed 08-March-2018].
[2] N. Vallina-Rodriguez, J. Amann, C. Kreibich, N. Weaver, and V. Paxson, "A tangled mass: The android root certificate stores," in *Proc. ACM CoNEXT*, 2014.
[3] K. Allix, T. F. Bissyandé, J. Klein, and Y. Le Traon, "Androzoo: Collecting millions of android apps for the research community," in *Proceedings of the 13th International Conference on Mining Software Repositories*. ACM, 2016, pp. 468–471.
[4] A. Razaghpanah, N. Vallina-Rodriguez, S. Sundaresan, C. Kreibich, P. Gill, M. Allman, and V. Paxson, "Haystack: In situ mobile traffic analysis in user space," *arXiv preprint arXiv:1510.01419*, 2015.
[5] https://developer.android.com/guide/topics/permissions/overview.html, [Online; accessed 08-March-2018].
[6] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *ACM SIGPLAN Notices*, 2014.
[7] "Tacyt, Eleven Paths, Telefónica," https://www.elevenpaths.com/technology/tacyt/index.html, [Online; accessed 08-March-2018].

Gestión Dinámica de Seguridad en Dispositivos Móviles

P. García Teodoro, J. Camacho, G. Maciá-Fernández, J.A. Gómez Hernández,
M. Robles Carrillo, J.A. Holgado Terriza, A. Muñoz Ropa
Universidad de Granada - *Network Engineering & Security Group* - CITIC
{pgteodor, josecamacho, gmacia, jagomez, mrobles, jholgado, aropa}@ugr.es

***Resumen*—Los dispositivos móviles están pasando a ser las plataformas más adoptadas por los usuarios para el acceso a Internet. Esta amplia penetración lleva aparejados, no obstante, importantes problemas de seguridad. Tomando como punto de partida resultados científico-técnicos previos del grupo de investigación, el proyecto MDSM plantea el diseño y desarrollo de un sistema de gestión dinámica de seguridad para dispositivos móviles. Concebido para ser operativo en un entorno de red real, dos son los módulos funcionales que definen la propuesta: un sub-sistema de control de acceso y un módulo de monitorización y supervisión continuada en el tiempo. Ambas funcionalidades se sustentan en el análisis multivariante para concluir desviaciones sospechosas respecto del comportamiento esperado. Además de los beneficios esperados en detección, el acceso a información llevado a cabo respetará en todo momento la privacidad del usuario.**

***Index Terms*—Análisis multivariante, control de acceso, detección de anomalías, dispositivo móvil, monitorización**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

La creciente penetración y dependencia social de Internet en los últimos años ha venido de la mano del también creciente uso de dispositivos móviles de usuario, tales como teléfonos inteligentes (*smartphones*) y tabletas. Diversos estudios evidencian que la adquisición de este tipo de dispositivos ha superado ya la de equipos tradicionales como los de sobremesa e incluso la de portátiles [1]. En consonancia con ello, el tráfico móvil a nivel mundial en Internet está en constante aumento, superando ya en este momento el 50 % del total según diferentes estudios (y llegándose a situar en algunos casos en torno al 70 %) [2]. Paralelamente a ello, cabe destacar los elevados riesgos de seguridad asociados a los dispositivos móviles no solo para usuarios finales sino también para empresas y organizaciones de todo tipo [3], [4].

En este contexto, son numerosos los esfuerzos realizados por la comunidad científica y la industria con el fin de mejorar la seguridad de equipos y sistemas en entornos de movilidad. Por una parte, cabe mencionar el desarrollo de soluciones conocidas como MDM (*Mobile Device Management*), cuyo principal objetivo es la monitorización y gestión remotas de un móvil, sin importar el operador o proveedor de servicios del mismo. Se posibilita así a la corporación acceder a la configuración del dispositivo personal, haciendo de forma efectiva que esté bajo su control (*p.ej.*, [5], [6], [7]). Al margen de este tipo de tecnologías, la provisión de seguridad en entornos móviles resulta per se un ámbito de actuación prolífico en lo que se refiere a la propuesta de soluciones y su impacto. Dos tipos básicos de aproximaciones destacan aquí: aquellas en las que el proceso de monitorización y detección se realiza en el propio dispositivo final (intra-dispositivo) [8], y aquellas en las que dicho proceso tiene lugar fuera del dispositivo para reducir costes y/o consumo (extra-dispositivo) [9], [10]. Ambas aproximaciones, intra- y extra-dispositivo, resultan complementarias. Esta complementariedad hay que entenderla, además, en el contexto general de la tendencia en la industria de la detección de intrusiones, donde se persigue la agregación de diversas fuentes de información a través de los conocidos SIEM (*Security Information & Event Management*) [11], sistemas que permiten a los equipos CSIRT (*Cyber/Computer Security Incident Response Team*) identificar, comprender y priorizar los eventos de seguridad ocurridos.

II. MDSM: Gestión dinámica de seguridad en dispositivos móviles

En el marco global antes descrito y tomando como base desarrollos previos del grupo NESG (*Network Engineering & Security Group* - https://nesg.ugr.es) [12], [13], [14], los autores acaban de poner en marcha el proyecto de referencia TIN2017-83494-R, cuyo objetivo general es el *"diseño, despliegue y evaluación de una herramienta de control de acceso y gestión de seguridad remotos, para garantizar la operación segura de un dispositivo móvil en una infraestructura de red dada, que mantenga la privacidad del usuario y que implique un coste de recursos reducido"*.

Dicha herramienta se ha venido a denominar MDSM (del inglés *Mobile device Dynamic Security Management*) y su funcionalidad se esquematiza en la Figura 1 en base a los siguiente módulos:

- *MDSM-m*: entidad MDSM dispuesta en el dispositivo móvil para su análisis de cara a su acceso (o no) a la infraestructura de red pretendida y su monitorización.
- *MDSM-r*: entidad MDSM desplegada en la red receptora y a través de la cual se gestionará el acceso de los dispositivos y su monitorización. MDSM-r podrá ser centralizado o distribuido en la red a fin de proporcionar escalabilidad a la solución final.

Cada una de estas dos entidades estará conformada por sendos módulos ideados para ofrecer la funcionalidad de seguridad deseada:

- *MDSM/CA*: control de acceso seguro.
- *MDSM/MS*: monitorización y supervisión de seguridad.

Por lo que respecta al **control de acceso** (módulos MDSM/CA), este debe suponer un nivel adicional a los ya existentes en el entorno (típicamente consistentes en un proceso de autenticación soportado en una clave conocida por

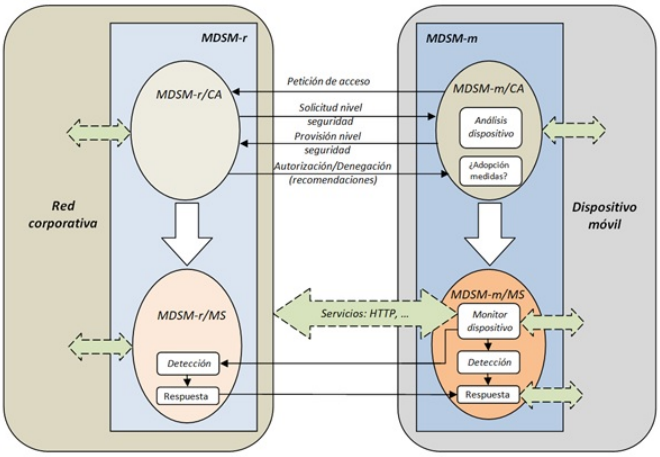


Figura 1. Esquema funcional del sistema MDSM.

las partes) y se sustentará en un valor de confianza relacionado con el nivel de seguridad del dispositivo. Dicho valor se obtendrá localmente a través de MDSM-m/CA a partir de cuestiones relacionadas con su estado: aplicaciones disponibles, permisos usados, actualizaciones instaladas, historial de visitas web, estado de la configuración del sistema, etc.

El resultado de la valoración de seguridad debe mantener la privacidad del usuario. Así, aunque este acceda a toda la información de seguridad en su dispositivo, dicha información será transformada antes de ser enviada a MDSM-r/CA. Esta transformación, desarrollada en el proyecto VERITAS (https://nesg.ugr.es/veritas), se basa en la compresión de la información de seguridad en dos valores denominados D-st y Q-st, ampliamente usados en la monitorización estadística de control de procesos o MSPC (por sus siglas en inglés) y recientemente extendidos a las redes en [13]. Esto implica dos importantes beneficios: *(i)* mantenimiento de la privacidad de la fuente, ya que no se utiliza la información interna al dispositivo sino su transformación; y *(ii)* reducción del volumen de datos a enviar al sistema central, permitiendo obtener una solución escalable y una reducción del consumo energético.

Como complemento al módulo MDSM/CA, y en coherencia con el objetivo global perseguido, una vez aceptado el acceso de un dispositivo a la red se procederá a la **monitorización** de este a lo largo del tiempo (módulo MDSM/MS) para determinar de forma dinámica la posible ocurrencia de eventos potencialmente maliciosos y/o perjudiciales para el conjunto. Esta funcionalidad se basará en el procedimiento general que sigue:

- Captura de la operación del dispositivo (MDSM-m/MS) en base a información multi-fuente tal como el estado de las interfaces, el uso de recursos locales, el acceso a recursos organizativos, las comunicaciones realizadas, etc.
- Empleo de un proceso de detección de eventos maliciosos basado en comportamientos anómalos, a dos niveles:
  - Intra-dispositivo (MDSM-m/MS), ligero para su ejecución en dispositivos de bajas prestaciones [12].
  - Extra-dispositivo (MDSM-r/MS), de forma que se permita seguir el comportamiento de los dispositivos en la red para aprender comportamientos multi-usuario y derivar eventos de alerta de significado global. En todo ello se adoptarán esquemas desarrollados en trabajos previos [13], [14].

El proceso de detección multi-nivel resultará en dos actuaciones subsiguientes: notificación al usuario de la situación de seguridad de su dispositivo y adopción efectiva de medidas que permitan resolver el evento de seguridad reportado.

III. Conclusiones

Amén de las contribuciones científico-técnicas esperadas de los distintos desarrollos referidos, es de destacar la novedad en sí del sistema global propuesto como metodología de securización de entornos móviles; tópico este que constituye un reto de primera magnitud en la actualidad y al que hay que atender de forma adecuada si deseamos avanzar en el despliegue y adopción generalizados de estas tecnologías y los servicios relacionados. Así mismo, es preciso mencionar también que en todo el desarrollo se abordarán también tareas relacionadas con el cumplimiento de legislación y normativa.

Agradecimientos

Este trabajo ha sido financiado parcialmente por el Gobierno de España, con fondos FEDER, a través del proyecto TIN2017-83494-R.

Referencias

[1] Statistica: “Shipment Forecast of Laptops, Desktop PCs and Tablets Worldwide from 2010 to 2020”. *Tech. report*, 2017. https://www.statista.com/statistics/272595/global-shipments-forecast-for-tablets-laptops-and-desktop-pcs.

[2] Ericsson: “Ericsson Mobility Report. On the Pulse of the Network Society”. *Tech. report*, 2016. https://www.ericsson.com/res/docs/2016/ericsson-mobility-report-2016.pdf

[3] Symantec: “ISTR: Internet Security Threat Report. April 2016”. *Tech. report*, 2016. https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf

[4] McAfee: “Trojans, Ghosts, and More Mean Bumps Ahead for Mobile and Connected Things”. *Tech. report*, 2017. https://www.mcafee.com/us/resources/reports/rp-mobile-threat-report-2017.pdf

[5] IBM: “IBM MaaS360”. https://www.ibm.com/es-es/marketplace/mobile-device-management

[6] VMware: “VMware AirWatch”. http://www.air-watch.com

[7] Cisco: “Cisco Meraki”. ‘https://meraki.cisco.com/products/systems-manager

[8] G. Suárez, J.E. Tapiador, P. Peris, A. Ribagorda: “Evolution, Detection and Analysis of Malware for Smart Devices”. *IEEE Communications Surveys & Tutorials*, vol. 16, n. 2, pp. 961-987, 2014.

[9] K. Tam, S.J. Khan, A. Fattori, L. Cavallaro: “CopperDroid: Automatic Reconstruction of Android Malware Behaviors”. *22th Annual Network and Distributed System Security Symposium (NDSS)*, pp. 1-15, 2015.

[10] S. Jadhav, S. Dutia, K. Calangutkar, O. Tae, H.K. Young, N.K. Joeng: “Cloud-based Android Botnet Malware Detection System”. *17th International Conference on Advanced Communication Technology (ICACT)*, pp. 347-352, 2015.

[11] O. Rochford, K.M. Kavanagh, T. Bussa: “Critical Capabilities for Security Information and Event Management”. *Gartner*, 2016. https://www.gartner.com/doc/3406918/critical-capabilities-security-information-event

[12] A. Ruiz-Heras, P. García-Teodoro, L. Sánchez-Casado: “ADroid: Anomaly-based Detection of Malicious Events in Android Platforms”. *International Journal of Information Security*, vol. 16, n. 4, pp. 371-384, 2017.

[13] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, G. Maciá-Fernández: “PCA-based Multivariate Statistical Network Monitoring for Anomaly Detection”. *Computers & Security*, vol. 59, pp. 118-137, 2016.

[14] G. Maciá-Fernández, J. Camacho, P. García-Teodoro, R.A. Rodríguez-Gómez: “Hierarchical PCA-Based Multivariate Statistical Network Monitoring for Anomaly Detection”. *8th IEEE International Workshop on Information Forensics and Security (WIFS)*, pp. 1-6, 2016.

DroidSentinel: ¿Está mi dispositivo móvil participando en un ataque DDoS?

A. Herranz González, B. Lorenzo Fernández, D. Maestre Vidal, G. Rius García,
M. A. Sotelo Monge, J. Maestre Vidal y L.J. García Villalba

Grupo de Análisis, Seguridad y Sistemas (GASS, http://gass.ucm.es)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, España
E-mail: {andrhe01, borjalor, diemae01, grius, masotelo, jmaestre}ucm.es, javiergv@fdi.ucm.es

***Resumen*—Este artículo introduce la estrategia DroidSentinel para la detección de la participación de dispositivos móviles en ataques de Denegación de Servicio basándose en el análisis de flujos de tráfico saliente. A diferencia de gran parte de las aproximaciones en la bibliografía, la solución propuesta asume la no estacionalidad del entorno de monitorización, considerando únicamente la información reportada por cada extremo origen. Los datos recolectados por los dispositivos son transferidos a un servidor de análisis remoto, el cual provee capacidades avanzadas de agregación de métricas, clasificación, predicción y construcción de umbrales de decisión. De esta manera se reduce el impacto inherente a la ejecución de tareas analíticas completas causado en el sistema protegido. Los resultados preliminares observados al considerar diferentes métricas y modos de uso del dispositivo han sido prometedores, demostrando la eficacia de la propuesta en los diferentes escenarios de evaluación implementados.**

***Index Terms*—Android, denegación de servicio, dispositivos móviles, sistema de detección de intrusiones**

Tipo de contribución: *Investigación original*

I. Introducción

El importante aumento de los ataques de Denegación de Servicio Distribuido o DDoS (del inglés *Distributed Denial of Service*) observado en la última década ha puesto sobre aviso a las principales organizaciones para la ciberdefensa [1], siendo actualmente una amenaza bien conocida por la sociedad de la información. Un ejemplo claro de este problema pudo observarse en octubre del año 2016, cuando los servidores DNS del proveedor Dyn registraron uno de los más complejos y mediáticos ataques DDoS [2]. Entre sus consecuencias destacó la inutilización de decenas de servicios, páginas web y redes sociales como Twitter, Reddit, Github, Amazon o Spotify. Esto fue posible gracias a la explotación de una vulnerabilidad presente en millones de dispositivos de distinta naturaleza conectados a Internet, los cuales forman parte del denominado Internet de las Cosas o IoT (del inglés *Internet of Things*) [3]. La amenaza fue orquestada desde una red de equipos zombis (del inglés *botnet*) gestionada por el *malware* Mirai [4], [5]. Dada la popularidad de los servicios afectados, el ataque sirvió para agravar la incertidumbre de muchos de sus usuarios, los cuales, a raíz de este suceso o ataques similares han llegado a preguntarse: ¿Están mis dispositivos participando en un ataque coordinado?, y de ser así ¿cuál es su naturaleza?, ¿en qué medida lo están haciendo? o ¿cómo puedo evitarlo? A pesar de la importancia de luchar contra estas amenazas en su origen (del inglés *source-based detection*), desde el punto de vista de las redes de comunicaciones, este problema apenas ha sido analizado por la comunidad investigadora [6], [8], cuyos esfuerzos se han centrado en el análisis de tráfico en los extremos intermedios y finales de la intrusión, o en la identificación de infecciones por *malware* de control remoto [9]. Afortunadamente, a raíz de la emergencia de las nuevas tecnologías de red (redes definidas por *software* o SDN (del inglés *Software-Defined Networking*), virtualización de funciones de red o NFV (del inglés *Network Function Virtualization*), etc.) y los avances hacia consolidar las redes de quinta generación 5G [10], la detección de los ataques DDoS en su origen vuelve a jugar un papel esencial, facilitando la definición de defensas basadas en redes autoorganizadas o SON (del inglés *Self-Organizing Networks*) [7]. Esta necesidad contrasta con un estado del arte lleno de propuestas obsoletas [8], que habitualmente no asumieron los escenarios de red venideros.

Con el fin de contribuir al desarrollo de soluciones capaces de lidiar con estos problemas, este artículo introduce la estrategia DroidSentinel. Se trata de una aproximación inicialmente desarrollada para operar en dispositivos Android, que tiene como objetivo principal el análisis de flujos de tráfico de salida en busca de indicios de actividades maliciosas, en particular, aquellas que desenmascaren la participación de un dispositivo concreto en ataques DDoS. DroidSentinel se centra en la identificación de comportamientos discordantes deducidos de la ocurrencia de eventos inesperados en entornos de monitorización con características no estacionarias. Esto requiere de la estimación del comportamiento del tráfico monitorizado, la elaboración de umbrales que permitan decidir su naturaleza, y un continuo proceso de recalibrado que facilite su adaptación a cambios en la distribución de los datos a analizar. La primera versión de DroidSentinel fue desarrollada en el Hackaton llevado a cabo durante el evento Cybercamp 2017 organizado por INCIBE, cuyo código fuente está disponible en [11]. Actualmente se enmarca en el proyecto SELFNET (H2020-ICT-2014-2/671672) e implementa parte de sus capacidades de monitorización, agregación y análisis con el fin de llegar a ser una solución de seguridad práctica a los problemas previamente planteados, teniendo en el punto de mira los escenarios de red venideros. Las principales contribuciones de esta investigación son la revisión

en profundidad del problema de los ataques DDoS y las diferentes propuestas para su mitigación desde el punto de vista académico, la introducción de una estrategia para su identificación adaptada a las características del tráfico saliente de dispositivos móviles, su adaptación a una arquitectura avanzada para la autoorganización de esquemas defensivos, el desarrollo de un banco de pruebas que implemente la metodología de evaluación considerada y la discusión de los resultados obtenidos en la experimentación realizada.

El resto del artículo ha sido organizado de la siguiente manera: en la sección II se describen las características de los ataques DDoS y las diferentes aproximaciones para su mitigación. En la sección III se introducen los principios de diseño de DroidSentinel. En la sección IV se detalla su estrategia de detección. En la sección V se describe la metodología considerada para su evaluación. En la sección VI se discuten los resultados objetivos. Finalmente, en la sección VII se exponen las conclusiones y propuestas de trabajo futuro.

II. TRABAJOS RELACIONADOS

Aunque existen diferentes estrategias capaces de denegar el servicio de un elemento de red, este artículo se centra en aquellas basadas en inundación [6], las cuales han sido tipificadas como ataques DDoS de tasa alta (del inglés *high-rate*) y ataques DDoS de tasa baja (del inglés *low-rate*) [13]. La primera familia de amenazas se basa en la inyección puntual de una gran volumen de tráfico/peticiones, mientras que los ataques DDoS de tasa baja tratan de pasar desapercibidos por medio de la adopción de patrones de emisión incrementales o de activación/desactivación, los cuales son menos visibles por las técnicas de detección convencionales [14], [15]. Además, el atacante a menudo se vale de estrategias de reflexión [16] y/o amplificación [17] para agravar el impacto de la intrusión. Un ejemplo claro de ello se observa en los ataques DDoS basados en inundación de enlaces o LFA (del inglés *Link flooding Attacks*) [18], donde los flujos de peticiones de tasa baja procedentes de regiones con alta densidad de tráfico son reaprovechados con el fin desbordar la capacidad de cómputo de los elementos intermedios de red, de este modo dificultando su identificación. La denegación de servicio basada en inundación originalmente era lograda partiendo de un único punto de ataque, siendo comúnmente conocida como DoS (del inglés *Denial of Service*). Pero la mejora de la capacidad de cómputo de los dispositivos actuales, los avances en la computación en la nube (del inglés *cloud computing*) y la tendencia a la adopción de mecanismos de auto-escalado y/o balanceo de carga, han llevado a que el atacante requiera de una gran cantidad de elementos comprometidos capaces de inyectar tráfico, por lo que habitualmente se recurre al uso de *botnets* [19] para su ejecución. Éstas cada vez son más extensas y se adaptan mejor a los nuevos escenarios de red [5]. Además, han evolucionado hacia una mayor robustez y capacidad de de evasión de técnicas de mitigación [20], lo que supone una peligrosa lanzadera de intentos de denegación de servicio. No obstante, dada la complejidad y la extensa bibliografía relacionada con este problema, no se profundizará en ellas, revisándose en [21] algunos de sus aspectos más relevantes.

La mayor parte de los esfuerzos de la comunidad investigadora asumen las circunstancias previamente mencionadas. Pero dada la complejidad del problema a tratar, la solución es frecuentemente dividida en cuatro desafíos diferentes [20]: prevención, detección, mitigación e identificación del origen. La prevención de DDoS se centra en evitar que el ataque llegue a la víctima, abarcando medidas que implican desde políticas de filtrado hasta la redistribución del tráfico. Nótese que a diferencia que en su mitigación, no se requiere de la identificación previa de la intrusión, teniéndose en cuenta rasgos unívocos del tráfico legítimo [22], pruebas de Turing [23], protocolos de seguridad [24] o sistemas basados en reputación [25]. Las investigaciones centradas en la detección de DDoS requieren de la observación de la intrusión, ya sea mediante el análisis del tráfico involucrado y/o el estudio de eventos a nivel de red [6]. Con este fin se han adoptado diferentes técnicas de análisis, como el modelo oculto de Márkov [28], redes neuronales artificiales [29], entropía [14], máquinas de vector soporte [30] o árboles de decisión, discutiéndose en éste último la eficacia de diferentes técnicas de aprendizaje automático. Tal y como se observa en [8], dentro de este grupo, la detección en el extremo origen desempeña un papel minoritario. Sus principales líneas de investigación se han centrado en la validación de los destinatarios del tráfico saliente [32] y en el reconocimiento de patrones de tráfico anómalos [6]. Por ejemplo, en D-WARD [33] se propone la construcción de modelos del uso normal del tráfico que fluye a través del sistema protegido en base a métricas extraídas a nivel de flujo, a partir de los cuales es posible distinguir comportamientos discordantes. Otra aproximación clásica se ilustra en [34], donde se estudia la proporcionalidad entre el tráfico saliente y el entrante. Con la llegada de las tecnologías SDN se ha vuelto a profundizar en este paradigma de detección [8], donde a menudo son objeto de análisis las tablas de flujo del protocolo OpenFlow. Esto permite detectar ataques DDoS originados en grupos de sistemas comprometidos [35], los cuales pueden ser dispositivos móviles [36]. No obstante, requieren de la monitorización y el estudio de información obtenida en al menos pequeñas/medianas regiones de red. Una vez reconocida la intrusión, actúan las estrategias de mitigación. Estas pueden implicar el despliegue reactivo/proactivo de algunas de las técnicas de prevención anteriormente citadas, o incluso reforzar la capacidad de detección de regiones colindantes y establecer regiones de cuarentena [7]. También pueden asumir el despliegue de otras contramedidas relacionadas con el modelo de seguridad activa, como la instanciación de tarros de miel y señuelos [26], o la redirección del tráfico malicioso hacia servidores sumidero [27]. Dentro de este paradigma se enmarcan las principales técnicas de identificación del origen del ataque. Su objetivo inicial es localizar al intruso, para lo cual es frecuente la adopción de técnicas de marcado de paquetes, discutiéndose en [37] la eficacia de muchas de ellas. Dado que éstas dependen directamente de las características de la red [38] y afrontan las restricciones impuestas por las diferentes políticas/éticas para la privacidad y protección de datos en las regiones que atraviesa el ataque [39], su objetivo frecuentemente se torna en tratar de llegar lo más cerca del intruso, lo que permite extender el radio de acción de las contramedidas a aplicar.

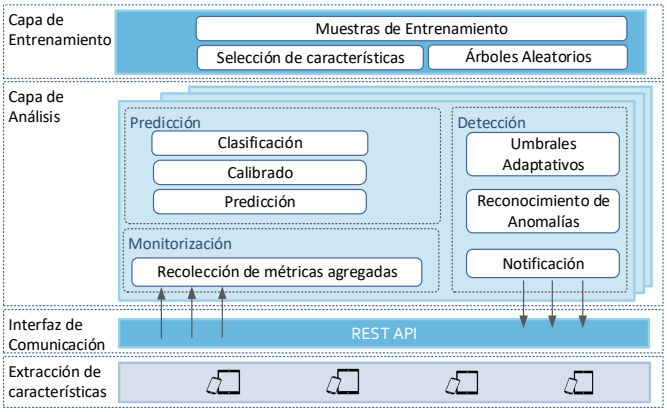


Figura 1: Arquitectura DroidSentinel.

III. DROIDSENTINEL

III-A. Principios de diseño

Tal y como se ha descrito en la Sección 2, la defensa frente a los ataques DDoS puede abordarse desde diferentes perspectivas, que abarcan desde la prevención hasta la identificación del origen de las amenazas [20]. Además, dada la complejidad de los escenarios emergentes de red, pueden plantear una gran cantidad de desafíos, como la decisión del lugar de actuación de las medidas defensivas [6], la naturaleza de la información a modelar [14] o la implementación de políticas de gestión de seguridad [39]. Con el fin de facilitar la comprensión del trabajo realizado, se ha asumido como objetivo principal el desarrollo de una estrategia de detección de ataques DDoS en el extremo origen adaptable a procesos no estacionarios en la información a analizar. A diferencia de propuestas similares, sólo se considera una única fuente de información, que es el dispositivo protegido [35]. Los objetivos secundarios más relevantes son su implementación como solución para sistemas Android, y su integración como sensor en un esquema de autoorganización de redes de nueva generación, lo que permite aprovechar sus capacidades analíticas y contribuir a la definición de casos de uso de mayor sofisticación. Para este fin se ha seleccionado el proyecto SELFNET (H2020-ICT-2014-2/671672) [12], el cual se alinea con los grupos de trabajo que avanzan hacia el desarrollo de redes 5G. Con el fin de delimitar y asentar las bases de la investigación realizada, se han asumido las siguientes premisas:

- La detección de la participación de un dispositivo Android en un ataque DoS en base al estudio de su tráfico saliente es posible. Esta es la hipótesis alternativa de la investigación, siendo su opuesto la hipótesis nula.
- Los ataques DoS basados en inundación dirigidos desde dispositivos Android principalmente se distinguen de la actividad normal en sus distribuciones de número de peticiones y volumen observado en los flujos de tráfico inyectados [40] . En los ataques DDoS además varía el número de clientes involucrados [41].
- El estudio basado en el análisis de discordancia de la entropía en métricas agregadas a nivel de flujo permite el reconocimiento de actividades DDoS en escenarios convencionales [42].
- La extracción de métricas avanzadas y su análisis en un servidor dedicado reduce considerablemente su impacto en el sistema protegido. Esta es una práctica habitual en dispositivos móviles [43], capaz de reducir entre otros su consumo energético.
- Se asume la no estacionalidad de la información inferida a partir de flujos de tráfico saliente de los dispositivos móviles protegidos, ya que ésta depende en su mayor parte de los hábitos del usuario.

El ámbito de la investigación realizada ha sido delimitado por las siguientes restricciones:

- No se ha tenido en cuenta la protección de los canales de comunicación frente a ataques hacia la integridad, disponibilidad y confidencialidad de la información [44].
- A pesar de que SELFNET ofrece capacidad avanzadas de correlación de incidencias y actuación, su aprovechamiento queda fuera del alcance de esta contribución.
- Aunque en la actualidad existen diferentes estrategias para la evasión de métodos de detección similares a los implementados, no se ha profundizado en los mecanismos adoptados para su prevención [42].
- No se han considerado el problema de la protección de información sensible inherente a las actividades de red compartidas por los usuarios.
- No se profundiza en la representación del conocimiento ni en los modelos de datos implementados para la gestión y almacenamiento de la información recolectada.

III-B. Arquitectura

En la Fig. 1 se muestran los componentes de la arquitectura DroidSentinel, cuya estructura de capas adopta los principios de SELFNET. Los dispositivos de usuario llevan a cabo la extracción de métricas agregadas que son enviadas a través de una interfaz de alto nivel a la Capa de Análisis. Esta capa lleva a cabo el proceso de reconocimiento de posibles amenazas DDoS descrito en la Sección IV, compuesto por las fases de: Monitorización, Predicción y Detección. Debido a que la Capa de Análisis centraliza la labor de detección, su despliegue debe ser escalable a múltiples instancias. Por otra parte, la Capa de Entrenamiento actúa como módulo auxiliar para la generación del modelo de clasificación, que es utilizado en la selección del algoritmo predictivo. Finalmente, los resultados de la detección son notificados a los dispositivos como respuesta a los envíos de métricas monitorizadas, completándose así el ciclo de detección.

III-C. Métricas

A lo largo de la investigación realizada se han considerado diferentes niveles de procesamiento de información, lo que conlleva la necesidad de extraer características muy diferentes que faciliten el análisis del conocimiento adquirido. Para la automatización de la decisión de la estrategia de modelado y/o predicción de las series temporales analizadas se ha considerado la batería de 100 métricas proporcionada por la herramienta TSFRESH, la cual ha sido desarrollada en el marco del proyecto alemán iPRODICT [45]. Esta colección tiene en cuenta desde atributos estadísticos básicos (picos, valores máximos, mínimos, etc.) hasta medidas de correlación y evolución de la serie temporal (ruido blanco, tendencia, estacionalidad, autocorrelación, etc.), habiéndose aplicado directamente sobre la colección M3-Competition [46] en la etapa

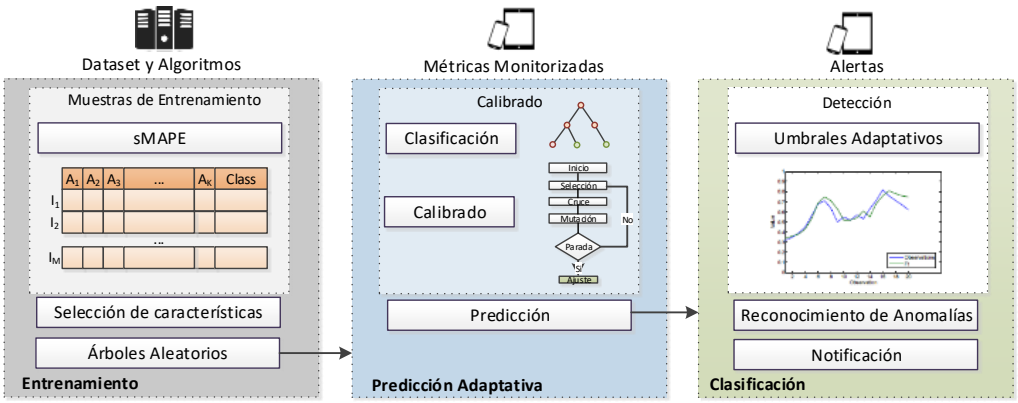


Figura 2: Análisis de información en DroidSentinel

de entrenamiento del sistema. Los flujos de tráfico saliente del dispositivo protegido son monitorizados y estructurados en formato IPFIX [47]. De cada flujo monitorizado se calculan el número de paquetes N, total de *bytes* transmitidos S y el número de destinos D. Las dos primeras métricas son agregadas en función de la relación entre tráfico saliente y entrante, y su nivel de desorden [34]. En el primer caso se considera el error cuadrático medio normalizado E, expresado de la siguiente manera:

$$E = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sigma^2} \quad (1)$$

donde n es el número total de flujos registrados en la observación, x_i es el valor de la métrica obtenida del tráfico del flujo i, y $\hat{x}_i$ es el valor de la misma métrica calculada sobre tráfico en sentido opuesto. Por ejemplo $E_\tau(N_{int}, N_{out})$ es la diferencia entre el número de paquetes entrantes $x = N_{in}$ y salientes $x = \hat{N}_{out}$ monitorizados a lo largo de un periodo de observación.

Por otro lado, el nivel de desorden de las observaciones es calculado en base a la entropía normalizada de Shannon, decisión justificada en trabajos como [14], donde su eficacia frente a otras métricas agregadas ha sido demostrada en escenarios DDoS convencionales. Al igual que en la bibliografía, la entropía se calcula a partir de la siguiente expresión:

$$H(X) = \frac{-\sum_{i=1}^{n} p_i \log_a p_i}{\log_a n} \quad (2)$$

donde n es el total de flujos monitorizados durante la observación, y $p_1, p_2, \cdots, p_n$ son las probabilidades de las instancias $x_1, x_2, \cdots, x_n$ de la variable aleatoria X, la cual representa una de las métricas de flujo del conjunto $X \in \{N, S\}$. Por ejemplo, $H(X)_N$ define la entropía del número total de paquetes registrados en cada flujo de tráfico monitorizado durante una observación. Nótese que cuando $H(X) = 0$ la variable es determinista. Su opuesto es $H(X) = 1$, e implica que el nivel de desorden es máximo.

IV. DETECCIÓN DE DDOS EN SU ORIGEN

DroidSentinel basa su estrategia de detección en el estudio de series temporales univariantes construidas a partir de métricas agregadas, las cuales son deducidas tanto de tráfico saliente monitorizado en el dispositivo protegido, como de colecciones de series temporales de referencia (en la experimentación, el conjunto M3-Competition [46]). Con este fin se distinguen tres grandes etapas de procesamiento de datos: entrenamiento, predicción adaptativa y clasificación (ver Fig. 2). En la primera de ellas se definen los criterios que permiten decidir el modelo predictivo que mejor se adapte a los datos a analizar a partir de las muestras de referencia. En la etapa de predicción adaptativa se calibran las estrategias de modelado para mejorar los pronósticos a realizar. Por lo tanto, se tienen en cuenta las características de las series temporales a analizar, las cuales han sido previamente monitorizadas en la salida del dispositivo. Finalmente, en la etapa de clasificación se decide cuándo los errores de predicción son significativos, lo que lleva al descubrimiento de anomalías y a la detección de indicios de actividades sospechosas. A continuación se describe en detalle cada uno de estos procesos.

IV-A. Entrenamiento y selección de algoritmos de predicción

El servidor dedicado provee las familias de métodos de predicción implementadas en el componente de análisis de SELFNET [49], entre las que se encuentran modelos basados en medias móviles, autorregresión y alisado de series temporales. Con el fin adaptar la estrategia de detección a la no estacionalidad de los datos a estudiar, antes de calcular cada proyección se decide el modelo que mejor se ajuste a ellas. Por lo tanto, se requiere de una colección de muestras de referencia a partir de la cual sea posible extraer las características más relevantes de las series temporales [45]. En la etapa de entrenamiento se ajusta el clasificador que decide el mejor método de predicción. Se ha implementado la combinación de árboles de decisión conocida como Árboles Aleatorios (del inglés *Random Forest*) [50] por su precisión, eficacia al estudiar cantidades grandes de muestras, y capacidad de operar con eficiencia al considerar listas largas de atributos. Cada muestra considerada para este fin es representada por los 100 atributos extraídos de una serie temporal de referencia, la cual pertenece a la clase que identifica el algoritmo de predicción que mejor ha sido capaz de operar sobre ella. La clase se obtiene al analizar la serie temporal con todos los algoritmos de predicción, determinándose aquél con el que ha mostrado menor media simétrica del porcentaje de error absoluto o sMAPE (del inglés *Symmetric Mean Absolute Percentage Error*). Nótese que este es el principal criterio adoptado en la M3-Competition [46], y se expresa como:

$$sMAPE = 200\,\% \sum_{t=1}^{n} \frac{|\hat{x}_t - x_t|}{|\hat{x}_t| + |x_t|} \quad (3)$$

donde n es la observación más reciente de la serie temporal $x_1, x_2, \cdots, x_n$ de métricas agregadas a pronosticar. Por simplicidad y eficiencia se ha considerado un horizonte de predicción de una observación $t+1$. Finalmente, cabe destacar que una de las principales desventajas de los clasificadores basados en Árboles Aleatorios es su tendencia al sobreajuste (del inglés *overfitting*). DroidSentinel reduce este problema con una fase de selección previa conducida por un algoritmo voraz de discriminación de características [51] y su evaluación en base a su significancia en procesos predictivos [52].

IV-B. Predicción y Adaptación al estado de la red

Partiendo de la premisa de que, tras cada observación, la distribución de la información monitorizada puede mostrar cambios representativos, debe asumirse el despliegue de una estrategia de predicción adaptativa. La comunidad investigadora típicamente ha afrontado este problema distinguiendo dos grandes paradigmas [53]: adaptación activa y pasiva. En el primero se busca los puntos de inflexión en los cambios registrados en el entorno, y en el segundo simplemente se asume su existencia, lo que implica el continuo recalibrado de los algoritmos analíticos implementados. La existencia de ataques DDoS basados en inundación con capacidad de disimular incrementos abruptos del volumen de datos generado [31] nos lleva a hipotetizar que la adaptación activa es más efectiva, donde la precisión de la detección no recae meramente en la identificación de puntos de inflexión. Por este motivo ha sido adoptada por la versión actual de DroidSentinel, dejando la exploración de estrategias de adaptación alternativas como líneas abiertas de trabajo futuro. En particular, tras cada observación se deciden el método de predicción que mejor se ajusta a los atributos de la serie temporal y el mejor calibrado de sus parámetros de ajuste. La etapa de selección se lleva a cabo en base al modelo de clasificación basado en Arboles Aleatorios construido durante el proceso de entrenamiento. De este modo, si por ejemplo la serie temporal a estudiar presenta una tendencia clara, se prioriza el uso de técnicas de modelado que se comporten mejor bajo estas circunstancias (v.g. Holt-Winters) frente a aquellos que la analizan con mayor dificultad (v.g. doble alisado exponencial) [54].

El calibrado del método de predicción seleccionado se lleva a cabo mediante un algoritmo genético básico [55], donde la población a evolucionar son sus posibles ajustes, y el genotipo representa el conjunto de atributos a configurar (ver Tabla I). La función de aptitud devuelve el sMAPE calculado a partir de las observaciones previas al instante actual mostrado por la serie temporal. Este algoritmo itera sobre operaciones de cruce basadas en pivotar los genotipos centrándose en un punto decidido arbitrariamente, y en mutaciones aleatorias de genes en su descendencia. Las condiciones de parada son que se alcance un número de iteraciones máximo o que algún individuo muestre aptitud optima ($sMAPE = 0$). A partir del ajuste que provee el mejor individuo se lleva a cabo la inferencia de la evolución de la métrica analizada para la siguiente observación. La población final resultante sirve de población inicial para las siguientes ejecuciones del algoritmo.

Tabla I: Descripción del algoritmo genético implementado

Rasgo	Características a destacar
Individuo	El genotipo es un vector donde cada gen es un parámetro de ajuste del algoritmo de predicción.
Población inicial	La primera población se genera aleatoriamente. En los siguientes calibrados, la población inicial es la resultante de la ejecución previa.
Aptitud	sMAPE del mejor calibrado.
Selección	Selección proporcional a la función de aptitud
Cruce	Intercambio de genes a partir de un pivote arbitrario.
Mutación	Mutación uniforme de un gen arbitrario.
Condición de parada	Máximo número de iteraciones o solución óptima.

IV-C. Umbrales Adaptativos y Clasificación

En la fase de clasificación y toma de decisiones se decide la naturaleza de las series temporales construidas a partir de métricas agregadas. En el marco de DroidSentinel se asume que una observación es discordante cuando es inesperada, es decir, cuando la diferencia entre el pronóstico de su evolución y la observación realizada es significativa. Debido a que la proyección de valores continuos habitualmente arroja errores de predicción, la principal dificultad en esta etapa de procesamiento de información consiste en establecer el umbral que separa lo normal de lo sospechoso. Afortunadamente el proyecto SELFNET [49] integra capacidades analíticas ampliamente aceptadas por la comunidad investigadora para dicho fin, habiéndose implementado la metodología de construcción de umbrales adaptativos descrita en [56]. En base a esto se asume el siguiente intervalo de predicción:

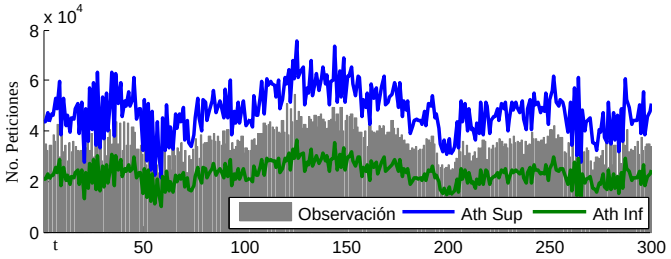
$$Ath_{up} = \hat{x}_{n+1} + K\sqrt{\sigma^2(E_t)} \quad (4)$$

$$Ath_{low} = \hat{x}_{n+1} - K\sqrt{\sigma^2(E_t)} \quad (5)$$

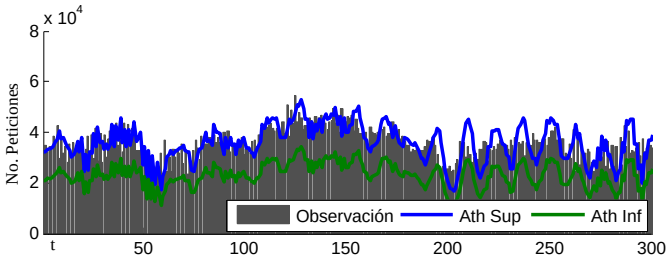
donde $\hat{x}_{n+1}$ es el pronóstico realizado para el horizonte $n+1$, E_t es la distancia euclidiana entre $\hat{x}_{n+1}$ y x_{n+1} y K es el parámetro de ajuste que rige el nivel de restricción con el que actúa el sensor. Cuando la observación realizada supera Ath_{up} o es inferior a Ath_{low}, DroidSentinel detecta una anomalía relacionada con la posible participación del dispositivo en ataques DDoS. Ésta es notificada al usuario y a los niveles de toma de decisiones y gestión de incidencias de SELFNET, donde en base a esta información y la conciencia situacional adquirida, es posible coordinar acciones de autoorganización similares al las descritas en [58], como por ejemplo incrementar la vigilancia en las regiones bajo los efectos de nodos comprometidos o el despliegue de tarros de miel.

V. EXPERIMENTACIÓN

La eficacia de la estrategia desarrollada ha sido probada mediante una metodología de evaluación experimental, en la que se ha medido el impacto en su eficacia al variar diferentes parámetros de ajuste: métricas básicas, métricas agregadas y ajuste del nivel de restricción de los umbrales predictivos estimados. Dado que principal objetivo de la investigación realizada ha sido el desarrollo de una estrategia adaptativa de detección de ataques de denegación de servicio en el extremo origen, los resultados descritos en este artículo se centran en la precisión lograda al analizar tráfico de diferente naturaleza. Las actividades monitorizadas son etiquetadas como normales



(a) Predicción Convencional



(b) Predicción Adaptativa

Figura 3: Ejemplo de adaptación a no estacionariedad.

(legítimas) y anómalas (sospechosas). Por lo tanto, el estudio realizado es análogo a la evaluación de clasificadores binarios, la cual frecuentemente es llevada a cabo desde la perspectiva que ofrecen dos métricas (sensibilidad, especificidad) y su mejor relación, tal y como es frecuente en la bibliografía, estimada mediante el índice de Youden [57]. La primera determina la capacidad de señalar actividades de naturaleza maliciosa como sospechosas, y la segunda permite valorar su capacidad de reconocer actividades de naturaleza normal como legítimas.

La colección de pruebas considerada reúne muestras de tráfico saliente capturado en 35 dispositivos Android diferentes pertenecientes a usuarios distintos, todos ellos alumnos de la facultad de informática de la Universidad Complutense de Madrid. Fueron tomadas en distintos periodos de monitorización, separados en intervalos de 1, 3 y 5 días, en diferente franja horaria. Por motivos de privacidad, cada muestra publicada contiene únicamente las métricas básicas y agregadas consideradas en el estudio realizado. Se ha considerado una granularidad de 3 minutos por observación, y una longitud de 120 observaciones por serie temporal, resultando una colección de 210 muestras. Principalmente recopilan la actividad normal del usuario, aunque algunos de ellos han optado por incluir subprocesos que simulen navegación HTTP (del inglés *web scraping*), por ejemplo *TrafficGenerator*, *Gehaxelt*, *BeforeRain*, etc. Se pidió a los usuarios que periódicamente ejecutaran ataques de denegación de servicio de 12 minutos TCP o UDP (4 observaciones) contra extremos de red virtualizados ubicados en un entorno aislado de la subred de la facultad de informática. En total se generaron 70 muestras de ataques. El tráfico era indistintamente dirigido contra un único punto (DoS) o contra varios de ellos (DDoS), para lo cual se valieron de varias herramientas de código abierto como *Warchild* o *TCP Attack*. El banco de pruebas y las herramientas utilizadas para su gestión está disponible en [11].

VI. RESULTADOS

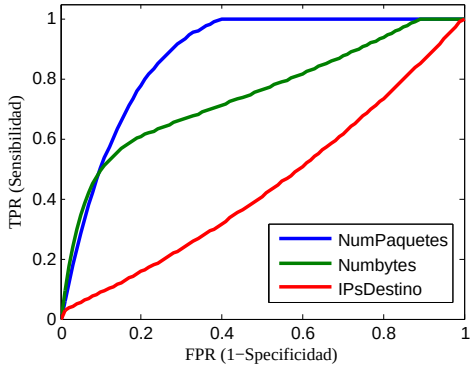
A continuación se ilustra un caso de estudio que facilita la comprensión de las características del esquema de predicción adaptativa propuesto. También se revisan los resultados obtenidos por DroidSentinel al analizar tráfico real.

VI-A. Caso de Estudio

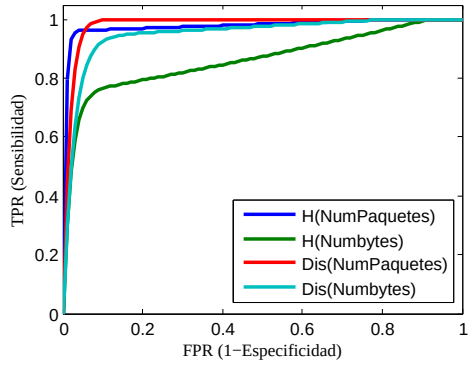
En la Fig. 3 se muestran los resultados obtenidos al calcular la significancia de la estimación del número de peticiones salientes legítimas registradas en un dispositivo Android. Para ilustrar con mayor claridad el ejemplo, se ha considerado una métrica de bajo nivel directamente extraída de los flujos de tráfico monitorizados, ya que están son más sensibles a cambios en el entorno protegido. En Fig. 3a se muestra el intervalo de predicción calculado sobre una configuración estática, definida en una etapa de calibrado que tiene lugar a lo largo de las primeras 20 observaciones. Nótese que por facilitar la comprensión del ejemplo, la serie temporal analizada presenta 300 observaciones, siendo mayor que las consideradas en la evaluación de la precisión del sistema. En ella se fija tanto el algoritmo de predicción, como sus parámetros de ajuste. Para K=1.35 el error de predicción medio es del 11.25 %; no obstante, el 40.3 % de las observaciones han sido erróneamente etiquetadas como amenazas. Por otro lado, en Fig. 3b tanto el algoritmo de predicción como su configuración se han recalibrado en cada nueva observación registrada. En este caso, para K=1.35 el error de predicción medio es del 19.2 %. Sin embargo, la tasa de falsos positivos registrada ha sido del 6.6 %. Esta mejora se debe a los continuos cambios en el modelo de tráfico construido, en los que se penaliza el error medio en pro de reducir el impacto de crecimientos/decrecimientos inesperados debidos a cambios en la distribución de datos analizada. Estos han sido frecuentes en el comportamiento de los diferentes usuarios que han contribuido a la recolección de muestras, y aparentemente inherentes al usos de dispositivos móviles.

VI-B. Eficacia

Los resultados obtenidos para las distintas métricas se muestran en Fig. 4 sobre el espacio ROC (del inglés *Receiver Operating Characteristic*). Este resume la relación entre la variación de la sensibilidad y especificidad registradas al variar el parámetro de ajuste que limita el intervalo de predicción, K en la experimentación realizada. El análisis de tráfico a partir de métricas básicas, en particular número de paquetes N, total de *bytes* transmitidos S y el número de destinos D, resultó en la eficacia mostrada en Fig. 4a. Los áreas bajo la curva ROC o AUC (*Area Under Curve*) observados fueron AUC(N)=86.3, AUC(S)=0.729 y AUC(D)=0.45, todos ellos calculados mediante una aproximación trapezoidal con un margen de error máximo de 0.05. Por lo tanto, sólo el estudio del total de paquetes enviado por los dispositivos ha sido medianamente viable, debido a que, al participar en un ataque distribuido, a menudo el número de víctimas contra las que se inyecta tráfico es reducido (normalmente un único elemento de red). Además, su carga útil no es muy diferente a la del tráfico legítimo, típicamente amplificada a partir de su paso por elementos de red explotados con fines de amplificación (por ejemplo, servidores DNS).



(a) Métricas básicas



(b) Entropía

Figura 4: Precisión de distintas métricas al variar K

En Fig. 4b se muestran los resultados en el espacio ROC obtenidos al estudiar dos métricas agregadas: entropía y distancia euclidiana. La efectividad del análisis basado en entropía ha dependido directamente de la métrica básica a partir de la cual ha sido calculada. Con la entropía del número de paquetes por flujo, los resultados obtenidos han mejorado considerablemente, registrándose AUC=0.985. Este parámetro ha permitido definir una tasa de acierto del 96.1 % y una tasa de falsos positivos del 4.1 %. Este hecho es debido a que los ataques ejecutados han alterado significativamente el número de peticiones del extremo origen a las víctimas. Esto conlleva un incremento importante del número de paquetes transmitidos, y por lo tanto una fluctuación en su grado de desorden en el tráfico monitorizado. Por otro lado, el análisis basado en la entropía del número de bytes por flujo ha arrojado resultados muy similares a los logrados con su métrica básica, observándose AUC=0.775. Esto es debido a que el tráfico legítimo en sí ya muestra una fluctuación constante sobre este parámetro, lo que lleva a que parte de las variaciones derivadas del ataque pasen mucho más desapercibidas.

En Fig. 4b se muestra también la eficacia de la propuesta al considerar como métrica agregada, la distancia euclidiana entre parámetros del tráfico saliente y entrante. Tal y como puede observarse, el estudio del número de paquetes ha resultado más eficaz, registrándose AUC=0.985. El mejor ajuste ha arrojado una tasa de acierto del 98.5 % y de falsos positivos del 0.7 %. Por otro lado, y a diferencia de las pruebas anteriores, el análisis de la diferencia entre bytes transmitidos y recibidos ha resultado mucho más preciso. En particular se ha observado AUC=0.9255, tasa de acierto del 91.4 % y tasa de falsos positivos del 9.32 %; lo que mejora con creces los resultados logrados con la métrica básica y el estudio de su entropía. Es importante resaltar que la gran eficacia de esta métrica agregada reside en el hecho de que los ataques de denegación de servicio ejecutados prácticamente sólo han involucrado tráfico saliente.

A raíz de los resultados observados es posible concluir que en la experimentación realizada se ha probado la eficacia de DroidSentinel al detectar ataques de denegación de servicio salientes. Cabe destacar la precisión obtenida mediante el estudio de métricas derivadas del número de paquetes por traza de tráfico y la comparativa de características del tráfico saliente y entrante.

VII. CONCLUSIONESS

A lo largo del artículo se ha introducido una estrategia de reconocimiento de ataques de denegación de servicio en su extremo origen. Esta se ha caracterizado por su capacidad de adaptación a procesos no estacionarios en la distribución de la información monitorizada, integración en un esquema de autoorganización de redes de nueva generación (aunque este escenario no ha sido explotado para servir a casos de uso más complejos), y su implementación como solución a sistemas Android. Para su evaluación se han considerado capturas de tráfico cedidas por diferentes usuarios reales, las cuales incluyen capturas de actividades habituales, y ataques de denegación de servicio basados en inundación lanzados desde distintas herramientas. El método propuesto ha sido capaz de distinguir las intrusiones con precisión, así como de adaptarse a la no estacionaridad inherente al uso de este tipo de tecnologías. No obstante, las propias restricciones establecidas por los principios de diseño anticipan una serie de carencias que deben ser tenidas en consideración de cara a su despliegue en casos de usos reales, como salvaguardar la privacidad de la información transmitida al servidor de análisis dedicado, la defensa de los canales de control o el fortalecimiento de la estrategia propuesta frente a ataques de evasión basados en imitación del tráfico legítimo.

AGRADECIMIENTOS

Los autores agradecen la financiación que les brinda el Programa Marco de Investigación e Innovación Horizonte 2020 de la Comisión Europea a través del proyecto SELFNET: *Framework for Self-Organized Network Management in Virtualized and Software Defined Networks* (H2020-ICT-2014-2.

REFERENCIAS

[1] CCN-CERT, “IA-16/17 Ciberamenazas y Tendencias. Edición 2017”. https://www.ccn-cert.cni.es/informes/informes-ccn-cert-publicos/2224-ccn-cert-ia-16-17-ciberamenzas-y-tendencias-edicion-2017.html.

[2] V.A.F. Almeida, D. Doneda, J.S. Abreu, “Cyberwarfare and Digital Governance”, *IEEE Internet Computing*, Vol. 21(2), pp. 68-71, 2017.

[3] E. Bertino, N. Islam, “Botnets and Internet of Things Security”, *Computer*, Vol. 50(2), pp. 76-79, 2017.

[4] C. Kolias, G. Kambourakis, A. Stavrou, J. Voas, “DDoS in the IoT: Mirai and Other Botnets”, *Computer*, Vol. 50(7), pp. 80-84, 2017.

[5] M. Antonakakis, T. April, et al. , “Understanding the Mirai Botnet”, Proc. *26th USENIX Security Symposium*, Vancouver, Canada, 2017.

[6] S. T. Zargar, J. Joshi, D. Tipper, “A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks”, *IEEE Communications Surveys & Tutorials*, Vol. 15(4), pp. 2046-2069, 2013.

[7] J. Maestre Vidal, A. Lucila Sandoval, L.J. García Villalba, "Adaptive Artificial Immune Networks for Mitigating DoS flooding Attacks", *Swarm and Evolutionary Computation*, Vol. 38, pp. 94-108, 2018.

[8] Q. Yan, F.R. Yu, Q. Gong, J. Li, "Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey, Some Research Issues, and Challenges", *IEEE Communications Surveys & Tutorials*, Vol. 18(1), pp. 602-622, 2016.

[9] D. Acarali, M. Rajarajan, N. Kmminos, I. Herwono, "Survey of approaches and features for the identification of HTTP-based botnet traffic", *Journal of Network and Computer Applications*, Vol. 76, pp. 1-15, 2016.

[10] L. Gavrilovska, V. Rakovic, V. Atanasovski, "Visions Towards 5G: Technical Requirements and Potential Enablers", *Wireless Personal Communications*, Vol. 87(3), pp. 731-757, 2016.

[11] DroidSentinel'. https://github.com/borjalor/DroidSentinel.

[12] EU SELFNET Project "Self-Organized Network Management in Virtualized and Software Defined Networks". Project reference: H2020-ICT-2014-2/671672. Funded under: H2020. http://www.selfnet-5g.eu.

[13] W. Wei, F. Chen, Y. Xia, G. Jin, "A rank correlation based detection against distributed reflection DoS attacks", *IEEE Communications Letteres*, Vol. 17, No. 1, pp. 173-175, 2013.

[14] M.H. Bhuyan, D. Bhattacharyya, J. Kalita, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection", *Pattern Recognition Letters*, Vol. 51, No. 1, pp. 1-7, 2015.

[15] C. Li, J. Yang, Z. Wang, F. Li, Y. Yang, "An empirical evaluation of information metrics for low-rate and high-rate DDoS attack detection", Proc. *IEEE Global Communications Conference*, San Diego, US, 2015.

[16] L. Xiao, W. Wei, W. Yang, Y. Shen, X. Wu, "A protocol-free detection against cloud oriented reflection DoS attacks", *Soft Computing*, Vol. 21(13), pp. 3713-3721, 2017.

[17] D.C. MacFarland, C.A. Shue, A.J. Kalafut, "The best bang for the byte: Characterizing the potential of DNS amplification attacks", *Computer Networks*, Vol. 116, pp. 12-21, 2017.

[18] L. Wei, Q. Li, Y. Jiang, J. Wu, "Towards mitigating Link Flooding Attack via incremental SDN deployment", Proc. *IEEE Symposium on Computers and Communications*, Messina, Italy, 2016.

[19] V. Matta, M. Di Mauro, M. Longo, "DDoS Attacks With Randomized Traffic Innovation: Botnet Identification Challenges and Strategies", *IEEE Transactions on Information Forensics and Security*, Vol. 12(8), pp. 1844-1859, 2017.

[20] G. Vormayr, T. Zseby, J. Fabini, "Botnet Communication Patterns", *IEEE Communications Surveys & Tutorials*, Vol. 19(4), pp. 2768-2796, 2017.

[21] N. Hoque, D.K. Bhattacharyya, J.K. Kalita, "Botnet in DDoS Attacks: Trends and Challenges", *IEEE Communications Surveys & Tutorials*, Vol. 17(4), pp. 2242-2270, 2015.

[22] H. Luo, Y. Lin, H. Zhang, M. Zukerman, "Preventing DDoS attacks by identifier/locator separation", *IEEE Networks*, Vol. 27(6), pp. 60-65, 2013.

[23] C. Wang, T.N.N. Miu, X. Luo, J. Wang, "SkyShield: A Sketch-Based Defense System Against Application Layer DDoS Attacks", *IEEE Transactions on Information Forensics and Security*, Vol. 13(3), pp. 559-573, 2018.

[24] S. Khanna, S.S. Venkatesh, O. Fatemieh, C.A. Gunter, "Adaptive Selective Verification: An Efficient Adaptive Countermeasure to Thwart DoS Attacks", *IEEE/ACM Transactions on Networking*, Vol. 20(3), pp. 715-728, 2011.

[25] Y. Wang, I.R. Chen, J.H. Cho, A. Swami, K.S. Chan, "Trust-Based Service Composition and Binding with Multiple Objective Optimization in Service-Oriented Mobile Ad Hoc Networks", *IEEE Transactions on Services Computing*, Vol. 10(4), pp. 660-672, 2017.

[26] K. Wang, M. Du, S Maharjan, Y. Sun, "Strategic Honeypot Game Model for Distributed Denial of Service Attacks in the Smart Grid", *IEEE Transactions on Smart Grid*, Vol. 8(5), pp. 2474-2482, 2017.

[27] M.H. Jhaveri, O. Cetin, C. Gañán, T. Moore, M. Van Eeten, "Abuse Reporting and the Fight Against Cybercrime", *ACM Computing Surveys*, Vol. 49(4), No. 68, 2017.

[28] P. Holgado, V.A. Villagrá, L. Vázquez, "Real-time multistep attack prediction based on Hidden Markov Models", *IEEE Transactions on Dependable and Secure Computing*, (in proof) DOI:10.1109/TDSC.2017.2751478, 2017.

[29] A. Saied, R.E. Overill, T. Radzik, "Detection of known and unknown DDoS attacks using Artificial Neural Networks", *Neurocomputing*, Vol. 172, pp. 385-393, 2016.

[30] W.L. Al-Yaseen, Z.A. Othman, M.Z.A.Nazri, "Real-time multi-agent system for an adaptive intrusion detection system", *Pattern Recognition Letters*, Vol. 85, pp. 56-64, 2017.

[31] E. Adi, Z. Baig, P. Hingston, "Stealthy Denial of Service (DoS) attack modelling and detection for HTTP/2 services", *Journal of Network and Computer Applications*, Vol. 91, pp. 1-13, 2017.

[32] P. Ferguson, D. Senie, "Network ingress filtering: Defeating denial of service attacks which employ IP source address spoofing". IETF RFC 2827, May 2000.

[33] J. Mirkovic, G. Prier, P. Reiher, "Source-End DDoS Defense", Proc. *2nd IEEE International Symposium on Network Computing and Applications*, Cambridge, MA, US, 2003.

[34] T.M. Gil, M. Poletto, "MULTOPS: a data-structure for bandwidth attack detection", Proc. *10th USENIX Security Symposium*, Washington, DC, US, Vol. 10, No.3, 2001.

[35] S.A. Mehdi, J. Khalid, S.A. Khayam, "Revisiting Traffic Anomaly Detection Using Software Defined Networking", Proc. *14th International Symposium on Recent Advances in Intrusion Detection*, Menlo Park, CA, US, pp. 161-180, 2011.

[36] R. Jin, B. Wang, "Malware Detection for Mobile Devices Using Software-Defined Networking", Proc. *2nd GENI Research and Educational Experiment Workshop*, Salt Lake City, UT, US, pp. 81-88, 2013.

[37] N.M. Alenezi, M.J. Reed, "Uniform DoS traceback", *Computers & Security*, Vol. 45, No. 1, pp. 17-26, 2014.

[38] A.R. Kiremire, M.R. Brust, V.V. Phoha, "Using network motifs to investigate the influence of network topology on PPM-based IP traceback schemes", *Computer Networks*, Vol. 72, No. 1, pp. 14-32, 2014.

[39] D.E. Denning, "Framework and principles for active cyber defense", *Computers & Security*, Vol. 40, pp. 108-113, 2014.

[40] P. Farina, E. Cambiaso, G. Papaleo, M. Aiello "Are mobile botnets a possible threat? The case of SlowBot Net", *Computers & Security*, Vol. 58, pp. 268-283, 2017.

[41] M.A. Sotelo Monge, J. Maestre Vidal, L.J García Villalba, "Entropy-Based Economic Denial of Sustainability Detection", *Entropy*, Vol. 19(12), No. 649, 2017.

[42] I. Ozcelik, R.R. Brooks, "Deceiving entropy based DoS detection", *Computers & Security*, Vol. 48, No. 1, pp. 234-245, 2015.

[43] G. Suarez-Tangil, J. E. Tapiador, P. Peris-Lopez, S. Pastrana, "Power-aware anomaly detection in smartphones: An analysis of on-platform versus externalized operation", *Pervasive and Mobile Computing*, Vol. 18, pp. 137-151, 2015.

[44] H.Y. Lateef, A. Imran, M.A. Imran, L. Giupponi, M. Dohler, "LTE-advanced self-organizing network conflicts and coordination algorithms", *IEEE Wireless Communications*, Vol. 22(3), pp. 108-117, 2015.

[45] TSFRESH: Time Series Feature extraction based on scalable hypothesis tests. https://github.com/blue-yonder/tsfresh.

[46] S. Makridakis, M. Hibon, "The M3-Competition: results, conclusions and implications", *International Journal of Forecasting*, Vol. 16(4), pp. 451-476, 2000.

[47] R. Hofstede, P. Celeda, B. Trammell, I. Drago, R. Sadre, A. Sperotto, A. Prass, "Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX", *IEEE Communications Surveys & Tutorials*, Vol. 16(4), pp. 2037-2064, 2014.

[48] C. Shannon, "A mathematical theory of communication", *Bell Systems Technical Journal*, Vol. 27(3), pp. 379-423, 1948.

[49] M.A. Sotelo Monge, J. Maestre Vidal, L.J. García Villalba, "Reasoning and Knowledge Acquisition Framework for 5G Network Analytics", *Sensors*, Vol. 17(10), No. 2405, 2017.

[50] L. Breiman, "Random Forests", *Machine Learning*, Vol. 45(1), pp. 5-32, 2001.

[51] B. Hu, X. Li, S. Sun, M. Ratcliffe, "Attention Recognition in EEG-Based Affective Learning Research Using CFS+KNN Algorithm", *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, Vol. 15(1), pp. 38-45, 2018.

[52] M.A. Hal, "Correlation-Based Feature Selection for Machine Learning", Ph.D dissemination, University of Waikato, 1999.

[53] G. Ditzler, M. Roveri, C. Alippi, R. Polikar, "Learning in Nonstationary Environments: A Survey", *IEEE Computational Intelligence Magazine*, Vol. 10(4), pp. 12-25, 2015.

[54] E.S. Gardner, "Exponential smoothing: The state of the art—Part II", *International Journal of Forecasting*, Vol. 22(4), pp. 637-666, 2006.

[55] A. Kamrani, R. Wang, R. Gonzalez, "A Genetic Algorithm Methodology for Data Mining & Intelligent Knowledge Acquisition", *Computers & Industrial Engineering*, Vol. 40(4), pp. 361-337, 2001.

[56] S. Makridakis, S.C. Wheelwright, R.J. Hyndman, "Forecasting: Methods and Applications", John Wiley & Sons, 1998.

[57] T. Fawcet, "An introduction to ROC analysis", *Pattern Recognition Letters*, Vol. 27(8), pp. 861-874, 2006.

[58] M. Gil Perez et al., "Dynamic reconfiguration in 5G mobile networks to proactively detect and mitigate botnets", *IEEE Internet Computing*, Vol. 21(5), pp. 25-36, 2017.

A Summary of TriFlow: Triaging Android Applications using Speculative Information Flows

Omid Mirzaei*, Guillermo Suarez-Tangil†, Juan Tapiador‡ and Jose M. de Fuentes‡
*‡ Universidad Carlos III de Madrid, Spain
† University College London, UK
Email: *omid.mirzaei@uc3m.es, †guillermo.suarez-tangil@ucl.ac.uk, ‡{jestevez, jfuentes}@inf.uc3m.es

***Abstract*—Information flows in Android can be effectively used to give an informative summary of an application's behavior and are shown to be extremely useful in characterizing risky behaviors. However, identifying flows in an application is computationally expensive. Thus, it is critical to prioritize applications that are likely to pose a risk. In this work, we develop a triage mechanism to rank applications considering their potential risk. Our approach, called TRIFLOW, relies on static features that are quick to obtain. TRIFLOW combines a probabilistic model to predict the existence of information flows with a metric of how significant a flow is in benign and malicious apps. Based on this, TRIFLOW provides a score for each application that can be used to prioritize analysis. TRIFLOW also provides an explanatory report of the associated risk. We evaluate our tool with a representative dataset of benign and malicious Android apps. Our results show that it can predict the presence of information flows very accurately and that the overall triage mechanism enables significant resource saving.**

***Index Terms*—Android security, malware analysis, information flow, app triage.**

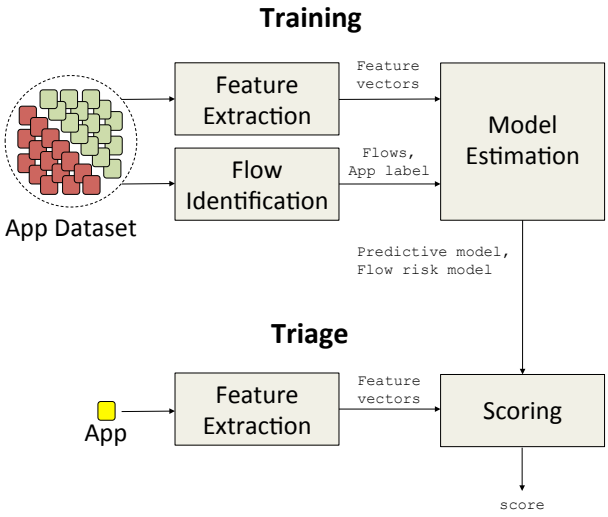


Fig. 1: Architecture of the proposed system.

I. INTRODUCTION

The sheer number of Android apps available in current markets, along with the ratio at which new apps are submitted, makes impossible to manually analyze all of them. Automated analyses also have their limitations and some techniques might require a substantial amount of time per app. This has motivated the need for a multi-staged analysis pipeline in which apps should be initially triaged to allocate resources intelligently and guarantee that the analysis effort is devoted to those samples that potentially have more security interest.

One of the salient features of Android's security model is its permission-based access control system. Apps may request access to security- and privacy-sensitive resources in their manifest files. However, using permissions alone to assess risk has important limitations. To overcome these limitations, we present TRIFLOW [1], a lightweight IF-based triage mechanism to identify Android apps with potentially dangerous behaviors. Here, we do rely on information flows which can be effectively used to give an informative summary of an application's behavior. TRIFLOW introduces the notion of *speculative information flows*. This means that TRIFLOW extracts some features from apps, and, then, predicts the existence of a flow based on them (Fig. 1). Each predicted flow is then scored by TRIFLOW in terms of its potential risk, which depends on the flow's observed prevalence in goodware and malware. We make our results and our implementation of TRIFLOW publicly available in GitHub[1].

The rest of this paper is organized as follows. Section II describes our approach for fast triage of apps. In Section III, we present and discuss the results of our evaluation and Section IV concludes the paper.

II. APPROACH

A high-level view of TRIFLOW is shown in Fig. 1. The system is first trained using a dataset of benign and malicious apps. The goal of this phase is to obtain the two items that will be later used to score apps (as shown by Eq. 1), including a predictive model (θ_f) that outputs the probability of each possible information flow to be present in the app given a feature vector obtained from the app's code, and a model ($I(f)$) that measures how informative each information flow is considering its relative frequency of occurrence in malware and benign apps. Thus, the score of an app (bottom part of Fig. 1) is calculated by simply multiplying each flow's likelihood by its weight and summing up for all flows observed in the app.

[1] https://github.com/OMirzaei/TriFlow

$$\text{score}(a) = \sum_{f} \theta_f I(f). \quad (1)$$

Information flows are predicted based on some static features whose presence correlate with the presence of flows and are quick to obtain. We have considered source and sink API methods for this purpose. The major goal of flows weighting is to identify malicious information flows, i.e. those flows which are common in malware but rare in benign apps. Another goal is to filter out those flows which are common in both classes. To reach these goals, TRIFLOW assigns weights to flows based on their prevalence in malware and benign apps.

III. EVALUATION

Three main experiments are conducted to show the performance of our proposed system which are related to info-flow prediction, info-flow weighting, and triaging. We have used Drebin as our malware dataset and the apps in Google Play as our benign dataset (Table I).

TABLE I: Overview of the datasets used in this work.

Type	Dataset	Type	Samples
Malware (MW)	Drebin [2]	Malware	5,560
Goodware (GW)	Google Play	Goodware	11,456
		Total	17,016

Mode	Split	Ratio	Samples
Modeling (Training)	4,000 MW 4,000 GW	1:1	8,000
Triage (Testing)	1,560 MW 7,456 GW	1:5	9,016

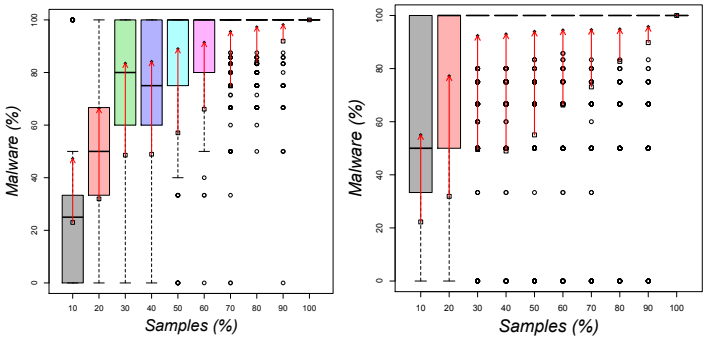
We have evaluated our info-flow based predictive model on three different datasets using non-stratified 5-fold cross validation (Table II). According to the results, 90% of flows in both benign and malware datasets have prediction errors less than 0.25.

TABLE II: Flow prediction error statistics after 5-fold cross-validation.

Dataset	Mean	Std. Dev.	Median
Drebin	0.0861	0.1272	0.0278
GooglePlay	0.0361	0.0734	0.0094
All	0.0376	0.0784	0.0089

In addition, we have calculated weights for 31,175 unique info-flows obtained from training dataset. From this amount, very rare flows which observed mainly in malware have weights higher than 1. This means that we can only rely on a small portion of flows to identify risky applications. For further analysis, we have also extracted some high-weighted flows.

TRIFLOW scores can be used to rank apps, prioritize analysis and save significant computational resources. Ideally, a triage system should maximize the time analysts spend on analyzing potentially harmful applications. To quantify the performance of our triage system, we have carried out an experiment and have compared our tool with RSS, a permissions-based risk scoring mechanism (Fig. 2). We have assumed that we have a market operator which receives batches of



(a) RSS (Sarma et al. [3]). (b) TRIFLOW.

Fig. 2: Results of the triage evaluation.

10 samples of apps per minute and its analysts are capable of processing different percentages of apps in each batch. Next, all apps in the batch are scored and a percentage of top ranked apps is given to the analysts for a deeper analysis. Then, we have measured what percentage of samples in that final block are malware and have repeated this process 900 times obtaining one percentage each time. Here, boxes show the distribution of values while red arrows represent the gain each scoring mechanism achieves with respect to a random prioritization policy (square symbols). Our results show that TRIFLOW can prioritize more malware samples per batch for all workloads than RSS.

TRIFLOW also provides analysts with a breakdown of score into the SuSi source and sink categories, and flows that contribute to the score along with their amount and percentage of contribution.

IV. CONCLUSION

In this paper, we designed and implemented a novel tool, called TRIFLOW, that automatically scores Android apps based on a forecast of their information flows and their associated risk. This scoring can prioritize analysis and save computational resources. Experimental results show that TRIFLOW can prioritize more malware than any other risk scoring mechanisms and is capable of spotting more malware only by observing a few number of them in each batch.

ACKNOWLEDGMENT

This work was supported by the MINECO grant TIN2016-79095-C2-2-R and by the CAM grant S2013/ICE-3095.

REFERENCES

[1] O. Mirzaei, G. Suarez-Tangil, J. Tapiador, and J. M. de Fuentes, "Triflow: Triaging android applications using speculative information flows," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*. ACM, 2017, pp. 640–651.

[2] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, and K. Rieck, "Drebin: Effective and explainable detection of Android malware in your pocket," in *Network and Distributed System Security Symposium (NDSS)*, 2014.

[3] B. P. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android permissions: A perspective combining risks and benefits," in *17th ACM Symposium on Access Control Models and Technologies*, ser. SACMAT '12, 2012, pp. 13–22.

IagoDroid: atacando el triaje de aplicaciones Android maliciosas

Alejandro Calleja*, Alejandro Martín†, Héctor D. Menéndez‡ and Juan Tapiador*, David Clark‡
*Departamento de Informática
Universidad Carlos III de Madrid
Madrid, Spain
Email: {accortin,jestevez}@inf.uc3m.es
†Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
Madrid, Spain
Email: alejandro.martin@uam.es
‡Department of Computer Science
University College London
London UK
Email: {h.menendez,david.clark}@ucl.ac.uk

***Resumen*—La detección de malware juega un papel fundamental para evitar la infección de sistemas o dispositivos. Cuando se detecta una muestra maliciosa, esta suele ser clasificada según una serie de familias conocidas, con el objetivo aplicar las contramedidas necesarias. Muchos métodos de clasificación de malware se basan en algoritmos de aprendizaje automático que son entrenados con un conjunto de muestras previamente etiquetadas según diferentes familias de malware. En este trabajo se resume una investigación ya publicada donde se presenta IagoDroid, un prototipo de herramienta enfocada a demostrar la posibilidad de forzar clasificadores de familias para asociar una muestra a una familia a la que no pertenece. IagoDroid emplea una búsqueda heurística basada en un algoritmo genético para construir un nuevo vector de características cuya clasificación se aleja de la familia original.**

***Index Terms*—Malware classification, Adversarial learning, Genetic algorithms, Iagodroid**

Tipo de contribución: Trabajo publicado. Título original: ***Picking on the family: Disrupting android malware triage by forcing misclassification*** [1]

I. Introducción

Android, desde sus inicios, ha sido atacado por virus diseñados específicamente para esta plataforma. El gran número de usuarios cuyos dispositivos utilizan esta plataforma o las propias características de esta plataforma, como la posibilidad de instalar aplicaciones de terceros, hace de Android un blanco perfecto para los creadores de *malware*.

Con el objetivo de detectar estas aplicaciones maliciosas, son numerosas las herramientas existentes y enfoques propuestos para analizar muestras y obtener una evaluación que determine su naturaleza. Sin embargo, este proceso no se detiene aquí. Una vez detectada una muestra de malware maliciosa, resulta necesario conocer las contramedidas convienentes para mitigar los efectos y daños. Este proceso es conocido como *triaje*, y consiste en asignar muestras, ya detectadas como maliciosas, a categorías o familias de malware que comparten comportamiento, intenciones o detalles de implementación.

El proceso de triaje de aplicaciones maliciosas es importante y también sensible, asignar una aplicación a una categoría incorrecta puede conllevar aplicar contramedidas incorrectas y, por tanto, no conseguir detener la carga maliciosa a tiempo. Han sido múltiples las propuestas realizadas para realizar este proceso con técnicas de aprendizaje automático. En este caso, nos enfocamos en un método de clasificación de aplicaciones Android para describir, probar y demostrar que un ataque contra métodos basados en este enfoque es factible y puede ser llevado a cabo.

En el paper original de esta investigación, presentamos IagoDroid[1], un protipo de herramienta que es capaz de atacar RevealDroid [2], una herramienta de clasificación de aplicaciones Android basado en la extracción de diferentes características estáticas, incluyendo *taint analysis*, y en un algoritmo de clasificación de aprendizaje automático. IagoDroid, la herramienta propuesta, emplea un algoritmo evolutivo con el objetivo de producir etiquetados incorrectos de muestras en este clasificador. En esta investigación también proponemos una segunda contramedida contra este tipo de ataques. Esta contramedida consiste en una modificación de la herramienta de clasificación y la cual hemos denominamos RevealDroid*.

II. IagoDroid: atacando clasificadores basados en aprendizaje automático

Tradicionalmente, las herramientas basadas en aprendizaje automático para la detección o clasificación de aplicaciones en diferentes familias se basan en la extracción de un conjunto de características. Fruto de estas, cada aplicación queda representada por un vector de características. En el caso de RevealDroid, estas características incluyen Intent-Actions, llamadas al sistema o flujos de información extraídos con FlowDroid [3].

El vector formado por cada aplicación es la entrada que recibe el algoritmo de clasificación. Una manipulación en este vector puede conllevar a la entrega de un resultado diferente por parte del método de clasificación. Sin embargo,

[1]El código fuente puede obtenerse desde: https://github.com/hdg7/IagoDroid

la manipulación de este vector implica la modificación de la aplicación. IagoDroid se centra en buscar modificaciones de este vector que no alteran la semántica de la aplicación original, lo cual se consigue mediante modificaciones incrementales: se añaden funcionalidades en el código que nunca llegan a ser ejecutadas. Por ejemplo, puede declararse una llamada al sistema en el código dentro de un predicado opaco que nunca llega a ser realmente ejecutada.

II-A. Algoritmo genético

La elección de las modificaciones necesarias para producir este cambio de familia no es un proceso trivial. IagoDroid emplea un algoritmo genético para realizar una búsqueda heurística con el objetivo de producir un nuevo vector cuya nueva clasificación sea diferente de la original.

De este modo, se ha diseñado un algoritmo genético donde la primera población se inicializa según el vector original de la muestra cuya familia se desea modificar. Debido a las características concretas del problema a resolver, los operadores de cruce, mutación y reproducción han sido implementados con el objetivo de obtener individuos válidos, es decir, individuos cuyo vector únicamente añade funcionalidades y no elimina ninguna (lo que forzaría a romper la semántica original de la aplicación). Por otro lado, el operador de selección es elitista, tomando los n mejores individuos en cada generación.

En el caso de la función *fitness*, se han probado dos enfoques distintos. En el primero, se utiliza una función de pertenencia a la clase original según el resultado del clasificador entrenado. De esta forma se persigue que los nuevos vectores se alejen de la clase original, es decir, se buscan nuevos vectores cuyo grado de pertenencia a la clase original disminuya, de forma que queden asignados a nuevas clases o familias de malware. En segundo lugar, la búsqueda genética se encamina a buscar nuevos vectores cuya clasificación pertenezca a una nueva familia previamente seleccionada.

III. Experimentación

Para la experimentación se han utilizado 1919 muestras pertenecientes a 19 familias distintas del dataset DREBIN [4]. Como algoritmo de aprendizaje automático para realizar el proceso de entrenamiento, se ha utilizado el algoritmo C4.5 [5], al igual que se hace en la herramienta RevealDroid. Una vez lanzado el algoritmo genético para evaluar qué número de vectores (representantes de las aplicaciones contenidas en el dataset mencionado) pudieron ser transformados para que su etiquetado pasara a ser el de una nueva familia distinta a la original. Para la mayoría de aplicaciones, una única generación del algoritmo permitió conocer un nuevo vector cuya clasificación pertenecía a una familia distinta. Además, en la mayoría de los casos, la modificación de un único punto (característica) en el vector de la muestra permitió el cambio de familia. En el caso de realizar un cambio a una nueva familia concreta se observó que las familia origen y destino determinan la viabilidad de esta transformación.

Con el objetivo de comprobar que los cambios indicados por el nuevo vector pueden ser implementados, un conjunto de muestras fue alterado manualmente. Para ello, cada aplicación tiene que ser descomprimida para acceder a los ficheros de código fuente (*DEX)*. En segundo lugar, estos ficheros que contienen bytecode, son transformados en código *smali*, el cual permite leer el código en un formato legible. Una vez aplicadas las modificaciones sobre este código, un nuevo ejecutable puede generarse con la herramienta *Backsmali*[2].

IV. Contramedida

Una vez demostrada la viabilidad de llevar a cabo el ataque descrito, en esta investigación también se propone una contramedida para evitar este tipo de ataques. Esta medida consiste en el entrenamiento de un conjunto de árboles predictores, en vez de un único árbol. Cada uno de estos árboles toma un conjunto aleatorios de características, de forma que el atacante desconoce qué características son tomadas por qué árbol. Los resultados muestran que la contramedida, denominada RevealDroid*, es capaz de detectar las muestras que están tratando de confundir al clasificador hasta en el 99 % de los casos.

Agradecimientos

Esta trabajo ha sido desarrollado gracias a los siguientes proyectos: EphemeCH (MINECO TIN2014-56494-C4-4-P) y CIBERDINE (CM S2013/ICE-3095), ambos a cargo de European Regional Development Fund FEDER; SeMaMatch EP/K032623/1 y InfoTestSS EP/P006116/1 from EPSRC; SPINY (MINECO TIN2013-46469-R) y SMOG-DEV (MINECO TIN2016-79095-C2-2-R) y Justice Programme of the European Union (2014-2020) 723180 – RiskTrack – JUST-2015-JCOO-AG/JUST-2015-JCOO-AG-1. Los contenidos de esta publicación son de única responsabilidad de su autores y no reflejan de ninguna forma la opinión de la Comisión Europea.

Referencias

[1] A. Calleja, A. Martín, H. D. Menéndez, J. Tapiador, and D. Clark, "Picking on the family: Disrupting android malware triage by forcing misclassification," *Expert Systems with Applications*, vol. 95, pp. 113–126, 2018.

[2] J. Garcia, M. Hammad, B. Pedrood, A. Bagheri-Khaligh, and S. Malek, "Obfuscation-resilient, efficient, and accurate detection and family identification of android malware," *Department of Computer Science, George Mason University, Tech. Rep*, 2015.

[3] S. Arzt, S. Rasthofer, C. Fritz, E. Bodden, A. Bartel, J. Klein, Y. Le Traon, D. Octeau, and P. McDaniel, "Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps," *Acm Sigplan Notices*, vol. 49, no. 6, pp. 259–269, 2014.

[4] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C. Siemens, "Drebin: Effective and explainable detection of android malware in your pocket." in *Ndss*, vol. 14, 2014, pp. 23–26.

[5] J. R. Quinlan, *C4. 5: programs for machine learning*. Elsevier, 2014.

[2]https://github.com/JesusFreke/smali

Patch for Nothing and Slow for Free

Antonio Nappa
Minsait by Indra
anappa@indra.es

Jorge López Hernández-Ardieta
Minsait by Indra
jlhardieta@minsait.com

Mayank Dhiman
Independent Researcher
mayank.dhiman1@gmail.com

Abstract- **In this paper we analyze the performance impact of the countermeasures against the (in)famous micro architectural side-channel attacks known as Spectre and Meltdown. We also analyze the patching procedure and its effectiveness. Despite the big impact these vulnerabilities have and will have in the future, the effort to make the patching process reliable and definitive is far from perfect. We show a maximum performance loss of 47% and that it is not possible to fix all the vulnerabilities in all the affected systems.**

***Index Terms*- side-channel, vulnerability patching**

Tipo de contribución: *Investigación en desarrollo*

I. Introduction

Recently the micro architectural vulnerabilities codenamed Spectre [2] and Meltdown [1] have appeared in the wild. These vulnerabilities exploit speculative execution, which is a feature of modern CPUs, which allows filling the execution pipeline and achieving high performance using three techniques: out-of-order execution, multiple branch prediction and dataflow analysis.

After the disclosure of CVE-2017-5715, CVE-2017-5753, CVE-2017-5754 [6,7,8], which dates the beginning of 2018, software vendors and hardware makers have made a considerable effort to fix or mitigate these vulnerabilities, because they allow reading protected memory and executing arbitrary code. A few days after disclosure, different patches and mitigations have appeared and, according to independent reports and declarations from vendors, the users, who decide to install the patch, will incur in performance degradation of their systems.

In this study, we perform an analysis of the available fixes, their effectiveness and their performance impact. We show in our analysis that after applying all the available fixes the system remains vulnerable to at least one version of the vulnerability.

II. Approach

We have chosen two COTS hypervisors, VirtualBox 5.2.6 r120293, which at the time of writing is the latest available version and, VMWare ESXi 6.5 (Free Edition) to test the effectiveness of the patches and their performance impact. The target operating system is GNU Linux Ubuntu 16.04 LTS a widespread operating system, especially in development and research environments which will be supported by Canonical until April 2021. The vulnerable version ships with the version 4.10.0-28 of the Linux Kernel, while the patched version uses 4.13.0-36. The ESXi system is equipped with a dual Intel Xeon E5-2643 4-Core CPU with a working frequency of 3.3GHz with 64GB of ECC RAM. While the VirtualBox system is equipped with a dual core Intel i5-6300U at 2.4GHz with 16GB of non-ECC RAM. We have chosen a server-like CPU and a desktop-like CPU. We have installed two virtual machines for each hypervisor, one with the latest available kernel that includes the fixes for the vulnerabilities and one that does not have the update.

Identify a vulnerable system. To identify a vulnerable or patched system we have used a script which is available on GitHub [9] that has become a de-facto standard to check the Spectre and Meltdown vulnerability exposure on Linux systems. The script confirms that our patched version should be not vulnerable to any of the vulnerability variants, including Spectre v1, which should be mitigated thanks to the lfence serialization instruction, which limits speculative execution.

Application Instrumentation. We have used a combination of tools to instrument and measure the performance of web applications and databases. Prometheus [11] was our choice to retrieve performance metrics of CPU and memory. Then we have developed two basic applications, a Prometheus instrumented web server and a Prometheus enabled MySQL database. To plot the performance indicators we used Grafana, an off-the-shelf tool for advanced data visualization.

Existing PoCs. There is a downloadable PoC of Spectre v1 vulnerability (CVE-2017-5753) which is used as a running example in the Spectre paper [2]. The PoC is useful only to prove that it is possible to retrieve information from the cache and read a piece of memory of its same process, which is not accessed through the logic of the program. There are also other PoCs available from Google Project Zero [12].

III. Preliminary Results

In this section, we show the results of our load test on the patched and not patched system. For the web load test, we have used Apache Bench, a benchmarking tool from the Apache Foundation. We have performed six tests of six million requests each with 100 concurrent threads, every test lasts approximately 10 minutes. For the database load test, we have used the Malicia Dataset [3] database to perform a realistic test on a dataset released to more than 70 institutions worldwide. The (read-only) test includes several queries including SELECT and JOIN. Every test lasts approximately 10 minutes we have repeated the test 6 times.

In Table I we show the results of our macro benchmark, which uses as a metric the number of requests/transactions the system could handle per second. In the case of the web load test, the patched systems incur in an average performance loss in number of processed requests per second of 27% in ESXi

Table I
PERFORMANCE TEST RESULTS
MACRO-BENCHMARK

Test Type	ESXi Patched	ESXi Vulnerable	VirtualBox Patched	VirtualBox Vulnerable
Web Test	5568 **req/s**	7658 **req/s**	5090 **req/s**	7542 **req/s**
DB Test	447 t**xn/s**	495 **txn/s**	438 **txn/s**	491 **txn/s**

Table II
PERFORMANCE TEST RESULTS
MICRO-BENCHMARK
(AVG. LOSS)

Test Type	ESXi Patched	VirtualBox Patched
Web Test	45%	47%
DB Test	9%	10%

and 33% in VirtualBox. While average performance loss of the database test is 9% in ESXi and 11% in VirtualBox.

Table II shows the results of our micro benchmark, which is the performance loss in the ratio between CPU usage and requests served per second. The web test is the one that shows a higher performance loss, 45% in ESXi and 47% in VirtualBox, with respect to the macro benchmark, while the database test shows similar results. Further detailed plots can be found at [13].

IV. PRELIMINARY DISCUSSION

Our measurement was performed on two different systems one server-like and one desktop-like. This, to show how different environments perform after applying the vulnerabilities patches. We believe that the results of two different systems cannot be compared against each other because there are architectural differences. Nonetheless, what can be compared is a performance loss trend, which is clearly visible in both testing platforms.

In a further (read and write) test that we have performed with MySQL and sysbench [10], we discovered a lower performance loss (3%) or in some cases the patched system was performing slightly better (1-2%) than the vulnerable one. This preliminary result opens a discussion on how to properly evaluate the performance of two different kernels or CPUs, which, besides having a fix for a vulnerability also may include other modifications which can alter the measurement.

Patch Effectiveness. The only fully functional PoC available for Spectre v1 appears to work also on patched systems, with a less reliable score, but still retrieving the secret value. We plan to finish the development of other PoCs to confirm that the patches are effective.

Patch Availability. The high demand for a prompt reaction to these potentially catastrophic vulnerabilities from software vendors and hardware maker has made the patching solution very difficult to achieve in a proper way. The information on how to patch the systems is very scattered and it is difficult for the average user to have a comprehensive idea on how to fix properly these issues. According to a recent study [4], patching should be as automated as possible t o achieve good coverage of the vulnerable population.

V. PRELIMINARY CONCLUSIONS

The discovery of Spectre and Meltdown has shaken CPU makers in such a way that they might rethink the whole Von-Neumann architecture to overcome vulnerabilities of this type, which when patched can make systems up to 47% slower for CPU bounded programs. The fact that the available PoC in the wild for Spectre v1 does not get patched but only partially mitigated. And also, that in order to fix Spectre v2 an update of the CPU microcode is needed or a recompilation of all the software with retpoline option of GCC [5] makes the patching procedure more difficult for users. We believe that applying a patch to a kernel or to a CPU is a very delicate task because the side effects are not completely understandable at the beginning of the patching activity. However, during the review process of this paper patches for the tested system have improved significantly [14]. Unfortunately, there is still a lack of a complete and definitive solution. For these reasons the patching procedures need an improvement, at the time of writing it was very difficult for us to find a reliable source to understand how to patch all the vulnerabilities.

REFERENCES

[1] Lipp, Moritz and Schwarz, Michael and Gruss, Daniel and Prescher, Thomas and Haas, Werner and Mangard, Stefan and Kocher, Paul and Genkin, Daniel and Yarom, Yuval and Hamburg, Mike. “Meltdown”, ArXiv e-prints, January 2018

[2] Kocher, Paul and Genkin, Daniel and Gruss, Daniel and Haas, Werner and Hamburg, Mike and Lipp, Moritz and Mangard, Stefan and Prescher, Thomas and Schwarz, Michael and Yarom, Yuval. “Spectre Attacks: Exploiting Speculative Execution”, ArXiv e-prints, January 2018

[3] Antonio Nappa, M. Zubair Rafique, Juan Caballero. "The MALICIA Dataset: Identification and Analysis of Drive-by Download Operations" In International Journal of Information Security, June 2014

[4] Antonio Nappa, Richard Johnson, Leyla Bilge, Juan Caballero, Tudor Dimitras. “The Attack of the Clones: A Study of the Impact of Shared Code on vulnerability Patching” In Proceedings of the 36th IEEE Symposium on Security & Privacy, San Jose, CA, May 2015

[5] GCC-7.3 https://www.phoronix.com/scan.php?px=GCC-7.3-Released&page=news_item

[6] CVE-2017-5715 https://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2017-5715

[7] CVE-2017-5753 https://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2017-5753

[8] CVE-2017-5754 https://www.cve.mitre.org/cgi-bin/cvename.cgi?name=2017-5754

[9] Spectre and Meltdown Checker https://github.com/speed47/spectre-meltdown-checker

[10] SysBench https://github.com/akopytov/sysbench

[11] Prometheus https://prometheus.io/

[12] Google Project Zero – Meltdown and Spectre https://goo.gl/aHMR9C

[13] Paper Additional Resources – http://jnic18spectre.ddns.net

[14] Spectre and Meltdown Ubunrtu Status https://goo.gl/FZsBJ9

Evaluación y selección de un ecosistema de herramientas para un enfoque preventivo y continuo en modelos de desarrollo seguro de software

José Andrés
Félix de Sande
ViewNext S.A.
CENIT Cáceres (PCTEx)
Av. Universidad s/n
Cáceres
jafelix@viewnext.com

José Carlos
Sancho Núñez
Cátedra ViewNext-UEx
Escuela Politécnica
Av. Universidad s/n
Cáceres
jcsanchon@unex.es

Andrés
Caro Lindo
Universidad de Extremadura
Escuela Politécnica
Av. Universidad s/n
Cáceres
andresc@unex.es

Resumen- **Anticipación, previsión y prevención son características claves cuando se refieren a la seguridad en el desarrollo de sistemas software [1]. La transición desde un modelo de desarrollo basado en la realización de actividades reactivas hacia un modelo evolucionado basado en los tres conceptos anteriores implica extender las actividades de seguridad a lo largo de todas las fases de nuestro ciclo de desarrollo seguro (S-SDLC). Cada vez es mayor el grado de implantación de metodologías ágiles de desarrollo, en las que resulta clave el acceso continuo a la información del estado del producto en desarrollo. Además, el enfoque hacia la simplificación y automatización de los procesos, unido a la utilización de iteraciones cortas de desarrollo y entornos multitecnología y/o multicanal, genera la necesidad de contar con herramientas que permitan auditar de manera continua los riesgos de seguridad de un sistema software. Este trabajo se basa en el análisis de herramientas que den soporte a este proceso, bajo distintas variables de estudio. Como resultado, se presenta una valoración comparativa con el objetivo de facilitar la selección de un ecosistema lo más polivalente posible adecuado al contexto del sistema a desarrollar.**

***Index Terms*- S-SDLC, SAST, DAST, IAST**

Tipo de contribución: *Investigación en ciberseguridad / Contribuciones científicas originales*

I. Introducción

Desde hace varios años, la tendencia en ingeniería del software se orienta a la aplicación de modelos basados en metodologías ágiles que además alineen las estrategias de los equipos de desarrollo (Dev) y de operaciones/soporte (Ops). Se espera que ambos equipos colaboren a lo largo de todo el ciclo de vida. En los últimos años, a este binomio se añade una dimensión adicional, la seguridad (DevSecOps). El objetivo consiste en integrar las actividades relacionadas con esta dimensión desde el primer momento. Se pretende disponer de un enfoque basado en la monitorización y prevención continua, así como en la automatización de procesos (figura 1). Todo ello sin afectar de manera traumática a la productividad de los equipos de desarrollo, sin perder de vista los objetivos de negocio de la compañía [2].

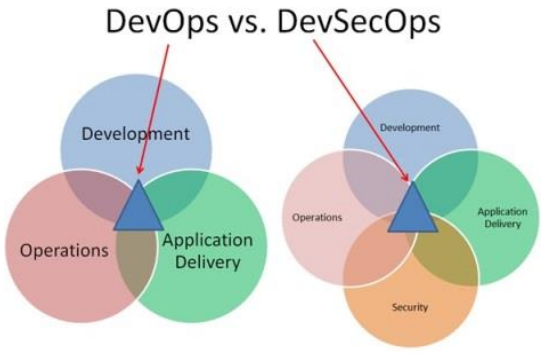


Fig. 1. DevOps vs DevSecOps

Bajo este paradigma es absolutamente crucial contar con un ecosistema de herramientas que permita conocer, de manera continua, el estado de riesgo o cómo de vulnerable es un sistema que aún está en desarrollo. Y, además, que dicho conjunto de herramientas sea lo más completo y polivalente posible. En este sentido, el conjunto de herramientas debe abarcar el mayor número de tecnologías, incluyendo un amplio espectro de detección de riesgos de seguridad. Respetando, además, los estándares existentes de clasificación de vulnerabilidades de seguridad (OWASP Top 10, SANS Top 25, CWE, CVE).

Partiendo del modelo de desarrollo de software seguro propuesto en [3], este trabajo se centra principalmente en la evaluación de herramientas aplicables a cuatro de las actividades contempladas en el área llamada *'Metodología SDL'* (figura 2). Concretamente las actividades de *'Revisión de Diseño'*, *'Revisión de Desarrollo'*, *'Testing de Seguridad'* y *'Validación de Salidas'*, y más en detalle en la evaluación de herramientas de análisis estático (SAST), análisis dinámico (DAST), análisis interactivo (IAST) y de ingeniería inversa. En cualquier caso, los resultados son exportables a cualquier otra particularización de modelos S-SDLC, puesto que el ecosistema de herramientas analizadas es fácilmente acoplable a modelos que cumplan los estándares y normas habituales. Cualquier investigación relacionada con S-SDLC encontrará de gran utilidad la comparativa realizada en este trabajo.

Fig. 2. Modelo de desarrollo de software seguro Viewnext

II. PROCESO DE EVALUACIÓN DE HERRAMIENTAS. CRITERIOS DE ANÁLISIS

El análisis se ha realizado en dos fases:

Una primera fase donde se ha realizado una investigación inicial identificando las principales alternativas comerciales y libres existentes actualmente, realizando una comparativa basada en los siguientes parámetros de análisis:

- Coste de licenciamiento/restricciones de uso por licencia
- Nivel de cobertura (OWASP Top 10, CWE, SANS Top 25)
- Grado de implantación y uso
- Soporte de la comunidad o del fabricante
- Frecuencia de actualización
- Integración con entornos DevOps (entornos CI, herramientas).
- Extensibilidad mediante plugins o addons
- Ayuda que presenta para la identificación y resolución de los riesgos detectados (recomendaciones y buenas prácticas)
- Lenguajes soportados

Como complemento a la fase anterior, una segunda etapa basada en la realización de pruebas de concepto (PoC) y posterior análisis de los resultados obtenidos en las mismas. Para ello, se seleccionaron las herramientas con una mayor cobertura de los parámetros analizados en la primera fase. Estas PoCs se realizaron mediante la evaluación de ejemplos reales de aplicaciones que se encontraban en fase de desarrollo, analizando informes reales, y completando el estudio con las siguientes variables adicionales:

- Calidad de los informes.
- Fiabilidad de los resultados obtenidos frente al nivel de cobertura teórico
- Falsos positivos
- Usabilidad

Tras ello, se presentan una serie de conclusiones sobre los escenarios de aplicación recomendados para las herramientas seleccionadas en la última fase.

III. SAST, DAST, IAST Y RASP. DIFERENTES ENFOQUES PARA LA EVALUACIÓN CONTINUA DE VULNERABILIDADES

Las herramientas de análisis estático de la seguridad de las aplicaciones (SAST), están orientadas a la detección de vulnerabilidades en fases tempranas del desarrollo. Estas tareas se realizan mediante el análisis estático del código fuente o versiones compiladas del mismo para la detección de patrones de diseño o codificación que evidencien riesgos o vulnerabilidades explotables. Las herramientas basan su estrategia en la realización de pruebas de caja blanca, en concreto, durante las actividades de *'Revisión de Diseño'* y *'Revisión de Desarrollo'* del Modelo de Desarrollo de Software Seguro. Generalmente detectan debilidades más sencillas de resolver, pero también tiene un mayor índice de falsos positivos. Además, no pueden detectar riesgos derivados de los propios entornos de ejecución. En la mayoría de los casos permiten su integración con entornos integrados de desarrollo (IDEs) y en sistemas de integración y evaluación continua del código (sistemas IC).

Por su parte, las herramientas de análisis dinámico de la seguridad de las aplicaciones (DAST), están orientadas a la detección de vulnerabilidades en versiones en ejecución de las aplicaciones, como se pone de manifiesto en [4]. Suelen utilizarse en fases más avanzadas del ciclo de desarrollo (durante las actividades de *'Testing de Seguridad'* o *'Validación de Salidas'* del Modelo de Desarrollo de Software Seguro). Basan su estrategia en la realización de pruebas de caja negra y generalmente detectan debilidades más complejas de resolver. Son capaces de detectar riesgos derivados de los propios entornos de ejecución y en algunos casos permiten su integración en los entornos integrados de desarrollo (IDEs) y en sistemas de integración y evaluación continua del código (sistemas IC). La figura 3 resume estas características.

SAST	DAST
Test de seguridad (Caja Blanca)	Test de seguridad (Caja Negra)
Requiere código fuente	Requiere ejecutable
Encuentra vulnerabilidades en fases iniciales del SDLC	Encuentra vulnerabilidades hacia el final del SDLC
Corrección de vulnerabilidades menos caro	Corrección de vulnerabilidades más caro
No descubre problemas relativos a entorno y de tiempo de ejecución	Descubre problemas relativos a entorno y de tiempo de ejecución
Normalmente soporta cualquier tipo de software	Normalmente escanea solo aplicaciones / servicios web y similares

Fig. 3. SAST vs DAST

Las herramientas de análisis interactivo de la seguridad (IAST), se basan en la instrumentalización de las aplicaciones en ejecución o en la utilización de agentes software en los propios entornos de ejecución. Estas herramientas permiten obtener información sobre la ejecución de las aplicaciones y sus eventos de seguridad.

Un tipo especial de herramientas IAST son las RASP (herramientas de auto-protección de aplicaciones en ejecución). Estas añaden una componente reactiva ante la detección de un mal uso o ante un indicio de ataque. Así, permiten, por ejemplo, cerrar la sesión del usuario de manera automática, provocar el cierre de una aplicación, el envío de

alertas al equipo encargado de la seguridad de la misma o incluso avisos de advertencia al potencial atacante.

IV. HERRAMIENTAS DE INGENIERÍA INVERSA. COMPLEMENTO EN ENTORNOS MÓVILES

Las herramientas de ingeniería inversa permiten decompilar y analizar los recursos de una aplicación. E incluso modificar el código binario para evaluar el comportamiento de la aplicación antes escenarios no contemplados durante su desarrollo. Estas herramientas se pueden utilizar para la evaluación de aplicaciones. Además, deben acompañarse de herramientas que permitan evitar la utilización de las técnicas anteriores para la obtención de información o evaluación de vulnerabilidades y potenciales vectores de ataque con intereses malintencionados. Este tipo de herramientas tiene especial importancia cuando las soluciones desarrolladas son aplicaciones para dispositivos móviles, o soluciones de escritorio donde la aplicación y sus recursos quedan más expuestos al acceso completo por parte de usuarios con intereses perniciosos. Los ataques principales provocados por la utilización de técnicas de ingeniería inversa afectan principalmente a los principios de integridad (modificación de código y reempaquetado de aplicaciones para evadir los controles de acceso o ejecutar lógica alternativa) y confidencialidad (obtención de información sensible como contraseñas, claves criptográficas o datos personales o información sobre los recursos de la aplicación como algoritmos utilizados o servicios invocados) [5].

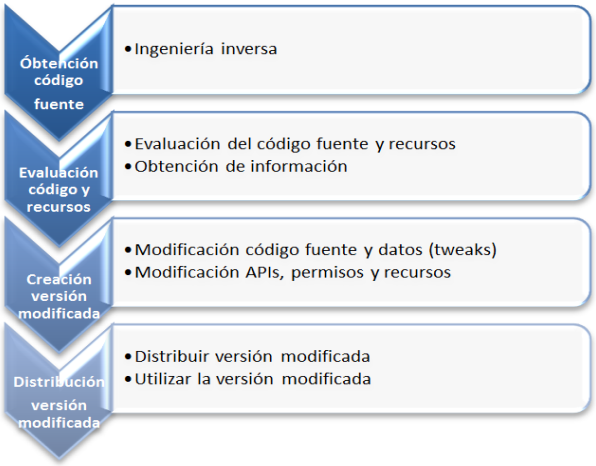


Fig. 4. Ataque por ingeniería inversa

Entre las herramientas de ingeniería inversa para entornos Android, destacan, ***ApkExtractor***, que permite extraer una aplicación instalada en un dispositivo Android, ***dex2jar*** que permite decompilar ficheros con extensión *.dex* (contienen el código ejecutable en la máquina virtual Dalvik de Android), y ***ApkTool***, que permite decompilar el código fuente de la aplicación, realizar cambios en el mismo y volver a generar el *apk* con el código malicioso incluido (un *tweak* es una remodelación para añadir nuevas funcionalidades, que en ocasiones puede incluir código malicioso). Para entornos iOS, destaca la combinación de ***iSpy*** con ***Cycript***, que permite el volcado de clases, la monitorización de instancias en ejecución, el bypass automático de detección de fallos o certificados SSL o la detección de llamadas a funciones vulnerables. ***Cycript*** permite la creación e inyección de *tweaks.*

Frente a estas herramientas, se deben utilizar herramientas como ***DexGuard*** (para Android) o ***iGuard*** (para iOS), que proporcionan protección frente a análisis estáticos o dinámicos de una aplicación, mediante la utilización de técnicas de ofuscación y encriptación, chequeos para validación de entornos (*jailbreak* o *rooting*), validación de certificados o detección de herramientas de depuración o inyección. Existe una versión opensource de estas herramientas (válida sólo para código Java), ***ProGuard,*** recomendable para aplicaciones híbridas o aplicaciones de escritorio.

V. COMPARATIVA HERRAMIENTAS SAST

Durante la fase primera de análisis se realizó una labor de investigación y evaluación de la información teórica disponible en la web sobre herramientas SAST. Se identificaron las principales alternativas comerciales y libres existentes (más habituales), obteniendo una tabla comparativa en base a los criterios citados. El resultado fue el siguiente:

- *SonarQube v7.0*

Tipo licencia/ restricciones de uso: Dispone de varias versiones (Community, Developer y Enterprise), además de una versión cloud para proyectos de código abierto. Coste de las licencias según estimación LOC (lines of code).

Lenguajes Soportados: *Community Edition* - incluye 9 lenguajes (Java, JS, C#, TypeScript, Flex, Python, PHP, Web y XML). *Developer Edition* - añade 7 lenguajes adicionales a la versión Community (C/C++, Objective-C, PL/SQL, ABAP, VB.NET, TSQL y Swift). *Enterprise Edition* - añade 4 lenguajes adicionales a la versión Developer (COBOL, PL/I, RPG y VB6).

Estándares normativos soportados: OWASP Top 10, SANS Top 25, MITRE-CWE.

Nivel adopción: Muy usada tanto en proyectos de software libre como en proyectos comerciales.

Soporte comunidad/fabricante: Comunidad muy activa (blogs, foros, repositorio de documentación y guías). Para las versiones de pago soporte incluido del proveedor (opcional en la versión *Developer*).

Frecuencia actualización: Actualización de versiones muy frecuente (última version liberada en Feb-18).

Integración entornos DevOps y otras herramientas: Integración con herramientas de gestión del ciclo técnico de las aplicaciones (Maven, Gradle, Ant, Makefile o MSBuild). Integración con sistemas IC (Jenkins, Bamboo, VS Team Foundation Server, Travis CI…). Integración con varios SCMs (CVS, SVN, Git, ClearCase, Jazz RTC...). Integración con herramientas de notificación y comunicación colaborativa (slack, e-mail, instant messaging...).

Extensibilidad (plugins o addons): *SonarLint* - Plugin para los IDEs más extendidos (Eclipse, IntelliJ, VisualStudio, Atom) para evaluación de código según se desarrolla. Amplio catálogo de plugins para integración con otras herramientas, cobertura de pruebas, reporting, analizadores externos o gestión de la autenticación / autorización, por ejemplo.

Nivel usabilidad y ayuda para la corrección de errores: Permite un nivel de parametrización y configuración amplio mediante la adaptación de cuadros de mando, quality gates, quality profiles y configuración de reglas. Su integración con IDEs de desarrollo facilita su uso. Presenta asistencia con recomendaciones, ejemplos y resaltado de código erróneo.

- *Kiuwan Code Security*

Tipo licencia/ restricciones de uso: No tiene una versión community, pero ofrece versiones de evaluación.

Lenguajes Soportados: Amplio abanico de lenguajes soportados (+20) incluyendo entre otros: JAVA, JS, PHP, SAP, COBOL, C/C++, C#, Android, iOS, Swift, VB.NET, ASP.NET, Python, PL/SQL, RPG…

Estándares normativos soportados: OWASP Top 10, MITRE-CWE, MISRA, NIST, PCI-DSS, CERT.

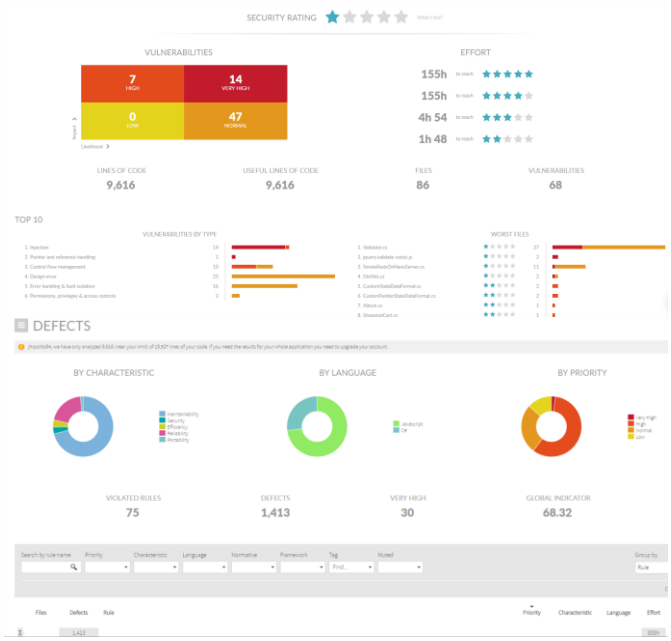


Fig. 5. Resultado análisis Kiuwan

Nivel adopción: Muy usada en proyectos comerciales.

Soporte comunidad/fabricante: Soporte incluido del proveedor. Cuenta con documentación, blog y foros de uso.

Frecuencia actualización: Actualización de versiones muy frecuente (tanto versiones menores como mayores).

Integración entornos DevOps y otras herramientas: Integración con herramientas de gestión del ciclo técnico de las aplicaciones (Maven, Gradle, Ant...). Integración con sistemas IC (Jenkins, Bamboo, VS Team Foundation Server...). Integración con varios SCMs (SVN, Git, RTC...)

Extensibilidad (plugins o addons): Plugin para los IDEs más extendidos (Eclipse, IBM RAD y Visual Studio) para evaluación de código según se desarrolla. Dispone de una herramienta adicional gratuita para evaluación de librerías open source utilizadas.

Nivel usabilidad y ayuda para la corrección de errores: Permite un nivel de parametrización y configuración amplio mediante la adaptación de cuadros de mando, SLAs, y configuración de reglas. Además, su integración con IDEs de desarrollo facilita su uso. Presenta asistencia con recomendaciones, ejemplos y resaltado de código erróneo.

- *IBM Security AppScan Source v9*

Tipo licencia/ restricciones de uso: Versión de pruebas disponible. Versiones para instituciones educativas (https://ibm.onthehub.com/). Incluido en IBM Cloud Platform.

Lenguajes Soportados: Amplia gama de lenguajes soportados, incluyendo: JAVA, COBOL, C/C++, JS, PHP, Perl, PL/SQL, .NET (C#, VB.NET, ASP.NET), VB6, ColdFusion, Objective C, Android… Soporte para frameworks de desarrollo: Struts, Spring, J2EE, EJB, JSF, JAX-WS, JAX-RS…

Estándares normativos soportados: OWASP Top 10, SANS Top 25, MITRE-CWE, PCI-DSS, DISA/STIG.

Nivel adopción: Muy usada en proyectos comerciales.

Soporte comunidad/fabricante: Soporte incluido del proveedor. Dispone de un 'Knowledge Center'.

Frecuencia actualización: Muy frecuente

Integración entornos DevOps y otras herramientas: Integración con herramientas de gestión del ciclo técnico de las aplicaciones (Maven, Gradle, Ant, Makefile...). Integración con sistemas IC y sistemas de seguimiento de tickets/bugs como Rational ClearQuest, HP Quality Center, RTC o Team Foundation Server). Integración con varios SCMs (CVS, SVN, Git, ClearCase, Jazz RTC...). Dispone de una interfaz por línea de comandos (CLI) y un API, lo que permite integrarlo con numerosas herramientas.

Extensibilidad (plugins o addons): Plugin para los IDEs más extendidos (Eclipse, IBM RAD, IBM MobileFirst, Visual Studio) para evaluación de código según se desarrolla.

Nivel usabilidad y ayuda para la corrección de errores: Permite un nivel de parametrización y configuración amplio mediante la adaptación de los cuadros de mando e informes y configuración de reglas. Además, su integración con IDEs de desarrollo facilita mucho su uso. Presenta asistencia con recomendaciones, ejemplos y resaltado de código erróneo.

- *Otras soluciones comerciales (Microfocus Fortify y CheckMarx Static Code Analysis)*

Tipo licencia/ restricciones de uso: Ambas disponen de

versiones on-premise y cloud, así como versiones de evaluación.

Lenguajes Soportados: Amplia gama de lenguajes soportados, incluyendo: JAVA, C/C++, JS, PHP, Scala, PL/SQL, .NET (C#, VB.NET, ASP.NET), ColdFusion, Objective C, Swift, Python. *Fortify* tiene soporte para COBOL y SAP, pero no para Android, mientras que *Checkmarx* soporta Android, Groovy o Ruby, pero no COBOL ni SAP.

Estándares normativos soportados: *Fortify:* OWASP Top 10, SANS Top 25, MITRE-CWE, PCI-DSS. *Checkmarx:* OWASP Top 10, SANS Top 25, MITRE-CWE, PCI-DSS, MISRA, HIPAA, BSIMM, CxSAST.

Nivel adopción: Ambas herramientas, nivel de utilización medio en proyectos comerciales.

Soporte comunidad/fabricante: En ambas herramientas se ofrece soporte por parte del proveedor. *Fortify:* Dispone de una comunidad con blogs, artículos y guías de uso (techbeacon.com). *Checkmarx*: No parece disponer de una comunidad, pero sí un blog con artículos y recursos.

Frecuencia actualización: Muy frecuente

Integración entornos DevOps y otras herramientas: Ambas tienen integración con herramientas de gestión del ciclo técnico de las aplicaciones (Maven, Gradle, Ant, ...), integración con sistemas IC y servidores de compilación, así como con sistemas de bug-tracking/ticketing. Además de soporte para varios SCMs. Ambas disponen de una interfaz API, lo que permite integrarlo con numerosas herramientas.

Extensibilidad (plugins o addons): Ambas tienen plugins de integración con los IDEs más extendidos para evaluación de código según se desarrolla. *Fortify*: Además dispone de un Marketplace con más plugins y addons. *Checkmarx*: Tiene además una herramienta para evaluación de librerías open source utilizadas.

Nivel usabilidad y ayuda para la corrección de errores: Ambas permiten un nivel de parametrización y configuración amplio mediante la adaptación de cuadros de mando e informes y configuración de reglas. Además, su integración con IDEs de desarrollo facilita mucho su uso. Presentan asistencia con recomendaciones, ejemplos y resaltado de código erróneo.

- *Soluciones para aplicaciones móviles (MobSF y QARK)*

Ambas herramientas se basan en licencias Opensource (GPL v2.0 y Apache License respectivamente). *MobSF* da soporte para Android, iOS y Windows Mobile, mientras que *QARK* sólo da soporte a Android. Ambas realizan tanto análisis SAST como DAST. Las dos herramientas están en versiones Beta, por lo que el grado de implantación es limitado. *MobSF* recibe actualizaciones frecuentes (última versión liberada en Dic-17), mientras que la última versión de *QARK* data de Sep-15. Las dos requieren conocimientos más avanzados para su utilización al ejecutarse o instalarse por línea de comandos, y presentan información de ayuda para la resolución de las vulnerabilidades detectadas.

Durante la segunda fase del análisis se seleccionaron las herramientas *SonarQube* y *Kiuwan* como herramientas multitecnología junto con *MobSF* como herramienta orientada a dispositivos móviles para realizar una PoC de confirmación de resultados sobre el código fuente de uno de los módulos desarrollados para la plataforma de gestión integral de un cliente de Viewnext, del sector energético.

Tras la realización de las pruebas de concepto se comprueba que tanto *Kiuwan* (figura 5) como *Sonar* presentan una gran usabilidad, presentando los resultados en cuadros de mando con widgets configurables, y permitiendo la definición de líneas base o la exportación de resultados a informes. En cuanto a los resultados, se comprueba *Kiuwan* tiene una orientación más clara hacia revisiones de seguridad, detectando mayor número de vulnerabilidades [6], mientras que *Sonar* es una herramienta de propósito más general que incluye aspectos de seguridad, pero también de calidad y mantenibilidad. Ambas herramientas por el carácter estático de sus revisiones, generan falsos positivos que es necesario revisar de manera manual. *Sonar* arroja valores más altos en este sentido [7]. Dichos defectos requieren una revisión manual por parte del usuario, pudiendo marcarlos como tal una vez comprobados.

La herramienta *MobSF* presenta también una interfaz visual donde se debe cargar el código de la aplicación a revisar (.apk en el caso de una aplicación Android). Una vez cargada, se muestran los resultados en un cuadro de mando.

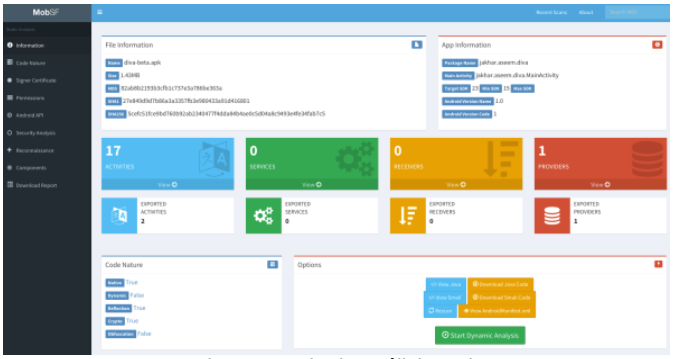


Fig. 6. Resultado análisis MobSF

El informe de resultados (figura 6) muestra información general de la aplicación (nombre, tamaño, versiones de la aplicación y del SDK), permisos requeridos, actividades, servicios, así como escuchadores y generadores de eventos. Muestra información sobre el estado del código (ofuscado, encriptado). Con respecto al análisis de código fuente, presenta un listado de los bugs clasificados por severidad e incluyendo una descripción del riesgo que implica e indicaciones para su resolución.

VI. COMPARATIVA HERRAMIENTAS DAST

La fase de evaluación teórica de herramientas comerciales y libres existentes, reflejó el siguiente resultado:

➢ *IBM Security AppScan v9*

Tipo licencia/ restricciones de uso: 3 versiones: Standard/Enterprise/Cloud. Tiene una versión de prueba.

Tests soportados: Pruebas de caja negra para aplicaciones web (Web 2.0, JS y AJAX) y servicios web (SOAP y REST). También tiene funcionalidades IAST. Realiza escaneos automáticos iterativos para elaborar el árbol de navegación. También permite la realización de test manuales.

Estándares normativos: Genera informes para 40 marcos regulatorios, incluyendo PCI-DSS, PA-DSS, ISO 27001/ISO 27002, HIPAA, GLBA y Basel II.

Nivel adopción: Muy usada en proyectos comerciales [8].

Soporte comunidad/fabricante: Soporte 24x7 y actualizaciones gratuitas durante 1 año.

Frecuencia actualización: Muy frecuentes, compromiso de actualización continua durante el período de vigencia de la licencia.

Integración entornos DevOps y otras herramientas: Integración con sistemas IC y sistemas de seguimiento de tickets/bugs. Incluido en IBM Cloud Platform.

Extensibilidad (plugins o addons): Permite la particularización y extensión a través del framework IBM Security AppScan eXtensions.

Nivel usabilidad y ayuda para la corrección de errores: Explicaciones detalladas de cada problema encontrado. Asistentes para configuración del escaneado. Se pueden definir y utilizar plantillas. Distintos tipos de informes de resultados (seguridad, cumplimiento de normativas/ estándares industriales, informes delta o informes a medida). La versión Enterprise incluye cuadros de mando y configuración de políticas de seguridad corporativas.

➢ *BurpSuite*

Tipo licencia/ restricciones de uso: Versión *Community,* con herramientas esenciales de revisión manual, y versión *Professional* que añade herramientas manuales avanzadas y escáner automático de vulnerabilidades web (figura 7).

Tests soportados: La versión *Community* analiza y muestra la estructura de un sitio web, para luego poder realizar pruebas manuales tipo 'man-in-the-middle', interceptando las peticiones y respuestas y modificando parámetros, propiedades y valores. Se pueden crear reglas para las intercepciones y modificaciones automáticas. La versión *Professional* chequea hasta 100 vulnerabilidades genéricas, y permite la realización de escaneados automatizados, pudiendo configurar los tipos de comprobaciones a realizar y el modelo de ejecución del escaneado. Incluye Fuzz Testing.

Estándares normativos: OWASP Top 10 (con variaciones para más de 100 tipos de vulnerabilidades distintas).

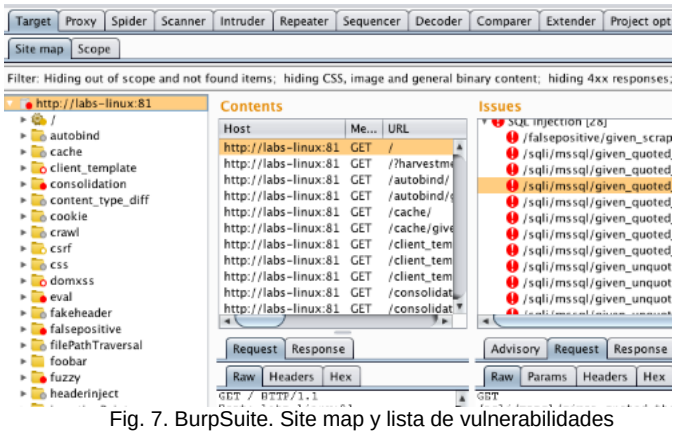


Fig. 7. BurpSuite. Site map y lista de vulnerabilidades

Nivel adopción: Muy usada en proyectos comerciales. Versión *Community* para desarrolladores y testers.

Soporte comunidad/fabricante: Soporte gratuito a través de su página. Existe un centro de soporte con amplia documentación y una comunidad muy activa.

Frecuencia actualización: Muy frecuentes, compromiso de actualización continua durante el período de vigencia de la licencia.

Integración entornos DevOps y otras herramientas: Se está desarrollando una extensión para la versión *Professional* para integrar la herramienta con los entornos CI más comunes.

Extensibilidad (plugins o addons): Dispone de un API para extender su funcionalidad y un repositorio con extensiones desarrolladas por la comunidad.

Nivel usabilidad y ayuda para la corrección de errores: La versión Enterprise incluye mucha información contextualizada para ayudar a corregir los problemas encontrados, así como permite exportar la información a informes que pueden ser particularizados.

➢ *Zed Attack Proxy (ZAP)*

Tipo licencia/ restricciones de uso: Opensource.

Tests soportados: Manuales (intercepting proxy) y automáticos (escaneado activo y pasivo). Orientada a su integración en entornos IC y automatización de pruebas. Incluye funcionalidades para pruebas de fuerza bruta y FuzzTesting, y una herramienta (spider) para la búsqueda de contenido oculto.

Estándares normativos: OWASP Top 10.

Nivel adopción: Muy utilizado sobre todo por desarrolladores y testers y en proyectos de software libre, y como como complemento a otras herramientas comerciales.

Soporte comunidad/fabricante: Existe un amplio soporte por parte de la comunidad que desarrolla y extiende ZAP, incluyendo documentación y tutoriales.

Frecuencia actualización: Actualizaciones muy frecuentes (semanales).

Integración entornos DevOps y otras herramientas: Existe un plugin para la integración con Jenkins.

Extensibilidad (plugins o addons): Existe un proyecto OWASP específico para el desarrollo de extensiones sobre ZAP (http://code.google.com/p/zap.extensions).

Nivel usabilidad y ayuda para la corrección de errores: Muy sencilla de utilizar, incluso por usuarios con poca experiencia. Genera informes HTML con los resultados.

- *Soluciones para aplicaciones móviles (Drozer /Needle)*

Ambas herramientas bajo licencia opensource (BSD). *Drozer* da soporte para Android, mientras que *Needle* está orientada a iOS. Desarrolladas y mantenidas por MWR InfoSecurity. Permiten lanzar exploits conocidos contra aplicaciones móviles, así como desarrollar y compartir exploits nuevos. Soporte a través de una cuenta de twitter (@mwrdrozer). Hay documentación y guías de uso. Actualización frecuente (últimas actualizaciones noviembre y julio de 2017 respectivamente). Requieren conocimientos más avanzados para su uso, al presentar interfaces basadas en lína de comandos. Se pueden extender con módulos adicionales.

Durante la segunda fase del análisis se seleccionaron BurpSuite Professional e IBM Security AppScan v9, ambas con versiones trial, basado en datos de implantación en Viewnext y nivel de cobertura y tests soportados por las versiones comerciales. Se analizó el mismo módulo del estudio de herramientas SAST. La prueba consistió en la elaboración automática del mapa de contenidos (crawl), y la realización del escaneado automático de vulnerabilidades (scan). Las dos herramientas arrojaron valores similares de detección de vulnerabilidades y ratio de falsos positivos [9].

VII. HERRAMIENTAS IAST (Y RASP)

Las herramientas de análisis interactivo de la seguridad de las aplicaciones, combinan el análisis dinámico de vulnerabilidades con la monitorización de la aplicación en tiempo real en el entorno de ejecución. Normalmente para ello realizan una instrumentalización de la aplicación mediante la utilizan de agentes que se ejecutan embebidos en la propia aplicación o incrustados en el entorno de ejecución. Cubren una de las principales carencias detectadas en las herramientas SAST y DAST [10], que es la posibilidad de evaluar el sistema en toda su dimensión (código fuente, entorno de ejecución, flujo de información, datos de configuración, peticiones e incluso librerías y frameworks utilizados), por lo que los informes de auditorías son más completos y arrojan menos falsos positivos. Entre este tipo de herramientas, hay un tipo especial, que son las aplicaciones de protección de la seguridad de las aplicaciones en tiempo real (Run-time Application Security Protection), que añaden adicionalmente la autoprotección de la aplicación frente a determinados ataques, respondiendo al mismo, por ejemplo, finalizando la sesión del atacante. En este caso más que herramientas para la realización de pruebas de seguridad, son herramientas de protección de las aplicaciones.

Dentro de este tipo de herramientas, se estudian:

- *Modsecurity*

Proporciona funcionalidades de generación de trazas de seguridad y adicionalmente realiza la monitorización del tráfico HTTP que llega a una aplicación para identificar patrones de ataque utilizando para ello un motor de reglas que incluye reglas preconfiguradas (OWASP ModSecurity Core Rule Set (CRS)) y la posibilidad de configurar nuestras propias reglas de validación y detección de ataques. *Modsecurity* permite su configuración en tres modos: *Modelo de Seguridad Negativa,* donde todo se permite el tráfico excepto el que se detecta sospechoso, en cuyo caso se incluye en las trazas de control o se rechaza. *Modelo de Seguridad Positiva,* donde todo el tráfico se bloquea por defecto excepto las peticiones que se sabe que son válidas (esto requiere un mayor conocimiento de la funcionalidad de la aplicación y que esta no sufra cambios frecuentes). *Modelo de Debilidades y Vulnerabilidades Conocidas,* que se apoya en la configuración de reglas antes mencionada, de manera que se pueda utilizar como mecanismo de contingencia (*virtual patching*) ante un ataque real reduciendo el tiempo de exposición mientras se subsana la vulnerabilidad.

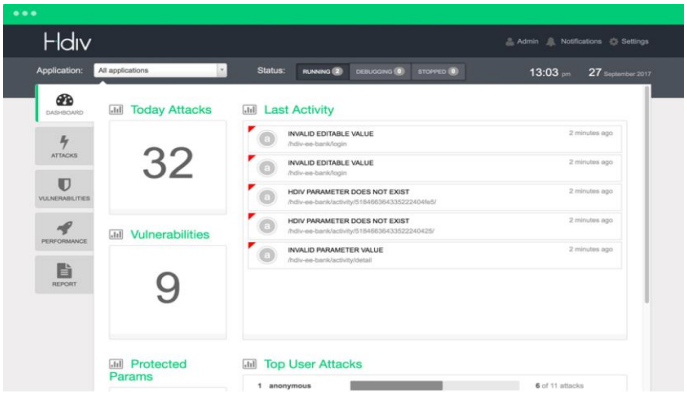


Fig. 8. Dashboard Hdiv

- *HDIV (IAST y RASP)*

Proporciona funcionalidad IAST y RASP en su versión *Enterprise* (dispone también de una versión *Community*). Como herramienta IAST detecta las vulnerabilidades de nuestro código en tiempo de ejecución. Facilita mayor información al respecto de los problemas encontrados con datos reales de ejecución. Explora vulnerabilidades también en librerías de terceros, trazándolas con CVE. Como herramienta RASP, proporciona la identificación de vulnerabilidades de seguridad y además la autoprotección ante fallos de diseño de negocio mediante validaciones de lista blanca en tiempo real y la generación de respuestas a dichos fallos. Integra cuadros de mando con información sobre vulnerabilidades y fallos o ataques detectados en tiempo real (figura 8). Además, permite integración con herramientas de ticketing y soluciones SIEM.

➢ *AppSensor*

Proyecto OWASP orientado a la definición de puntos de detección de ataques y los mecanismos de respuesta recomendados, de manera que se pueda limitar la ventana de oportunidad ante un ataque. En su versión actual, contiene más de 50 escenarios diferentes de detección y más de 12 acciones de respuesta. Esta información sirve como base para el desarrollo de soluciones IAST y RASP. Dentro del proyecto, se incluye una implementación de referencia.

Category Type	Description	Response ID	Titles
None	No response	ASR-P	No Response
Silent	User unaware of application's response	ASR-A	Logging Change
		ASR-B	Administrator Notification
		ASR-C	Other Notification
		ASR-N	Proxy
Passive	Changes to user experience but nothing denied	ASR-D	User Status Change
		ASR-E	User Notification
		ASR-F	Timing Change
Active	Application functionality reduced for user(s)	ASR-G	Process Terminated
		ASR-H	Function Amended
		ASR-I	Function Disabled
		ASR-J	Account Logout
		ASR-K	Account Lockout
		ASR-L	Application Disabled
Intrusive	User's environment altered	ASR-M	Collect Data from User

Response		Classifications								
		Purpose				Target User		Response Duration		
Code	Description	Logging	Notifying	Disrupting	Blocking	One	All	Instantaneous	Period	Permanent
ASR-A	Logging Change	●				●	○	○	○	
ASR-B	Administ'r Notification	●	●			●	●	●		
ASR-C	Other Notification	●	●			●		●		
ASR-D	User Status Change	●				●			●	
ASR-E	User Notification	●	●	●		●		●		
ASR-F	Timing Change	●		●		●	○	○	○	
ASR-G	Process Terminated	●	○	●		●		●		
ASR-H	Function Amended	●	○	●	●	●	○		●	○
ASR-I	Function Disabled	●	○	●	●	●	○		●	○
ASR-J	Account Logout	●	○	●	●	●		●		
ASR-K	Account Lockout	●	○	●	●	●			●	○
ASR-L	Application Disabled	●	○	●	●		●			●
ASR-M	Collect Data from User	●				●			●	
ASR-N	Proxy	●				●	○		●	○
ASP-P	No response									

□□□□ ● always, ○ sometimes

Fig. 9. AppSensor. Clasificación de respuestas a ataques

VIII. CONCLUSIONES FINALES

No existe una herramienta única que evalúe todos los riesgos y proteja un sistema software a lo largo de todas las fases de su ciclo de vida y desarrollo. Se recomienda combinar varios tipos en distintos momentos del SDLC [11]. Evaluaciones SAST integrados en sistemas IC, que automatizan auditorías de código para identificar anti-patrones, generando información en tiempo real.

La utilización combinada de herramientas DAST, permite integrar ciclos de escaneado dinámico y automático en la fase de testing continuo, así como validaciones manuales durante la fase de validación de releases. También facilitan la confirmación o descarte de falsos positivos detectados en el ciclo SAST. La utilización de herramientas IAST como evolución natural de los análisis DAST, permite añadir un nivel más de fiabilidad a los resultados de revisión pre-release, reduciendo el número de falsos positivos y permitiendo la instrumentalización de la aplicación en entornos de ejecución real. Si la solución integra capacidad de autoprotección y respuesta (RASP), se limita la ventana de exposición ante posibles atacantes.

Para aplicaciones orientadas a dispositivos móviles, es recomendable contar con herramientas de optimización, ofuscación y encriptación del código como *ProGuard*, para evitar o dificultar la aplicación de ingeniería inversa.

AGRADECIMIENTOS

Los autores agradecen la financiación de la Junta de Extremadura (Consejería de Economía, Comercio e Innovación), FEDER (Fondo Europeo de Desarrollo Regional) y Viewnext.

REFERENCIAS

[1] Félix de Sande, J.A. *"La evolución hacia un modelo preventivo"*. Blog 'The Next' (Viewnext). [Online]. Disponible: https://www.viewnext.com/la-evolucion-hacia-un-modelo-preventivo

[2] Romeo, Chris. *"A primer on secure DevOps: Why DevSecOps matters"*. TechBeacon. [Online]. Disponible: https://techbeacon.com/devsecops-foundations

[3] Sancho, J. C.; Caro, A.; García, P.; Félix, J.A.: "*Metodología de Implantación Empresarial de un Modelo de Desarrollo de Software Seguro*". Jornadas Nacionales de Investigación en Ciberseguridad - JNIC 2017. [Online]. Disponible: https://eciencia.urjc.es/bitstream/handle/10115/14540/Actas-JNIC-2017-repositorio.pdf

[4] Techbeacon - *"SAST, DAST, IAST, and RASP: Pros, cons and how to choose"*. TechBeacon. [Online]. Disponible: https://learn.techbeacon.com/units/sast-dast-iast-rasp-pros-cons-how-choose

[5] OWASP – *'Technical Risks of Reverse Engineering and Unauthorized Code Modification'*. [Online]. Disponible: https://www.owasp.org/index.php/Technical_Risks_of_Reverse_Engineering_and_Unauthorized_Code_Modification

[6] Kiuwan OWASP Benchmark results. [Online]. Disponible: https://www.kiuwan.com/blog/owasp-benchmark-kiuwan/

[7] SonarQube Java Plugin, OWASP Benchmark results. [Online]. Disponible: https://blog.sonarsource.com/sonarqube-enters-the-security-realm-and-makes-a-good-first-showing/

[8] Gartner Magic Quadrant for Application Security Testing. Security Intelligence Blog - IBM. [Online] Disponible: https://securityintelligence.com/ibm-retains-leadership-position-in-2017-gartner-magic-quadrant-for-application-security-testing/#.U812ERBx2Qw

[9] SecToolMarket – *"The WIVET Score of Web Application Scanners"* [Online]. Disponible: http://www.sectoolmarket.com/wivet-score-unified-list.html

[10] Robert Lemos – "Interactive application security testing: Ready for prime time?" TechBeacon. [Online]. Disponible: https://techbeacon.com/interactive-application-security-testing-it-ready-prime-time

[11] *"Application Security Cyber Risk Managed Services"*. Deloitte. [Online]. Disponible: https://www2.deloitte.com/content/dam/Deloitte/in/Documents/risk/in-cyber-risk-managed-services-application-security-noexp.pdf

Don't shoot the messenger

How a trusted channel may not be a necessary assumption for remote code-voting

Iñigo Querejeta-Azurmendi
Minsait
Universidad Carlos III Madrid
iquerejeta@minsait.com

Jorge López Hernández-Ardieta
Minsait
Universidad Carlos III Madrid
jlhardieta@minsait.com

Luis Hernández-Encinas
I. Tecnologías Físicas y de la Información
CSIC
luis@iec.csic.es

***Abstract*—Current literature, when migrating the idea of code-voting to remote electronic elections, makes one of the most naive trust assumptions: post mail, e-mail or sms are trusted channels. In this paper we propose an alternative view to code-voting to remove such an assumption. The presented idea maintains the most attractive properties of code-voting, mainly individual verifiability and usability towards the voter. Moreover, we present how such a construction can also be used to achieve coercion resistance in the re-voting setting with a trade-off in usability.**

***Index Terms*—Code-voting, e-Voting, Individual verifiability**

Tipo de contribución: Investigación en desarrollo

I. Introduction and Motivation

The internet voting (i-voting) setting has several differences compared with the case where the voter has to access a poll station and cast a vote in a voting booth. The first, and most relevant difference, is that in the former, the vote cast happens in an uncontrolled environment, meaning that we allow the voter to cast a vote in presence of an adversary, and therefore liberty of choice may be threatened. The second difference, and an implication of the first, is that vote cast happens from the voters device, meaning that the latter can be infected. Protecting liberty of choice in remote elections is usually managed by making it impossible for the voter to prove the direction of her vote. A more formal definition of such a concept is what is known as *coercion-resistance* [1]. To mitigate the problem of an untrusted voting device, whether it is a private PC or a machine in a poll station, *code-voting* is an elegant and user friendly solution.

Code-voting has been proposed both for face-to-face voting and for remote voting. In the former setting, each of the voters taking part in the election receives a code-sheet when they attend the poll station in order to cast a vote. This sheet consists of several columns containing voting codes, the candidates to vote for and verification codes (or acknowledge codes). These codes (both voting and verification) are unique per code-sheet. In order to cast a vote, instead of sending the candidate she wishes to vote for, the voter sends the voting-code related to the desired candidate, together with some information which relates this choice to its personal code-sheet. The verification process, while it differs depending on the protocol [2], [3], [4], [5], generally consists in showing the expected verification code to the voter. Once leaving the voting booth, the code-sheet (and therefore the relation among the codes) must be destroyed in order to prevent voters to prove which candidate they voted for.

However, when migrating this idea to remote voting, it can no longer be verified that the relation of codes has been destroyed, and therefore, privacy is endangered. In other words, the relation among codes and candidates is no longer private. Moreover, code-sheets are no longer assigned anonymously as they must be sent specifically to voters. This is another exposure of privacy and it is therefore necessary to assume trust in the transport channel [6], [7], [8]. This is a very strong assumption to make for which current literature has not presented a solution. Postal mail, sms, or e-mail are clearly not trusted channels, and assuming the opposite for a case as important as privacy in elections, is naive. So, why insist on using code-voting for the case of remote elections? The interesting property of code-voting which we find useful for the case of remote elections, is that it offers a comprehensible and straightforward verification progress, which impedes the voting device to cheat. In this document we present what we believe can be a promising research line that would remove this assumption in an elegant and intuitive way while maintaining usability towards the voter.

II. Removing the trust assumption

The direction to which our research is guided is to simply make the user (or more precisely, its voting device) choose the relation among vote codes, candidates and verification codes. This would maintain the verification properties in our system while improving some other properties additional to current code-voting schemes. Moreover, this would allow removing the assumed trust in the transport channel, as the code-sheet would no longer contain a relation among codes and candidates.

A. Idea motivating our research

Our construction is based on a very simple idea. We let the voter's device be the one that chooses the relation of codes with candidates by choosing a random permutation and casting the vote with the latter and the voting code. The verification process happens with the same matrix, which convinces the voter that the permutation was the one she intended. We hereby present a sketch of the protocol we are currently working on. We finish the section by pointing out the properties that such a construction would provide us.

Step 1: The standard order of the candidates, `CandOrd`, is published in a public, immutable database. Then, unique codes are randomly generated. To be exact, n vote codes and n verification codes per voter in the election roll, where n is the number of candidates. Finally, one code-sheet per voter is generated, where a voting-code and verification-code per candidate is included (making a total of $2n$ codes). We refer

to these codes as VoteCod and VerifCod respectively and can be thought of as arrays containing n integers. Each of these code-sheets is sent to the voters in the electoral roll.

Step 2: In the moment of voting, the voter's device calculates a random permutation, $\pi\colon \{1,2,\ldots,n\} \to \{1,2,\ldots,n\}$ and its square π^2. By an abuse of notation, π can be applied both to the whole set as to each of the individual elements depending on the considered input. To choose which are the vote and verification codes related to the desired candidate, it calculates $\pi(\texttt{VoteCod})$ and chooses the vote code $\texttt{VoteCod}[i]$ such that $\pi(\texttt{VoteCod}[i])$ equals the position of the desired candidate in CandOrd. The verification is the code in VerifCod in position $\pi^2(\texttt{VoteCod}[i])$.

Step 3: When casting a vote, the voter sends the encrypted permutation, in the shape of a permutation matrix, its square, a proof that these conditions hold and the vote code. The matrix is encrypted using a homomorphic encryption scheme, which allows the server to perform the following operation without ever decrypting the permutation matrix:

- Upon receipt of a vote, the server checks the position of $\texttt{VoteCod}[i]$, generates a binary vector with a single one in the relative position, and applies the encrypted permutation matrix.
- This results in an encrypted vector with a one in the position of the desired candidate, which allows the system to be used with a homomorphic tally scheme.
- Similarly the server performs this operation with the squared permutation matrix, which allows, once the election is closed, to determine the verification codes.

Such a protocol has several interesting properties. Firstly, the code-sheets do not contain compromising information, and therefore, an eavesdropper spying on the transport channel and the information contained in the code-sheet does not learn any information which can be used to determine the vote direction. Secondly, if the voting device is compromised and wants to cheat (and therefore change the vote intention) it can only do so in a probabilistic manner. In order to cheat, the voting device has to send a permutation matrix different to the one the voter agreed upon. As it has no knowledge of the order of the codes in the code-sheet, it can only randomly choose an alternate permutation and hope that the verification code will be the same. Moreover, if we assume the voting device never has access to the code sheet (it can be sent via sms or post mail), it cannot compromise vote secrecy, as only an attacker knowing both the order of the codes and the permutation can determine such information.

B. Mitigating Coercion

Finally, such a scheme can be used to mitigate coercion in a re-voting setting, which consists in allowing the voters to vote multiple times and therefore overwrite the coerced vote. Assume that the verification code chosen during coercion is $\texttt{VerifCod}[i]$. In order to avoid coercion, the voter selects a different permutation π', such that $\pi'^2(\texttt{VoteCod}[i]) = \pi^2(\texttt{VoteCod}[i])$, and therefore the verification process does not give away the coercion escape. Note that this calculation can only be performed by someone having knowledge of the vote and verification codes order, and therefore the machine can not selectively choose such a matrix. Such a raw explanation of this solution highly increases the complexity towards the voter, as she has to deal with permutations with certain properties. However, two things have to be considered here: firstly, it is a solution to coercion, which is a hard task for remote voting solutions in general and for the code-voting setting in particular. Secondly, this step will not be so roughly presented to the voter, as the interface can be designed in a manner that guides the voter in this task, and therefore makes it a more intuitive step.

III. Conclusions and future work

In what concerns a homomorphic encryption scheme to apply to the permutations and proving several relations between encrypted matrices needed in Step 2, there exist solutions in current literature [9], [10].

Determining which permutation to use, or how to construct the protocol is what we are currently working on, and as for now, we know what must not be done. On the one hand, the permutation π cannot simply be a rotation matrix, as if you know l and $\pi^2(l)$, then you learn $\pi(l)$, which in our case, would break vote privacy. Furthermore, we do not want a permutation, π, such that there exists a permutation π' with $\pi^2 = \pi'^2$. If this happens, then the voting device would be able to cheat with the voter 'verifying' correctly with a 100% chance of achievement.

The construction lacks a solution that prevents an adversary from obliging a voter to abstain during the election, or one for being able to verify that the verification code was not as expected. The former attack can be reproduced if the adversary has access to all verification codes, as it can verify whether one of these appears in the verification phase. For the case of the latter, our construction currently has no way to prove or verify whether a claim of incorrect verification is true or not. Our future work will aim to solve these problems, and once a convincing solution has been found, we will perform a complexity analysis.

References

[1] A. Juels, D. Catalano, and M. Jakobsson, "Coercion-resistant electronic elections," in *Proceedings of the 2005 ACM Workshop on Privacy in the Electronic Society*, ser. WPES '05, New York, NY, USA, pp. 61–70.

[2] A. Kiayias, T. Zacharias, and B. Zhang, "End-to-End Verifiable Elections in the Standard Model," in *EUROCRYPT*. Springer, 2015, pp. 468–498.

[3] D. Chaum, R. Carback, J. Clark, A. Essex, S. Popoveniuc, R. L. Rivest, P. Y. Ryan, E. Shen, and A. T. Sherman, "Scantegrity II: End-to-end Verifiability for Optical Scan Election Systems Using Invisible Ink Confirmation Codes," *Proceedings of the Conference on Electronic Voting Technology*, 2008.

[4] D. Chaum, "SureVote: Technical Overview," *Proceedings of the Workshop on Trustworthy Elections (WOTE '01)*, 2001.

[5] P. Y. A. Ryan, "PrêT à Voter with Confirmation Codes," in *Proceedings of the 2011 Conference on Electronic Voting Technology/Workshop on Trustworthy Elections*, ser. EVT/WOTE'11. Berkeley, CA, USA: USENIX Association, 2011, p. 10.

[6] P. Y. A. Ryan and V. Teague, "Pretty good democracy," in *WORKSHOP ON SECURITY PROTOCOLS*, 2009.

[7] R. Joaquim and C. Ribeiro, *CodeVoting Protection Against Automatic Vote Manipulation in an Uncontrolled Environment*. Berlin, Heidelberg: Springer, 2007, pp. 178–188.

[8] T. Truderung and A. Brelle, "Cast-as-intended mechanism with return codes based on PETs," in *Electronic Voting: Second International Joint Conference, E-Vote-ID, Bregenz, Austria, October 24-27, Proceedings*, 2017.

[9] R. Sunuwar and S. K. Samal, "Elgamal Encryption using Elliptic Curve Cryptography." 2015.

[10] J. Groth, "Linear Algebra with Sub-linear Zero-Knowledge Arguments." Springer, Berlin, Heidelberg, 2009, pp. 192–208.

Public Key Infrastructure Based on Multivariate Cryptography

Edgar González Fernández[1,2], Guillermo Morales-Luna[2], Feliú Sagols Troncoso[3], Luis Javier García Villalba[1]

[1]Group of Analysis, Security and Systems (GASS)
Department of Software Engineering and Artificial Intelligence (DISIA)
Faculty of Computer Science and Engineering, Office 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, Spain
Email: edggonza@ucm.es, javiergv@fdi.ucm.es

[2]Computer Science Department, CINVESTAV-IPN
Av. IPN 2508, San Pedro Zacatenco, 07300 Mexico City, Mexico
Email: egonzalez@computacion.cs.cinvestav.mx, gmorales@cs.cinvestav.mx

[3]Department of Mathematics, CINVESTAV-IPN
Av. IPN 2508, San Pedro Zacatenco, 07300 Mexico City, Mexico
Email: fsagols@math.cinvestav.edu.mx

***Abstract*—Financial transactions are performed, secured and authenticated through an almost blind trust in current technology and communication protocols depending on Public Key Cryptography. So far, cryptographic primitives deployed in real life systems have been resistant against classical cryptanalysis, but as research in the area of quantum computing advances, so does the fear of potential quantum attacks. For this reason several efforts are being made to provide and standardize secure primitives against quantum computers. We analyze some promising multivariate schemes and provide a test platform to produce and verify digital signatures encoded in ASN.1 syntax, a key element for developing inter-operability in Public Key Infrastructure (PKI).**

***Index Terms*—Information Security, Multivariate Cryptography, Public Key Infrastructure.**

Type of contribution: *Research in development*

I. Introduction

Most of the communications between financial, government and industrial entities are protected by PKI, a set of rules and techniques used primarily to provide security services via certification and key agreement processes. So far, the dominant algorithms for encryption and digital signatures, are based either in the Factorization Problem or the Discrete Logarithm Problem (DLP). These algorithms have been resistant against cryptanalytic attacks, and to guarantee inter-operability among different entities, several cryptographic primitives and communication protocols depending on them have been standardized. However, the advances in the field of quantum computing threatens them since these algorithms are vulnerable to quantum computer attacks based on Shor's algorithm [1].

It is unknown whether a general purpose quantum computer can be built or if it will soon be ready. However, several post-quantum public key cryptosystems have been proposed and are being considered by important institutions such as NIST to be part of the next generation of Public Key Cryptosystems. The main goal of this work is to integrate existing implementations and develop a practical tool to provide cryptographic primitives based on Multivariate Public Key Cryptography (MPKC). We introduce briefly the basic theory involving MPKC in section II. Later we discuss our current progress together with some details of our implementation in section III. Finally, a brief overview of the future work is found in IV.

II. Multivariate Cryptography

Multivariate Cryptography is based on the proved difficulty of solving a multivariate quadratic system, known as the $\mathcal{MQ}$ problem, which belongs to the class of NP-complete problems [2]. Besides, $\mathcal{MQ}$ is harder than many collision finding problems, in particular, it is quantum resistant. We denote by $\mathbb{F}_q$ the finite field of q elements, and consider positive integers m and n. Most of the PKC systems based on $\mathcal{MQ}$ consist of the following:

- an easily solvable quadratic map $Q : \mathbb{F}_q^n \to \mathbb{F}_q^m$,
- affine bijective maps $S : \mathbb{F}_q^n \to \mathbb{F}_q^n$, $T : \mathbb{F}_q^m \to \mathbb{F}_q^m$,
- a quadratic system P obtained as $P := T \circ Q \circ S$.

The system P is used as the public key and must be difficult to solve, while Q, S and T conform the private key. The main purpose of S and T is to hide the algebraic structure that makes Q easy to solve. An early attempt to develop a multivariate cryptosystem was published in [3], producing the *Matsumoto-Imai cryptosystem*. Since then, many other schemes have been developed by changing the way the central map Q is generated, being the most promising variations of the *Hidden Field Equations* (HFE) [4] and *Rainbow* [5] constructions.

In addition to quantum resistance, multivariate cryptosystems have attractive characteristics that makes them a suitable option for its implementation, e.g., fast computation, short signatures, diversity of constructions and natural resistance against timing attacks. On the other hand, many of the aforementioned schemes have shown vulnerabilities against cryptographic analysis. Even more, these cryptosystems produce very big keys compared to ECDSA and RSA. Though this is not a big restriction for devices with high memory capabilities, as servers, personal computers or mobile devices, it could be difficult to store keys in limited memory devices used in financial transactions, such as smart cards.

III. Implementation of Algorithms Based on $\mathcal{MQ}$

We have implemented the `mqcrypto` command-line application (available online: https://github.com/cinvestavcs-mexicocity/MQCrypto) to provide cryptographic primitives based on $\mathcal{MQ}$ and we expect to produce software to provide the full capabilities of PKI as a completion of the current work. The algorithms provided are a compilation of multivariate algorithms that have resisted cryptanalysis and are being considered by NIST to be part of the post-quantum public key standards.

A. Supported Algorithms

Up to now, we have integrated the following algorithms: Rainbow5640, Rainbow6440, Rainbow16242020, Rainbow256181212, Pflash, 3ICP, TTS6440 whose implementations and specifications can be found on https://bench.cr.yp.to/ebats.html. Additionally, we have implemented Sflash v1, Sflash v2 and UOV using the open-source software `SageMath`, available on http://www.sagemath.org/. This software is required if these algorithms are to be used.

The main cryptographic functions provided by the `mqcrypto` tool are described next:

- **Key generation**. Key pairs are created at random following the construction explained in section II. The user is asked for a password to generate an AES ciphering key which will be used to store the private key. Public and private keys are stored separately and the private one is never delivered in clear.
- **Signing**. The signature utility has as input a document, file or text, the ciphered private key, and the password used in the generation process. Also a digest method can be selected from Sha-256 and Sha-512. If no method is provided, Sha-256 is used by default. The output is either printed on the terminal or stored in a file if a path is provided.
- **Verifying**. This command verifies a signature using the corresponding public key. The inputs for this process are the signature, the public key, and the document or text that has been signed. Once it completes the execution, the command returns a positive answer if the sign is authentic and a negative one in other case.

In the following example the input file.txt will be signed and the signature will be stored in the file.sgn.

```
> mqcrypto sign Rainbow5640Private.pem -in file.txt
  -out file.sgn -sha256 -passin pass:password
```

The output is a Base64 string encoding an ASN.1 structured object. Codification of the key pair is exhibited with more detail in subsection III-B.

For a more detailed presentation of this software, we invite the reader to look at [6] and the additional documentation found on the Github page of the project.

B. Codification of Signatures and Keys

Resulting key pairs and message signatures follow an ASN.1 encoding. A public key consists of a set of m polynomials in n variables over a finite field. This information is encoded as follows

```
MPKCPublicKey ::= SEQUENCE {
    variables INTEGER, -- n
    polynomials INTEGER, -- m
    fieldchar INTEGER, -- p
    fielddegree INTEGER, -- k
    polynomialSet OCTET STRING, -- P}.
```

However, the values of n, m, p and k can be defined previously as domain parameters known to every entity involved in the authentication process. In this case, only the field `polynomialSet` must be provided. A private key is encoded as follows

```
MPKCPrivateKey ::= SEQUENCE {
    affine1 INTEGER, -- S
    affine2 INTEGER, -- T
    polynomialSet OCTET STRING, -- Q}.
```

In general, key length increases dramatically as m and n increase, since the growth of the number of coefficients per polynomial is quadratic. Nevertheless, some algorithms admit short codification for the private key by using numeric parameters. This is the case for schemes based on Matsumoto and Imai, where the parameter `theta` is sufficient to recover the private polynomial set.

```
MIPrivateKey ::= SEQUENCE {
    affine1 INTEGER, -- S
    affine2 INTEGER, -- T
    exponent INTEGER, --  theta}
```

IV. Future Work

We are working on the integration of our platform with software performing certificate operations to provide a complete set of PKI tools based on multivariate cryptography. For this purpose, it is also of great importance to the main ISO standards dealing with this matter. We are also working on implementations of Zero-Knowledge Protocols based on the $\mathcal{MQ}$ problem, for instance those found in [7], [8], to provide a broader set of services.

Acknowledgement

Edgar González Fernández, Guillermo Morales-Luna and Feliú Sagols Troncoso acknowledge the partial support of Mexican Consejo Nacional de Ciencia y Tecnología (Conacyt).

References

[1] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997.

[2] E. Sakalauskas, "The multivariate quadratic power problem over $\mathbb{Z}_n$ is NP-complete," *Information Technology and Control*, vol. 4, no. 1, pp. 33–39, 2012.

[3] T. Matsumoto and H. Imai, "Public quadratic polynomial-tuples for efficient signature-verification and message-encryption," in *Proceedings of the Advances in Cryptology*. Davos, Switzerland: Springer, 1988, pp. 419–453.

[4] J. Patarin, "Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms," in *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques*, Zaragoza, Spain, 1996, pp. 33–48.

[5] J. Ding and D. Schmidt, "Rainbow, a new multivariable polynomial signature scheme," in *Proceedings of the International Conference on Applied Cryptography and Network Security*, New York, USA, 2005, pp. 164–175.

[6] E. Pérez Villegas, "Plataforma de experimentación criptográfica basada en Geometría Algebraica," Master's thesis, Computer Science Department, CINVESTAV-IPN, Mexico City, 2017.

[7] V. Nachef, J. Patarin, and E. Volte, "Zero-knowledge for multivariate polynomials," in *Proceedings of the Progress in Cryptology*, Santiago, Chile, 2012, pp. 194–213.

[8] E. González Fernández, G. Morales-Luna, and F. Sagols Troncoso, "Procedimientos de autenticación de conocimiento nulo mediante técnicas de geometría algebraica," in *Actas de las Segundas Jornadas Nacionales de Investigación en Ciberseguridad*, Granada, Spain, 2016, pp. 96–102.

A summary of: Encryption by Heart (EbH) - Using ECG for time-invariant symmetric key generation

L. González-Manzano[a*], J. M. de Fuentes[a], P. Peris-Lopez[a] , C. Camara[a]
[a]Computer Security Lab (COSEC). Universidad Carlos III de Madrid
[a]{lgmanzan, jfuentes,pperis,macamara}@inf.uc3m.es

***Abstract*- Wearable devices are a part of Internet-of-Things (IoT) that may offer valuable data of their porting user. This paper explores the use of ElectroCardioGram (ECG) records to encrypt user data. Previous attempts have shown that ECG can be taken as a basis for key generation but they do not consider time-invariant keys. This paper addresses this challenge by proposing EbH, a mechanism for persistent key generation based on ECG. Experimental results over 24 hours for 199 users show that EbH, under certain settings, can produce permanent seeds (thus time-invariant keys) computed on-the-fly and different for each user ---up to 95.97% of users produce unique keys.**

Tipo de contribución: *Investigación publicada*

I. INTRODUCTION

Nowadays, wearable devices are a growing trend [1]. The widespread adoption of wearable devices and Implantable Medical Devices (IMDs) make biosignals to be available for different purposes. However, a critical remark is that biological signals must be properly secured.

Beyond the protection of the biosignals themselves, the focus of this paper is on their use to protect user personal information. Thus, using a portable device, e.g. a smartphone, users can store their data in a secure way. Encryption mechanisms are typically applied for this purpose. We apply a symmetric encryption algorithm (a single key for encryption and decryption) in which the key is derived from the ECG signal of the user. Previous approaches have already pointed out that ECG signals can be used to generate encryption keys that change over time [2]. Other authors have explored how different IMDs or body sensors may derive a shared key leveraging on their independently perceived ECG signal [3].

This paper tries to answer these questions: Is it possible to create encryption/decryption keys using the user's ECG signal? Is each created key different between users? Does each key remain invariant along the time? Is each key difficult to reproduce?

To address these questions, two main contributions are presented. Firstly, we present Encryption by Heart (EbH), a mechanism to create time-invariant symmetric encryption keys derived from ECG and secondly, the implementation of EbH, which is made freely available to encourage further research in this direction.

The structure of the paper is the following: Section II defines the proposal. Section III presents the evaluation and Section IV concludes the paper.

II. DESCRIPTION OF EBH

EbH requires the user wearing a smart device that provides ECG values. This device is wirelessly connected to a hub (e.g. smartphone), which contains the data to be protected. The overview is presented in Fig. 1.

Before protecting users' data, the ECG user model ($ECG_{Mod_{ui}}$) is built. For this purpose, the smart device provides a set of ECG values (ECG reference data $ECG_{Ref_{ui}}$) which are processed by the hub device. Then, at any time T_1 when the user wants to encrypt certain data, the hub gathers a set of ECG samples $ECG^{(obs(T_1))}{}_{ui}=\{ECG^{(T_{1\text{-}Lo})}{}_{ui}, ECG^{(T_{1\text{-}(Lo+1)})}{}_{ui},\ldots,ECG^{(T_1)}{}_{ui}\}$ where L_o is the amount of time the user is observed prior to deriving each seed.

The seed $S^{T1}{}_{ui}$ is derived by the hub from both $ECG_{Mod_{ui}}$ and its similarity to $ECG^{(obs(T_1))}{}_{ui}$. In a nutshell, $S^{T1}{}_{ui}$ is composed by a set of features, each one being the result of the similarity between the corresponding features of $ECG_{Mod_{ui}}$ and $ECG^{(obs(T\{*\}))}{}_{ui}$ (and getting values between {0, 1, -1}). For the sake of robustness, this comparison is only performed in those features of $ECG_{Mod_{ui}}$ that convey enough cardiac information. For the cases in which it does not happen, the corresponding feature $s^{T1}{}_{ui}(k)$ of $S^{T1}{}_{ui}$ is given a *neutral* value.

After obtaining the seed $S^{T1}{}_{ui}$, it is taken as input to a random process (e.g. a Pseudo-Random Number Generator (PRNG) [4]) to generate the encryption key K_{ui}. The design of such a key derivation function is outside the scope of EbH.

Once the user, at any time T_2, wants to decrypt previously encrypted data, the key generation process is the same as the one described. The main difference is that current ECG values ($ECG^{(obs(T_2))}{}_{ui}$) are taken for this calculation. As it is a symmetric decryption, it is only successful if the decryption key is the same as the encryption one.

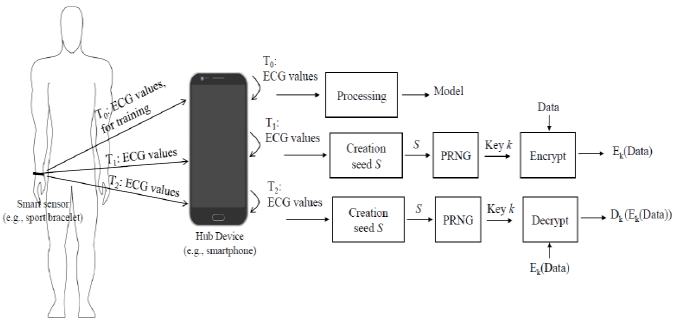


Fig 1. EbH overview

III. EVALUATION

Five parameters are involved in the evaluation; samples that compose $ECG_{Ref_{ui}}$; L_a which determines how much time each sample $ECG^{T^*}{}_{ui}$ represents; L_o; discard threshold *DT* used to discard features whose value is considered significantly small and thus unstable to be involved in the seed generation process; and tolerance margin *TM* used to consider that when $s^{T^*}{}_{ui}(k)$ is not *neutral*, a given feature $ECG_{Mod_{ui}}(k)$ and $ECG^{(obs(T^*))}{}_{ui}(k)$ are close enough.

Experiments have been carried out using dataset E-HOL-03-0202-003 provided by University of Rochester. It includes long-term recordings (around 24 hours) of 199 subjects.

Our prototype implementation of EBH is freely available[i].

A. Time-invariance and uniqueness

These two goals are achieved when each user generates the same seed over time and when it is different from that of other users. In general, the system outperforms well for a training set of 60%. The higher *DT*, the higher the amount of seeds. But *DT* and L_o do not have a consistent effect.

Time-invariance is significantly achieved. In particular, when $L_o = 10 \cdot L_a$, *DT*= 0.03, *TM* = 0.01 and 60 % of samples are taken to build $ECG_{Mod_{ui}}$, 208 seeds are produced for the 199 users. This implies 1.04 keys per user on average.

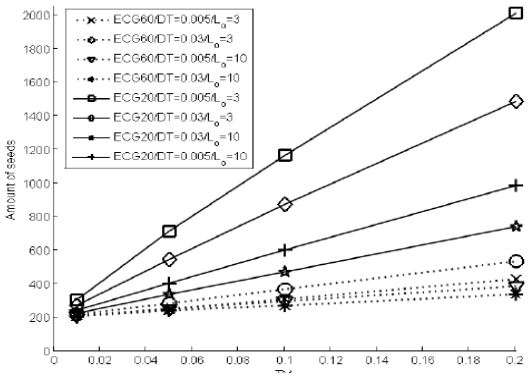


Fig 2. Amount of seeds $S^{T^*}{}_{ui}$ depending on *TM*. Series correspond to different combinations of the amount of samples in $ECG_{Mod\{ui\}}$ (ECG60 and ECG20 refer to 60% and 20% training set respectively), the value of L_o (in multiples of L_a) and *DT*.

With respect to distinctiveness, it is achieved if all seeds are different. Our results show that most configurations lead to unique seeds per user. However, a small amount of settings (only when *DT* = 0.005 for $ECG_{Mod_{ui}}$ with 60% of samples) lead to seeds that are shared by at least two users.

B. Invulnerability

The invulnerability of the generated seeds is given by their size and randomness. Concerning the size, see Fig. 3, measured in terms of the amount of symbols (2 bits per symbol as there are {0,1,-1,*neutral*} values), seeds range from 115 symbols to 175. Interestingly, as expected, having *DT* = 0.03 also causes that under that percentage the seed is around 118 symbols.

With respect to unpredictability, see Fig. 4, all parameters have some effect, being *DT* the one with higher impact. Surprisingly, having higher values of *DT* cause achieving better min-entropy. The reason behind is that *DT* removes some features that are consistently having the same value in the resulting seeds, thus becoming predictable. Moreover, min-entropy benefits from having smaller training sets.

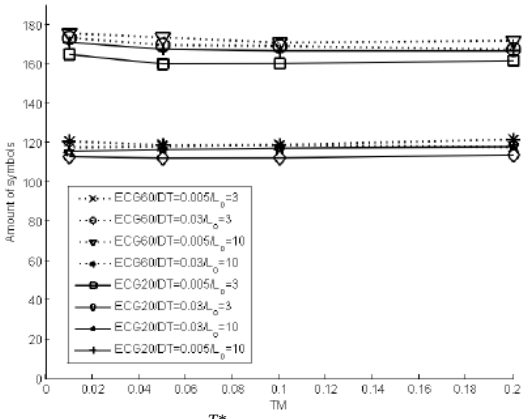


Fig 3. Amount of seeds $S^{T^*}{}_{ui}$ (in number of symbols) depending on *TM*.

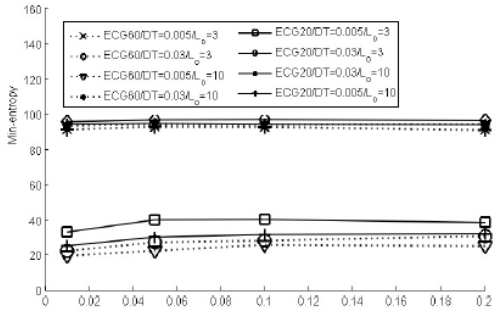


Fig 4. Min_entropy per seed $S^{T^*}{}_{ui}$ depending on *TM*.

IV. CONCLUSIONS AND FUTURE WORK

ECG biosignals have been applied in the crytographic area. Previous efforts have explored its application to protect smarthealth data or authenticate users. However, this paper presents EbH, a mechanism to derive time-invariant encryption keys. An extended version of this paper is in [5].

Future research works will be mainly focused on expanding this approach to other biosignals (such as PPG) and exploring other settings that may allow smaller training times.

ACKNOWLEDGEMENT

This work was supported by the MINECO grants TIN2013-46469-R (SPINY) and TIN2016-79095-C2-2-R (SMOG-DEV); by the CAM grant S2013/ICE-3095 (CIBERDINE)

REFERENCES

[1] H. Liu, H. Ning, Y. Yue, Y. Wan, L. T. Yang (2017), Selective disclosure and yoking-proof based privacy-preserving authentication scheme for cloud assisted wearable devices, Fut. Gen. Comp. Sys.

[2] Okoh, E., & Awad, A. I. (2015). Biometrics applications in e-health security: A preliminary survey. In HIS (pp. 92-103). Springer, Cham.

[3] Rostami, M., Juels, A., & Koushanfar, F. (2013). Heart-to-heart (H2H): authentication for implanted medical devices. In Proceedings of the 2013 ACM SIGSAC conference (pp. 1099-1112). ACM.

[4] Schneier, B. (2007). Applied cryptography: protocols, algorithms, and source code in C. john wiley & sons.

[5] González-Manzano, L., de Fuentes, J. M., Peris-Lopez, P., & Camara, C. (2017). Encryption by Heart (EbH)—Using ECG for time-invariant symmetric key generation. Fut. Gen. Comp. Sys., 77, 136-148.

[i] https://github.com/jmdefuentes/EbH

A summary of: "Heartbeats Do Not Make Good Pseudo-Random Number Generators"

Lara Ortiz-Martin
U. Carlos III de Madrid,
Leganés, Spain;
laortizm@inf.uc3m.es

Pablo Picazo-Sanchez
Chalmers University
Gothenburg, Sweden;
pablop@chalmers.se

Pedro Peris-Lopez
U. Carlos III de Madrid,
Leganés, Spain;
pperis@inf.uc3m.es

Juan Tapiador
U. Carlos III de Madrid,
Leganés, Spain;
jestevez@inf.uc3m.es

***Abstract*—The proliferation of wearable and implantable medical devices has given rise to an interest in developing security schemes suitable for these systems and the environment in which they operate. One area that has received much attention lately is the use of (human) biological signals as the basis for biometric authentication, identification and the generation of cryptographic keys. The heart signal (*e.g.*, as recorded in an electrocardiogram) has been used by several researchers in the last years. Specifically, the so-called *Inter-Pulse Intervals (IPIs)*, which is the time between two consecutive heartbeats, have been repeatedly pointed out as a potentially good source of entropy and are at the core of various recent authentication protocols. In this work, we report the results of a large-scale statistical study to determine whether such an assumption is (or not) upheld. For this, we have analyzed 19 public datasets of heart signals from the Physionet repository, spanning electrocardiograms from 1,353 subjects sampled at different frequencies and with lengths that vary between a few minutes and several hours. We have then applied a standard battery of randomness tests to the extracted IPIs and after analyzing these 19 public ECG datasets, our results raise doubts about the use of IPI values as a good source of randomness for cryptographic purposes. This has repercussions both in the security of some of the protocols proposed up to now, and also in the design of future IPI-based schemes.**

***Index Terms*—Randomness; Authentication; Privacy; Inter-Pulse Intervals; Biometric**

Tipo de contribución: *Investigación ya publicada (límite 2 páginas)*

I. Overview

Biometrics refer to identification and authentication methods that, using biological signals, can identify or validate the identity of a person. Biometrics have also been used to generate personal cryptographic keys [1] by using biological signals as a *Pseudorandom Number Generators (PRNG)*. Therefore, in order to check if a given sequence of numbers can be considered random, there are some well known tests like Shannon's entropy, Monte Carlo test or frequency test among others. However, instead of using a subset of tests, there are some public suites like ENT, DIEHARD or *National Institute of Standard and Technology Statistical Test Suite (NIST STS)*, which are commonly used for security applications.

In the last years, entropy analysis has demonstrated that *Electrocardiogram (ECG)* signals can be used as a source of entropy for security purposes [2], [3]. Many authors have claimed that the *Least Significant Bits (LSBs)* of the *Inter-Pulse Interval (IPI)* contain a high degree of entropy [4], [5], [6], [7]. In particular, this is done by calculating the IPI, which is the time-interval between two consecutive R-peaks of the ECG as can be seen in Figure 1. To provide grounds for this claim, most authors use some public databases to prove this entropy property. Thus, with this method, the resulting bits can be considered as random numbers and can be part of, for example, key generation protocols in authentication procedures.

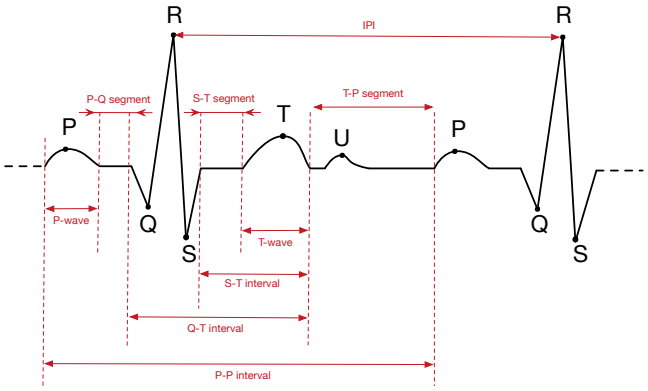


Fig. 1: A typical electrocardiogram (ECG) signal and its main features: peaks (P, Q, R, S, T, U), waves, segments and intervals.

Recent IPI-based authentication, identification, and key generation protocols (*e.g.*, [5], [6], [7], [8]) suffer from two main weaknesses. First, they only use measures of entropy to determine whether the generated cryptographic material (keys and other intermediate values such as nonces) are random or not. Second, the datasets used in these works are rather small and, therefore, possibly not significant enough. Additionally, such datasets contain ECG signals obtained both from healthy subjects and others that suffer some heart-related pathology, and it is unclear whether this feature has some influence on the overall quality (*i.e.*, randomness) of the derived bits. Some of these observations have been already raised in [9], in which authors pointed out the need to perform a more sound assessment of the quality of the generated keys using larger datasets and additional randomness tests. Nevertheless, the code which authors run these experiments is not available.

In [10], we overcome these weaknesses by performing an analysis of the randomness of 19 different public databases containing a substantial amount of heart signals. Our contributions can be summarized as:

Dataset	ENT	NIST STS	Avg. No. Samples	Median (IPI)	#Records	Frequency	Pathology
cebsdb	66.6%	93.3%	4,968,780	175	54	5,000	Healthy volunteers
ptbdb	66.6%	86.6%	108,818	68	545	1,000	Myocardial problems and Healthy controls
twadb	66.6%	86.6%	59,770	87	5	500	Myocardial problems
iafdb	66.6%	80.0%	19,707,034	37	5	1,000	Atrial fibrillation or flutter
cdb	66.6%	73.3%	5,120	12	53	250	Holter recordings
nstdb	66.6%	66.6%	650,000	1246	14	360	Physically active volunteers
mitdb	66.6%	60.0%	650,000	1113	46	360	Arrhythmia
qtdb	66.6%	60.0%	224,999	520.5	104	250	Holter recordings
stdb	66.6%	46.6%	624,166	1243	28	360	Stress tests
cudb	50.0%	40.0%	127,232	415	9	250	Ventricular problems
aami-ec13	66.6%	33.3%	55,522	48.5	10	720	Tachycardia
svdb	66.6%	33.3%	230,400	1192	47	128	Partial epilepsy
vfdb	50.0%	26.6%	525,000	1800	17	250	Tachycardia
szdb	83.3%	26.6%	17,245,701	4439	7	200	Partial epilepsy
slpdb	83.3%	26.6%	4,188,530	11517	17	250	Sleep Apnoea syndrome
edb	83.3%	26.6%	1,800,000	4405	90	250	Myocardial and hypertension
mghdb	66.6%	20.0%	1,479,358	2426	202	360	Unstable patients in critical care units
apnea-ecg	66.6%	20.0%	11,930	15786	77	100	Tachycardia
shareedb	83.3%	13.3%	10,553,116	46910	23	128	Hypertension

TABLE I: Characteristics vs. success rate datasets.

- We have downloaded 19 public databases with information about heart signals from different people. All datasets are taken from the Physionet repository [11]. We then extracted the last four bits of the IPI of each person per database, thus creating a bit stream whose quality can be tested.
- We analyzed all files independently to check if the ECG can be considered a good random number generator. To do so, two random number suites (ENT–general purpose–and NIST STS –security) have been run over all previously generated files. To the best of our knowledge, this is the first work that discusses how the ECG signal should be used in cryptographic protocols as a source of random numbers. Our tests are made public[1] to facilitate the replication of our results.
- An analysis of Table I, where the average of our results are shown, we extract the following conclusions:
  - When the number of IPIs (in median) is higher than 1800, the signals achieve extremely poor results (2 passed tests out of 15 in the worst case) in the NIST STS. Examples of these databases are `vfdb`, `szdb`, `slpdb`, `mghdb`, `edb`, `apnea-ecg` and `shareedb`.
  - When the number of IPIs (in median) is between 1800 and 415, then signals are in the borderline of passing (at least) half of the NIST STS. Examples of these databases are `svdb`, `cudb`, `stdb`, `qtdb`, `mitdb` and `nstdb`. There is also one exception to this rule: `aami-ec13` which has 48.5 IPIs in median and it achieves a 33.3% (5 passed tests out of 15) which is similar to `svdb` results.
  - When the number of IPIs (in median) is between 415 and 37, the databases achieve extremely good results (14 passed tests out of 15 in the best case) in the NIST STS. Examples of these databases are `cdb`, `twadb`, `pbdb`, `iafdb`, `cebsdb`. As before, there is an exception to this rule: `aami-ec13` which has 48.5 IPIs in median and it only passes 5 out of 15 tests.
- The results obtained in our analysis suggest two main conclusions: 1) a short burst of bits derived from an ECG record may seem random; but 2) large files derived from long ECG records should not be used for security purposes (*e.g.*, key generation algorithms). These conclusions should be taken with caution since they are conditioned to: 1) a particular IPI extraction algorithm; and 2) the 19 public databases studied.

[1] https://github.com/aylara/Random_ECG

ACKNOWLEDGMENTS

This work was supported by the CAM grant S2013/ICE-3095 (CIBERDINE: Cybersecurity, Data, and Risks), by the MINECO grant TIN2016-79095-C2-2-R (SMOG-DEV — Security mechanisms for fog computing: advanced security for devices) and by the Swedish Research Council (Vetenskapsrådet) under grant Nr. 2015-04154 (PolUser: Rich User-Controlled Privacy Policies).

REFERENCES

[1] L. Yao, B. Liu, G. Wu, K. Yao, and J. Wang, "A biometric key establishment protocol for body area networks," *International Journal of Distributed Sensor Networks*, vol. 2011, 2011.

[2] R. Altawy and A. M. Youssef, "Security tradeoffs in cyber physical systems: A case study survey on implantable medical devices," *IEEE Access*, vol. 4, pp. 959–979, 2016.

[3] G. Zheng, G. Fang, R. Shankaran, and M. A. Orgun, "Encryption for implantable medical devices using modified one-time pads," *IEEE Access*, vol. 3, pp. 825–836, 2015.

[4] R. M. Seepers, C. Strydis, P. Peris-Lopez, I. Sourdis, and C. I. De Zeeuw, "Peak misdetection in heart-beat-based security: Characterization and tolerance," in *EMBC*. IEEE, 2014, pp. 5401–5405.

[5] I. Vasyltsov and S. Lee, "Entropy extraction from bio-signals in healthcare iot," in *IoTPTS*. ACM, 2015, pp. 11–17.

[6] R. M. Seepers, C. Strydis, I. Sourdis, and C. D. Zeeuw, "Enhancing heart-beat-based security for mhealth applications," *IEEE Journal of Biomedical and Health Informatics*, vol. PP, no. 99, pp. 1–1, 2015.

[7] M. Rostami, A. Juels, and F. Koushanfar, "Heart-to-heart (h2h): authentication for implanted medical devices," in *CCS*, ser. CCS '13. ACM, 2013, pp. 1099–1112.

[8] D. K. Altop, A. Levi, and V. Tuzcu, "Deriving cryptographic keys from physiological signals," *Pervasive and Mobile Computing*, pp. –, 2016.

[9] L. E. Bassham, III, A. L. Rukhin, J. Soto, J. R. Nechvatal, M. E. Smid, E. B. Barker, S. D. Leigh, M. Levenson, M. Vangel, D. L. Banks, N. A. Heckert, J. F. Dray, and S. Vo, "Sp 800-22 rev. 1a. a statistical test suite for random and pseudorandom number generators for cryptographic applications," National Institute of Standards & Technology, Tech. Rep., 2010.

[10] L. Ortiz-Martin, P. Picazo-Sanchez, P. Peris-Lopez, and J. Tapiador, "Heartbeats do not make good pseudo-random number generators: An analysis of the randomness of inter-pulse intervals," *Entropy*, vol. 20, no. 2, 2018.

[11] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13).

Maude-NPA version 3.1

Santiago Escobar
Universitat Politècnica de València, Spain
sescobar@dsic.upv.es

Catherine Meadows
NRL, Washington, DC, USA
meadows@itd.nrl.navy.mil

Jose Meseguer
University of Illinois at Urbana-Champaign, USA
meseguer@illinois.edu

I. Introduction

Maude-NPA is an analysis tool for cryptographic protocols that takes into account many of the algebraic properties of cryptosystems that are not supported in other tools.

Maude-NPA uses an approach similar to its predecessor, the NRL Protocol Analyzer (NPA) [6], i.e., it is based on unification and performs backwards search from a final state to determine whether or not it is reachable. However, unlike the original NPA, it has a theoretical basis in *rewriting logic* and *narrowing* [7], and while NPA only could be used to reason equational theories involving a fixed set of rewrite rules, Maude-NPA can be used to reason about a wide range of cryptographic properties.

A description of Maude-NPA's formal foundations in rewriting logic, together with a soundness and completeness proof, are given in [1]. The most detailed description of how Maude-NPA works is given in [3].

The current version 3.1 [2] of Maude-NPA adds several useful new features to version 2.0, including:

1. *Equational Variant-based Unification in Full Generality*. Unification modulo a theory has been extended to its full generality for theories satisfying the *finite variant property* [4] when the equational theory is convergent modulo any combination of associativity and/or commutativity and/or identity axioms.
2. *Protocol Composition*. Protocols are often obtained by combining several sub-protocols. Such subprotocols can now be specified modularly, and the security properties of their compositions can be analyzed by the tool.
3. *Process Algebra*. In addition to the strand space notation [5] traditionally used by Maude-NPA, a more convenient notation based on process algebra allows easy specification of protocols with branching behavior and various kinds of choice and non-determinism.

II. The Maude-NPA syntax

Given a protocol $\mathcal{P}$ to be specified, protocol states are modeled as elements of an initial algebra $\mathcal{T}_{\Sigma_\mathcal{P}/E_\mathcal{P}}$, i.e. each state is an equivalence class $[t]_{E_\mathcal{P}} \in T_{\Sigma_\mathcal{P}/E_\mathcal{P}}$ where $\Sigma_\mathcal{P}$ is the set of symbols defining the protocol $\mathcal{P}$, and $E_\mathcal{P}$ specifies the *algebraic properties* of the cryptographic functions $\Sigma_\mathcal{P}$. The cryptographic properties $E_\mathcal{P}$ may vary depending on different protocols.

The signature $\Sigma_\mathcal{P}$ incorporates some predefined symbols for protocol infrastructure. A state is a term of the form $\{S_1 \,\&\, \cdots \,\&\, S_n \,\&\, \{IK\}\}$ where $\&$ is an associative-commutative union operator with identity symbol $\emptyset$.

The *intruder knowledge* $\{IK\}$ belongs to the state and is represented as a set of facts using the comma as an associative-commutative union operator with identity element $empty$. There are two kinds of intruder facts: *positive* knowledge facts (the intruder knows m, i.e., $m{\in}\mathcal{I}$), and *negative* knowledge facts (the intruder *does not yet know* m but *will know it in a future state*, i.e., $m{\notin}\mathcal{I}$), where m is a message expression.

Each S_i specifies the sequence of messages sent and received by a principal executing the protocol. *Strands* [5] are represented as a sequence of messages $[msg_1^\pm, msg_2^\pm, msg_3^\pm, \ldots, msg_{k-1}^\pm, msg_k^\pm]$ with $msg_i^\pm$ either msg_i^- (also written $-msg_i$) representing an input message, or msg_i^+ (also written $+msg_i$) representing an output message. Note that each msg_i is a term of a special sort Msg; this sort is extended by the user to allow any user-definable protocol syntax. Variables of a special sort Fresh are used to represent pseudo-random values (nonces) and Maude-NPA ensures that two distinct fresh variables will never be merged. Strands are extended with all the fresh variables created by that strand, i.e., $:: f_1, \ldots, f_k :: [msg_1^\pm, msg_2^\pm, \ldots, msg_k^\pm]$.

III. The Maude-NPA Execution Model

Strands are used to represent both the actions of honest principals (with a strand specified for each protocol role) and the actions of an intruder (with a strand for each action an intruder is able to perform on messages). In Maude-NPA strands evolve over time; the symbol | is used to divide past and future. That is, given a strand $[\ msg_1^\pm, \ \ldots, \ msg_i^\pm \mid msg_{i+1}^\pm, \ \ldots, \ msg_k^\pm\ ]$, messages $msg_1^\pm, \ldots, msg_i^\pm$ are the *past messages*, and messages $msg_{i+1}^\pm, \ldots, msg_k^\pm$ are the *future messages* ($msg_{i+1}^\pm$ is the immediate future message). A strand $[msg_1^\pm, \ldots, msg_k^\pm]$ is shorthand for $[nil \mid msg_1^\pm, \ldots, msg_k^\pm, nil]$. An *initial state* is a state where the bar is at the beginning for all strands in the state, and the intruder knowledge has no fact of the form $m{\in}\mathcal{I}$. A *final state* is a state where the bar is at the end for all strands in the state and there is no intruder fact of the form $m{\notin}\mathcal{I}$.

Since the number of states $T_{\Sigma_\mathcal{P}/E_\mathcal{P}}$ is in general infinite, rather than exploring concrete protocol states $[t]_{E_\mathcal{P}} \in T_{\Sigma_\mathcal{P}/E_\mathcal{P}}$ Maude-NPA explores *symbolic strand state patterns* $[t(x_1, \ldots, x_n)]_{E_\mathcal{P}} \in T_{\Sigma_\mathcal{P}/E_\mathcal{P}}(\mathcal{X})$ on the free $(\Sigma_\mathcal{P}, E_\mathcal{P})$-algebra over a set of variables $\mathcal{X}$. In this way, a state pattern $[t(x_1, \ldots, x_n)]_{E_\mathcal{P}}$ represents not a single concrete state but a possibly infinite set of such states, namely all the *instances* of the pattern $[t(x_1, \ldots, x_n)]_{E_\mathcal{P}}$ where the variables $x_1, \ldots, x_n$ have been instantiated by concrete ground terms.

The semantics of Maude-NPA is expressed in terms of *rewrite rules* that describe how a protocol moves from one state to another via the intruder's interaction with it. One uses Maude-NPA to find an attack by specifying an insecure state pattern called an *attack pattern*. Maude-NPA attempts to find a path from an initial state to the attack pattern via *backwards narrowing* (narrowing using the rewrite rules with

the orientation reversed). That is, a narrowing sequence from an initial state to an attack state is searched *in reverse* as a *backwards path* from the attack state to the initial state. Maude-NPA attempts to find paths until it can no longer form any backwards narrowing steps, at which point it terminates. If at that point it has not found an initial state, the attack pattern is judged *unreachable*; providing a proof of security rather than finding attacks. Note that Maude-NPA places *no bound on the number of sessions*, so reachability is undecidable in general. Note also that Maude-NPA does not perform any data abstraction such as a bounded number of nonces. However, the tool makes use of a number of sound and complete state space reduction techniques that help to identify unreachable and redundant states, and thus make termination more likely.

Maude-NPA relies on equational unification to perform each backwards narrowing step. Given two terms u and v and an equational theory $E_{\mathcal{P}}$ associated to a protocol $\mathcal{P}$, a substitution σ is a $E_{\mathcal{P}}$-unifier of terms u and v (or a unifier modulo $E_{\mathcal{P}}$) if $\sigma(u) =_{E_{\mathcal{P}}} \sigma(v)$. Maude-NPA provides built-in support for theories involving symbols with any combination of associativity (A), commutativity (C), and identity (U) axioms. Furthermore, by relying on variant-based equational unification [4], Maude-NPA allows users to augment the basic set of equational axioms supported with rewrite rules. Theories that can be supported this way include cancellation of encryption and decryption, Diffie-Hellman exponentiation, exclusive-or, and some approximations of homomorphic encryption.

IV. The Diffie-Hellman Example

In this section we give a brief description of an analysis of unauthenticated Diffie-Hellman, to show how Maude-NPA works. Diffie-Hellman uses exponentiation to share a secret between two parties. However, if it is not combined with some form of authentication it is subject to man-in-the-middle attacks. We give an example below of how Maude-NPA finds this attack.

We start by describing the strands of to honest principals.

$$(\mathbf{s1}) :: r, r' ::[\ (A; B; exp(g, n(A, r)))^+,\ (B; A; X)^-,\ (e(exp(X, n(A, r)), sec(A, r')))^+\ ]$$

This strand denotes principal Alice sending her name, Bob's name, and the generator g raised to the power of a new nonce generated by Alice using the Fresh variable r. Then, Alice waits for Bob's name, her name, and an unknown message X. Finally, Alice sends the secret, which uses another Fresh variable r', encrypted with X raised to her nonce.

$$(\mathbf{s2}) :: r'' ::[\ (A; B; Y)^-,\ (B; A; exp(g, n(B, r'')))^+,\ (e(exp(X, n(B, r'')), SR)^-\ ]$$

This strand denotes principal Bob waiting for Alice's name, his name, and an unknown message Y. Then, Bob sends his name, Alice's name, and the generator g raised to the power of a new nonce generated by Bob using the Fresh variable r''. Finally, Bob waits for the secret encrypted with Y raised to his nonce.

We also include strands describing the intruder abilities. These include encryption, decryption, concatenation, deconcatenation, exponentiation, multiplication, the generation of nonces, and fact that the intruder knows all names. To give an example, we give the strands describing encryption and decryption below.

$$(\mathbf{e})[K^-, M^-, e(K, M)^+] \quad (\mathbf{d})[K^-, M^-, d(K, M)^+]$$

The encryption/decryption cancellation properties are described using equations: $e(X, d(X, Z)) = Z$ and $d(X, e(X, Z)) = Z$. The key algebraic property on exponentiations $z^{x^y} = z^{x*y}$ is described using the equation: $exp(exp(W, Y), Z) = exp(W, Y * Z)$, where W is restricted to generators to provide a finitary narrowing-based unification procedure and symbol $*$ satisfies associativity and commutativity. The attack state pattern representing the famous man-in-the-middle attack is as follows (note that the strand is in its final position with the bar at the end):

$$:: r'' :: [(A; B; Y)^-,\ (B; A; exp(g, n(B, r'')))^+,\ (e(exp(X, n(B, r')), sec(a, r''))^- \mid nil]$$
$$\&\ SS\ \&\ \{sec(a, r') \in \mathcal{I},\ IK\}$$

Variables IK for the intruder knowledge and SS for the remaining strands are appropriately instantiated by narrowing. Our tool is able to find the following an initial state of the protocol, showing that the attack state is reachable. The concrete message exchange sequence is the following:

1.+(a;B;exp(g,n(a,r)))
2.-(a;B;exp(g,n(a,r)))
3.+(B;exp(g,n(a,r)))
4.-(B;exp(g,n(a,r)))
5.+(exp(g,n(a,r)))
6.-(exp(g,n(a,r)))
7.-(E)
8.+(exp(g,E*n(a,r)))
9.-(a;B;exp(g,E))
10.+(e(exp(g,E*n(a,r)),sec(a,r")))
11.-(a;b;exp(g,E'))
12.+(a;b;exp(g,n(b,r')))
13.-(a;b;exp(g,n(b,r')))
14.+(b;exp(g,n(b,r')))
15.-(b;exp(g,n(b,r')))
16.+(exp(g,n(b,r')))
17.-(exp(g,n(b,r')))
18.-(E')
19.+(exp(g,E'*n(b,r')))
20.-(exp(g,E*n(a,r)))
21.-(e(exp(g,E*n(a,r)),sec(a,r")))
22.+(sec(a,r"))
23.-(exp(g,E'*n(b,r')))
24.-(sec(a,r"))
25.+(e(exp(g,E'*n(b,r')),sec(a,r")))
26.-(e(exp(g,E'*n(b,r')),sec(a,r")))

Thanks: S. Escobar was partially supported by the EU (FEDER) and the Spanish MINECO under grant TIN 2015-69175-C4-1-R, by the Spanish Generalitat Valenciana under grant PROMETEOII/2015/013.

Referencias

[1] S. Escobar, C. Meadows, and J. Meseguer. A rewriting-based inference system for the NRL protocol analyzer and its meta-logical properties. *Theoretical Compute Science*, 367(1–2):162–202, 2006.

[2] Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-NPA Manual, Version 3.1.

[3] Santiago Escobar, Catherine Meadows, and José Meseguer. Maude-NPA: Cryptographic protocol analysis modulo equational properties. In *Foundations of Security Analysis and Design V, FOSAD 2007/2008/2009 Tutorial Lectures*, LNCS vol. 5705, pages 1–50. Springer, 2009.

[4] Santiago Escobar, Ralf Sasse, and José Meseguer. Folding variant narrowing and optimal variant termination. *The Journal of Logic and Algebraic Programming*, 81(7-8):898–928, 2012.

[5] F. J. Thayer Fabrega, J. Herzog, and J. Guttman. Strand Spaces: What Makes a Security Protocol Correct? *Journal of Computer Security*, 7:191–230, 1999.

[6] C. Meadows. The NRL protocol analyzer: An overview. *Journal of Logic Programming*, 26(2):113–131, 1996.

[7] J. Meseguer. Conditional rewriting logic as a unified model of concurrency. *Theoretical Computer Science*, 96(1):73–155, 1992.

A review of a Fair Protocol for Data Trading Based on Bitcoin Transactions

Sergi Delgado-Segura, Cristina Pérez-Solà, Guillermo Navarro-Arribas, Jordi Herrera-Joancomartí
Dep. of Information and Communications Engineering,
Universitat Autònoma de Barcelona
sdelgado@deic.uab.cat, cperez@deic.uab.cat, guillermo.navarro@uab.cat, jordi.herrera@uab.cat

***Abstract*—We present a fair protocol for data trading where the commercial deal, in terms of delivering the data and performing the payment, is atomic since the seller cannot redeem the payment unless the buyer obtains the data and the buyer cannot obtain the data without performing the payment. The protocol is based on Bitcoin scripting language and the fairness of the protocol can be probabilistically enforced.**

***Index Terms*—Bitcoin, fair exchange, blockchain.**

Tipo de contribución: *Investigación ya publicada (límite 2 páginas)*

I. Introduction

We propose a fair protocol for data trading on Bitcoin. Our protocol is fair since none of the participants have an advantageous position in the execution of the protocol. The protocol is atomic in the sense that either it is fully executed, ending the buyer with the data and the seller with the payment, or no party incurs in any loss. Our proposal is a practical one, based on Bitcoin scripting language, and can be deployed using existing technology, in contrast to other existing theoretical approaches. This is a review of the paper published as [1].

II. Bitcoin time and private key locked transactions

We refer the reader to [2] for detailed description of Bitcoin, and outline here only a few basic building blocks for our proposal: Bitcoin time locked transactions and private key locked transactions.

Time locked transaction outputs require a certain time in the future to be reached in order to be redeemed. There are two types of time-locks depending on whether the future time is absolute to Bitcoin, or relative to the transaction publishing time.

Private key locked transactions [3] are another special case of Bitcoin transactions in which the transaction output can be redeemed by anyone who provides a private key corresponding to a given public key.

Two different approaches can be used to implement private key locked transactions, via the definition of a new Bitcoin opcode or by taking advantage of a well-known vulnerability of ECDSA algorithm. The second one is more interesting since it does not require the introduction of a new opcode in Bitcoin which will require a soft-fork.

III. Implementing the private key locked transaction

The implementation of the private key locked transaction using the ECDSA signature vulnerability, described in [3], can be done using Bitcoin script. The `ScriptPubKey` of the output (and its corresponding `ScriptSig`) that implement the mechanism described above are depicted in Table I.

`ScriptPubKey:`	`OP_DUP <pubKey> OP_CHECKSIGVERIFY OP_SIZE <0x47> OP_EQUALVERIFY <sigmask> OP_AND <kprev> OP_EQUAL`
`ScriptSig:`	`<sig>`

TABLE I
Private key locked transaction scripts.

First, the script validates the signature against the specified public key. Then, the length of the signature is checked. Finally, a bitwise AND between the new signature and sig_{mask} is computed, and the result is compared with the k value of the previous signature. sig_{mask} is a byte array that has 1s on selected positions and 0s in the rest of positions in order to be able to extract information from the k value of the signature (see [3] for more details). If both values are equal, the script terminates successfully; otherwise, the script terminates with a False value on the stack, making it fail.

Note that the only way to ensure that the script succeeds is by providing a valid signature that has exactly the same k as the previous signature. Therefore, although the redeem `ScriptSig` that spends the output does not include the value of the private key directly, it is implicitly leaking its value by the ECDSA vulnerabilty.

Also note that the `ScriptSig` needed to spend the output only requires one value: the new signature.

IV. The data trading protocol

The protocol is run by two parties and no additional party, like a TTP is needed. Since our protocol is based on Bitcoin, both parties need to be connected with the Bitcoin network to send/receive transactions from the blockchain.

In our scenario, the buyer, B, wants to buy some data D to the seller, S, and he is willing to pay x bitcoins for such data. We assume that the data being sold can be divided in n different parts and each of those parts may have a meaning by itself, e.g. a movie or song clip, sensor data, etc.

A. Protocol description

The full protocol (Figure 1), can be divided in three main parts: the *Data correctness proof*, in which a cut & choose protocol between B and S is performed in order to convince B that the acquired data is correct; the *Signature commitment*, used for B to obtain a previous signature performed by S with

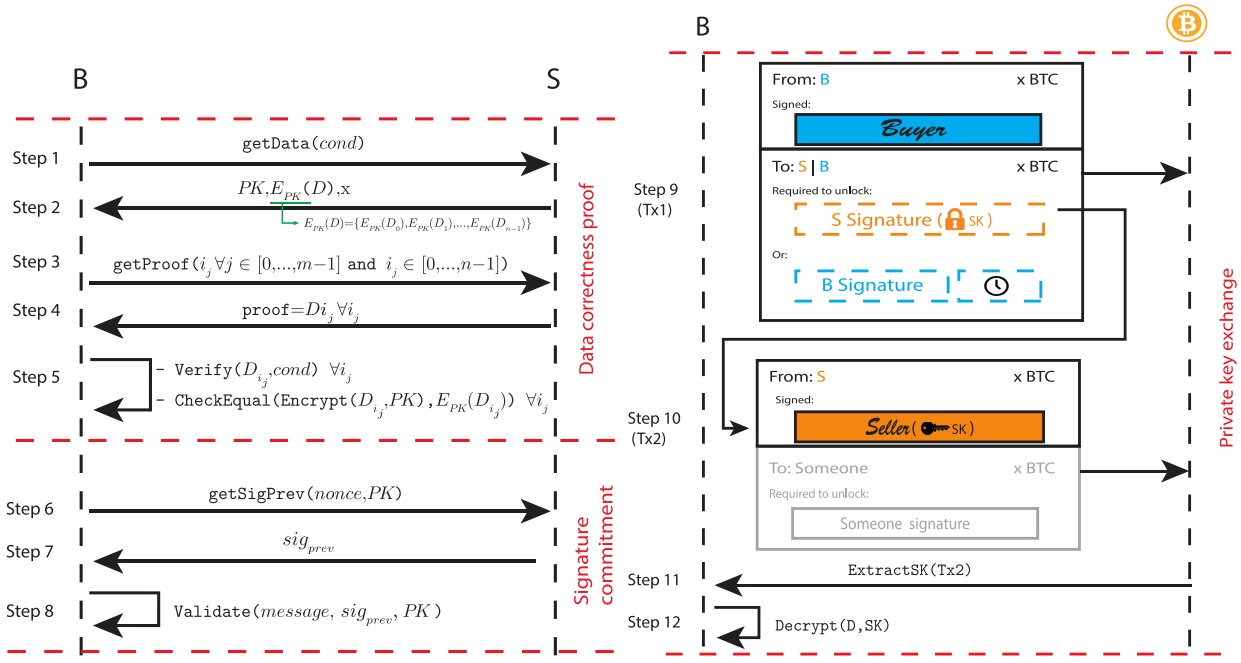


Fig. 1. Fair data trading protocol.

the private key used to encrypt the data; and the *Private Key Exchange*, used to exchange, atomically, the private key that allows to decrypt the sold information for the agreed amount of bitcoins.

We denote by $\{PK, SK\}$ a public key pair and $E_{PK}(\cdot)$ the encryption function using the public key PK.

1) Data correctness proof subprotocol: The buyer B starts the protocol by requesting data to the seller S. In such first step, B will indicate to S the data he is wiling to buy, and the conditions $cond$ that the data being sold has to hold. Then S generates a new $\{PK, SK\}$ and sends (Step 2): the public key PK, the requested data D encrypted using PK, and the data price x. In order to allow B to prove the data correctness, S does not send the D as a whole bunch of encrypted data, but split in n chunks which are encrypted individually: $E_{PK}(D) = \{E_{PK}(D_0), E_{PK}(D_1), ..., E_{PK}(D_{n-1})\}$.

Then B requests a correctness proof to S consisting in a random subset of non-encrypted data from D. To that end, B selects the subset by randomly choosing a set of m pieces from the encrypted dataset, that is $i_j \ \forall j \in [0, m-1]$, $i_j \in [0, n-1]$. B sends this information and S can build the correctness proof by choosing the unencrypted pieces of data that matches the received indexes, that is, $proof = \{D_{i_j} \quad \forall j \in [0, m-1], \ i_j \in [0, n-1]\}$. S sends such correctness proof to B.

Once B has received the $proof$, he verifies the correctness of D by checking that the proof satisfies the conditions. Furthermore, B validates that the received data also matches with the subset of received encrypted data, by recreating the data encryption using PK. Since the subset has been randomly chosen by B, the correctness of the full dataset can be proved with a given probability (see [1]).

2) Signature commitment subprotocol: B requests a signature Sig_{prev} over a nonce message performed with the private key SK generated by S. S sends Sig_{prev} and B validates that the signature is correct, using the public key PK that has received in Step 2.

3) Private Key Exchange subprotocol: B builds a private key locked transaction, Tx_1, to perform the atomic exchange between the private key, SK, and the bitcoin price x. The private key locked transaction includes another time constrain used for B to recover the amount of x bitcoins in case S decides not to reveal the private key by not spending the received transaction. B broadcasts the transaction Tx_1 to the Bitcoin P2P network. Once Tx_1 is included in a block, S can spend the output of such transaction with an input of a new transaction Tx_2 in which S will provide the second signature with the same k of sig_{prev}. Once Tx_2 appears on the blockchain, B will be able to recover the private key SK and decrypt the data $E_{PK}(D)$ he received in Step 2 to retrieve the purchased data.

V. Conclusion

This paper summarizes the proposal of a fair data trading protocol based on Bitcoin transactions. We refer the reader to the original publication [1] for more detailed information.

Acknowledgements

Work supported by the Spanish MINECO, grant TIN2014-55243-P, and Catalan AGAUR, grant 2014SGR-691.

References

[1] S. Delgado-Segura, C. Pérez-Solà, G. Navarro-Arribas, and J. Herrera-Joancomartí, "A fair protocol for data trading based on bitcoin transactions," *Future Generation Computer Systems*, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X17318344

[2] A. Narayanan, J. Bonneau, E. Felten, A. Miller, and S. Goldfeder, *Bitcoin and cryptocurrency technologies*. Princeton University Pres, 2016.

[3] S. Delgado-Segura, C. Pérez-Solà, J. Herrera-Joancomartí, and G. Navarro-Arribas, "Bitcoin private key locked transactions (short paper)," Cryptology ePrint Archive, Report 2016/1184, 2016, http://eprint.iacr.org/2016/1184.

Alternativas a Bitcoin y su Uso en el Cibercrimen

Esteban Alejandro Armas Vega[1], Ana Lucila Sandoval Orozco[1], Antonio López Vivar[1]
Carlos Quinto Huaman[1], Daniel Povedano Álvarez[1], Edgar González Fernández[1], Luis Javier García Villalba[1*]
Julio Hernandez-Castro[2], Tatiana Silva[3], Alejandro Prada[3]

[1]Grupo de Análisis, Seguridad y Sistemas (GASS)
Departamento de Ingeniería del Software e Inteligencia Artificial (DISIA)
Facultad de Informática, Despacho 431, Universidad Complutense de Madrid (UCM)
Calle Profesor José García Santesmases, 9, Ciudad Universitaria, 28040 Madrid, España
Emails: {esarmas, cquinto, edggonza, dpovedano}@ucm.es, {asandoval, alopezvivar, javiergv}@fdi.ucm.es

[2]School of Computing, University of Kent
Cornwallis South Building, Office 120
Canterbury CT2 7NF, United Kingdom
Email: j.c.hernandez-castro@kent.ac.uk

[3]Treelogic, Avda. Manoteras 38, Oficina D614, 28050 Madrid, España
Email: {tatiana.silva, alejandro.prada}@treelogic.com

***Resumen*—Bitcoin es una criptomoneda muy popular actualmente. Utiliza una tecnología llamada blockchain para su funcionamiento. Blockchain consiste en un libro mayor contable distribuido. Esta red distribuida está conformada por nodos, que son un conjunto de ordenadores. Dichos nodos se encargan de verificar que las transacciones se lleven a cabo y de esta forma lo validan. Esta tecnología no es única para Bitcoin, desde su nacimiento han aparecido muchas de las llamadas Altcoins. Estas criptomonedas alternativas se promocionan como una ventaja a Bitcoin gracias a su bajo coste, alta velocidad o mayor privacidad en sus transacciones. Operadores de malware, mercados ilegales posicionados en la web oscura o profunda han sabido utilizar Bitcoin para operar de manera pseudo-anónima y obtener redito económico de sus delitos. Sin embargo, en años recientes ha habido arrestos de cibercriminales de gran perfil gracias a los grafos de transacciones de Bitcoin, lo cual enfatiza el pseudo-anonimato de Bitcoin. Es por esto que las nuevas alternativas a Bitcoin se perfilan como una opción atractiva para su uso por parte de cibercriminales. Este trabajo puntualiza las características principales que los cibercriminales buscan a la hora de utilizar una criptomoneda diferente a Bitcoin y analiza su presencia en el mercado.**

***Index Terms*—Análisis Forense, Bitcoin, Cibercrimen, Ciberseguridad, Criptomoneda, Dark Net, Deep Web, Horizonte 2020, Malware, Plataforma, Ransomware, Sistema Inteligente, Troyano Bancario**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

En 2013, CryptoLocker por primera vez hizo uso de la plataforma de moneda digital de Bitcoin. Las ventajas de utilizar criptomonedas se hicieron evidentes a continuación, con una ganancia total, acreditada a los operadores del malware, de $ 3,000,000 [1], contrariamente a las cifras iniciales de $ 27 millones. Aunque CryptoLocker en sí se vio frustrado a principios de junio de 2014, el uso de la moneda digital como medio para cobrar rescates a gran escala se había demostrado eficaz y sentó un precedente. Desde entonces, el malware y las criptomonedas se han desarrollado rápidamente. Las monedas alternativas, han surgido en el mercado de divisas digital y se han diversificado rápidamente para ofrecer sus propios servicios. Un enfoque común es el uso de estas criptomonedas como combustible para un servicio relacionado, por ejemplo, Siacoin que se usa como medio de intercambio para su sistema de contrato de almacenamiento de archivos distribuidos [2]. Bitcoin, a fecha 31 de diciembre de 2017, cotizaba sobre los $ 9300 [1] por moneda, pero la volatilidad sigue siendo una característica definitoria de la mayoría de las criptomonedas. Se puede argumentar que el valor de la criptomoneda es secundario a sus otros atributos que se prestan a la actividad criminal. El anonimato de Bitcoin ha sido cuestionado repetidamente [3] y las nuevas criptomonedas que se centran en la privacidad y el anonimato han surgido como resultado de ello. Zcash, Dash y Monero ingresaron en un mercado que tiene una demanda creciente de monedas completamente anónimas y no trazables. La usabilidad es ahora un tema candente para muchas comunidades de moneda digital, que quieren reducir la barrera de entrada para los usuarios que lo utilizan por primera vez. Incluso ha habido discusiones hipotéticas sobre contratos inteligentes (un atributo de Ethereum y monedas de diseño similar) para automatizar aún más la cosecha de rescates de malware, ofreciendo muchos más beneficios en escalabilidad, velocidad y servicio para los operadores de malware.

Este trabajo proporciona una discusión contemporánea y prospectiva de las nuevas criptomonedas alternativas a Bitcoin y su papel en el malware con fines de lucro, con un enfoque en ransomware.

II. Nuevas Monedas

Altcoin es un término un tanto polémico en la comunidad de criptomonedas, y algunas comunidades de Ethereum y Monero argumentan que se refiere específicamente a monedas alternativas construidas en la plataforma de Bitcoin. En este trabajo se utilizará altcoin para referirse a todas las criptomonedas post-Bitcoin, incluidas las que no se basan en la plataforma de moneda digital de Bitcoin.

II-1. Monero XMR: Monero se promociona a sí mismo como una moneda segura, privada e imposible de rastrear. Al igual que muchas Altcoins, tiene una próspera comunidad de

Reddit y presencia en las redes sociales, con un enfoque en propugnar los beneficios de la privacidad en la sociedad civil. También ha visto un alto grado de atención por parte de los medios de comunicación en el último año, debido a su uso para el intercambio de fondos criminales.

II-2. Ethereum ETH/ETC: Ethereum en sí no es una moneda; es una red distribuida, descrita por primera vez en 2013 por Vitalik Buterin. Ether, la moneda que alimenta esa red, se usa para almacenar valor, que puede estar asociada a los llamados contratos inteligentes [4]. Estos contratos son acuerdos digitales, almacenados en blockchain, que permiten intercambiar una variedad de servicios para Ethereum. Ethereum no ha escapado a la atención de los ciberdelincuentes. CradleCore es un ejemplo de ransomware como servicio, que incluía la opción de que el ransomware permita el pago en Monero y Ethereum [5].

II-3. ZCash ZEC: Introducido en octubre de 2016, ZCash afirma ser la primera criptomoneda abierta y sin permisos que puede proteger completamente la privacidad de las transacciones utilizando protocolos criptográficos que incluyen pruebas de conocimiento cero. Al igual que Monero, proporciona servicios adicionales de privacidad y anonimato. De esta manera, su blockchain está protegido contra la observación. Todas las transacciones son registradas y verificadas por consenso, pero este se logra de manera que tanto el destinatario como el remitente permanezcan en el anonimato. Los detalles de la transacción también están protegidos y se mantienen privados. No existe información relativa al uso de Zcash en ransomware o ransomware-as-a-service (RaaS). Es posible que ZCash actualmente no sea tan popular entre los operadores de malware y de mercados en la red oscura y profunda (debido a una serie de problemas de confianza, entre los que destaca su registro como empresa de los Estados Unidos), pero ZCash sí ofrece un servicio diferente.

II-4. Dash DASH: Dash ha sido conocido anteriormente como DarkCoin y XCoin. Lanzado el 18 de enero de 2014 (como XCoin), su nombre fue cambiado a DarkCoin el 28 de febrero del mismo año. La moneda finalmente fue renombrada como Dash (Digital Cash) el 25 de marzo de 2015. Ofrece la misma funcionalidad básica que Bitcoin, junto con transacciones instantáneas, transacciones privadas y descentralizadas. La mayor diferencia estructural entre Dash y Bitcoin es la arquitectura de dos niveles que emplea para permitir nodos maestros [6]. Varios mercados en la red oscura y profunda aceptan Dash. Muchos blogs de criptomonedas, incluida la revista Bitcoin, plantean que la comunidad de la red oscura y profunda prefiere Monero sobre ZCash y Dash porque no confían en ninguno de estos protocolos alternativos [7]. A pesar de esto, Dash sigue siendo una alternativa válida para los medios descentralizados de lavado de dinero asociado con los cibercriminales.

III. Criptomonedas y Cibercrimen

Bitcoin es, con mucho, la criptomoneda dominante, tanto en términos de rendimiento del mercado como en la utilización por parte del cibercrimen. Desde CryptoLocker, la mayoría de las familias de ransomware han utilizado Bitcoin para sus pagos. La naturaleza descentralizada y desregulada del blockchain lo hace ideal para actividades ilícitas, aunque no es resistente a las técnicas analíticas de rastreo. Esto ha llevado a un interés en criptomonedas alternativas que cuentan con características mejoradas de anonimato y privacidad. Características como: facilidad de adquisición, experiencia de usuario, liquidez de la moneda y principalmente la seguridad y el anonimato a la hora de realizar transacciones.

El foco de cualquier investigación relacionada con criptomonedas es el flujo de dinero [8]. En algún momento, los operadores de un malware o mercados dentro de la red oscura y profunda querrán retirar sus ganancias en moneda fiduciaria, para gastarlas dentro del mercado normal. Esto se logra mediante intercambio o la compra de bienes para su posterior reventa, pero la consideración importante es evitar la detección. Los delincuentes, especialmente los sofisticados, también se darán cuenta de que como *blockchain* es inmutable, es aconsejable cubrir su rastro contra cualquier posible investigación en el futuro. Esto se puede lograr de muchas maneras: mezclando monedas o intercambiando lo adquirido por una Altcoin apropiadamente centrada en la privacidad siendo las dos estrategias las más comunes.

IV. Conclusiones

La privacidad y el anonimato son buscados por los operadores de malware y por los comerciantes de mercados dentro de la red oscura y profunda, pero el rol de las monedas como Monero, ZCash y Dash aún se están definiendo. Un ransomware denominado Kirk, hace poco descubierto, exige pago en criptmonedas distintas a Bitcoin, específicamente en Monero. El ransomware CryptoCore aún no se ha implementado, pero se ha puesto a la venta. Este código fuente de ransomware proporcionará al comprador la capacidad de implementar un ransomware que solicita el pago en Monero o Ethereum. La adopción de Monero y la existencia de varias Altcoins indican que el éxito frente a las formas tradicionales de malware y el pago en el mercado de la red oscura y profunda fomentarán la adopción de alternativas nuevas distintas a Bitcoin. Sin embargo, el beneficio no radica en la capacidad de solicitar múltiples monedas, sino en lo que ofrecen esas monedas.

Agradecimientos

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 700326. Website: http://ramses2020.eu

Referencias

[1] B. Akolkar, "Top 10 Cryptocurrencies of 2017 - WorldCoinIndex," https://www.worldcoinindex.com/news/top-10-cryptocurrencies-of-2017, 2017.
[2] Sia Tech, "Sia," https://sia.tech/, 2014.
[3] F. Reid and M. Harrigan, *An Analysis of Anonymity in the Bitcoin System*. New York, NY: Springer New York, 2013, pp. 197–223.
[4] Ethereum Project, "Ethereum Project," https://ethereum.org/.
[5] J. P. Buntinx, "CradleCore May Introduce new Monero and Ethereum Ransomware – The Merkle," https://themerkle.com/cradlecore-may-introduce-new-monero-and-ethereum-ransomware/.
[6] DASH, "Sitio Web Oficial de Dash — Moneda Digital Privada — Dash," https://www.dash.org/es/.
[7] J. Kouki, "Official Minergate Blog - Official Minergate Blog," https://minergate.com/blog/.
[8] A. Narayanan and M. Möser, "Obfuscation in Bitcoin: Techniques and Politics," *arXiv preprint arXiv:1706.05432*, 2017.

A review of Sustainable Securing of Medical Cyber-Physical Systems for Future Healthcare

Manuel Gil Pérez, Alberto Huertas Celdrán, Félix J. García Clemente, and Gregorio Martínez Pérez

Faculty of Computer Science, University of Murcia, 30100 Murcia, Spain
Email: mgilperez@um.es, alberto.huertas@um.es, fgarcia@um.es, gregorio@um.es

***Abstract*—Medical Cyber-Physical Systems (MCPS) is a new disruptive approach oriented to enable smart healthcare systems to monitor, process, and make autonomous decisions without needing to involve doctors and caregivers. Yet, current MCPS pose several open challenges, being security of medical devices one of the most critical for its close relationship with patients' safety. To deal with these issues, we propose in this paper a fog computing-oriented framework to enable real-time and dynamic management of physical and virtual medical devices, as well as a network infrastructure making up MCPS. To demonstrate the theoretical feasibility of our solution, we designed a use case that depicts security concerns of current MCPS.**

***Index Terms*—MCPS, Security, SDN/NFV, Fog Computing**

Tipo de contribución: Investigación ya publicada

I. Introduction

Historically, medical devices have been developed as stand-alone systems without communication capabilities. Yet, the disruptive vision of Medical Cyber-Physical Systems (MCPS) enables the promising next-generation of eHealth systems that are intended to interoperate efficiently, safely, and securely. Within this context, safety-critical interconnected systems that analyze patients' vital signs gathered from medical devices to infer the state of patient's health, and start treatments issuing information to doctors and medical actuators, should improve patients' safety in cost-efficient way.

Any implementation of the MCPS vision should consider several critical pillars and challenges. Among the different existing challenges, we highlight the security of MCPS. Since medical systems contribute to the care received by patients and manage their private information, any MCPS implementation must be secure and robust. However, the current healthcare industry lags behind in security. Healthcare solutions should clearly define their cyber security issues and establish clear procedures for managing attacks and data breaches. In this sense, current implementations of MCPS do not have enough capabilities to detect and mitigate cyber-attacks. This is a critical shortcoming because cyber-attacks affecting the security of MCPS components can endanger patient's life.

In order to address the aforementioned challenge, the close combination of fog computing with new technologies such as Network Function Virtualization (NFV) and Software Defined Networking (SDN) enables not only the increasing demand of simultaneous medical devices, bandwidth, and latency being required by MCPS, but also provides a flexible, scalable, and efficient resource management scenario suitable for MCPS. Specifically, this integration enables the deployment and management of medical and networking resources in the edge of the network to guarantee the security of MCPS.

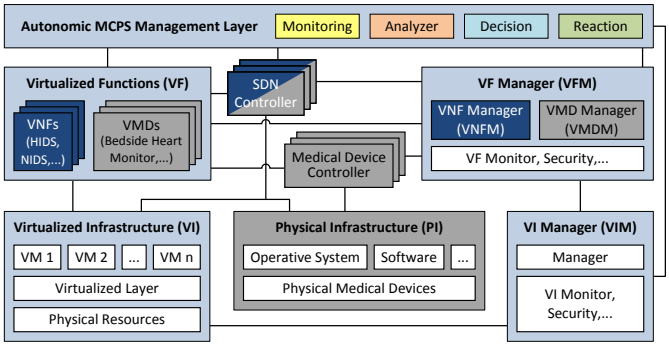


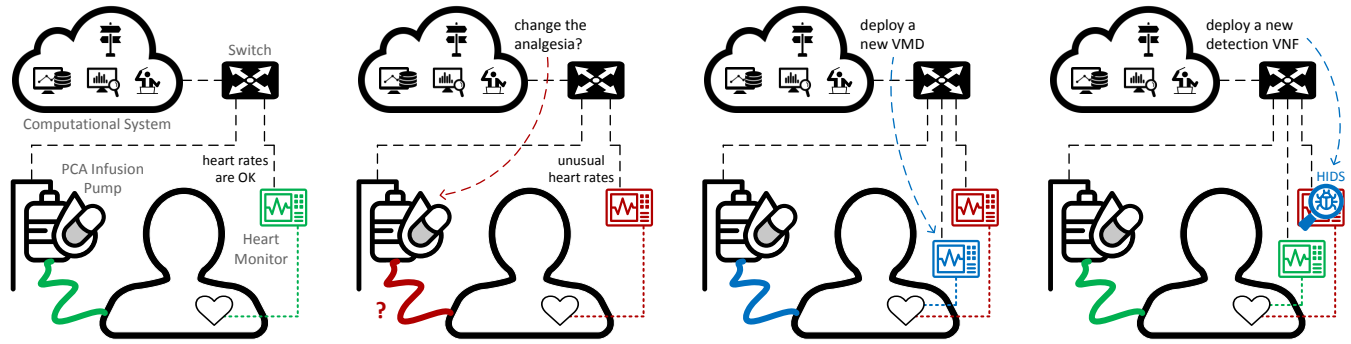
Fig. 1. Architecture of the proposed MCPS framework.

In this context, this paper is a review of [1] whose main contribution is a novel autonomic and sustainable framework supported by the fog computing paradigm. This framework is able to manage complex MCPS scenarios by considering both SDN and NFV technologies. The novelty of this framework is to enable the autonomic deployment and self-configuration in real time of Physical and Virtual Medical Devices (PMD and VMD). This novelty also lies in creating, modifying, and rearranging Virtual Network Functions (VNF) in the edge of the network infrastructure for improving the current security challenge of MCPS. To the best of our knowledge, our solution is the first combining these technologies to solve security problems of MCPS, for which the proposed framework will provide a flexible control of MCPS resources.

II. Proposed Architecture

This section describes our architecture supported by the fog computing paradigm, integrating the SDN paradigm with the ETSI NFV reference architecture to manage PMDs, VMDs, and network resources facing the MCPS security challenge. Fig. 1 depicts the blocks and internal connections making up the architecture, where medical devices and management elements are in gray and network resources in dark blue.

At Fig. 1, the *Virtualized Infrastructure Manager* (VIM) is in charge of creating, controlling, monitoring, and securing the whole life cycle of *Virtual Machines* (VM) instantiated on generic *Physical Resources* through the *Virtualized Layer*. At the same level, the *Physical Medical Devices* together with their *Software* are included in the *Physical Infrastructure* (PI). On top of both blocks, the *VF Manager* (VFM) is responsible for deploying, controlling, monitoring, and securing *Virtualized Functions* (VF). Specifically, the VFM consists of two managers: the *Virtual Network Function Manager* (VNFM) and the *Virtualized Medical Devices Manager* (VMDM). The



(a) Initial scenario (b) Considered scenario (c) Auditing scenario (d) Detection scenario

Fig. 2. MCPS scenario with security capabilities.

former creates, manages, and dismantles VNFs running on the VMs exposed by the *Virtualized Infrastructure* (VI), while the latter creates, manages, and also dismantles VMDs on the VMs provided by the VI. Thus, the combination of VNFs and VMDs provides flexibility and cost-efficiency during detection and mitigation of cyber-attacks.

At the same level as the VF, the *Medical Device Controller* enables interoperability of PMDs and VMDs by understanding the protocols of each medical device. This component is also able to control and configure the software of medical devices remotely (critical for reacting against cyber-attacks). The SDN paradigm manages network communications and connectivity of PMDs and VMDs automatically. The *SDN Controller* and its applications are capable of managing and mitigating cyber-attacks to medical devices. On the other hand, the *Autonomic MCPS Management Layer* (at top of Fig. 1) is able to monitor the information gathered from previous elements, analyzing it and making decisions to ensure security of MCPS.

III. Use Case Securing MCPS

This section describes cyber security concerns that current MCPS solutions may have during critical interventions. Let us suppose a patient being controlled by a MCPS with several medical devices monitoring patient's vital signs in real time; a network infrastructure to enable interoperability of medical devices; a computational system in charge of acquiring data and making decisions on patient's treatment; and a Patient-Controlled Analgesia (PCA) pump to deliver analgesia. Fig. 2 shows the normal status of medical devices (in green), security concerns of medical devices (in red), and actions took by our framework to manage such concerns (in blue).

Fig. 2a shows the initial situation of the scenario, where the patient's heart rate is being monitored by a *bedside heart monitor* that, at a given time, starts reporting that heart rate is out of usual range (Fig. 2b). However, correlated vital signs from the rest of medical devices do not indicate variations in patient's condition. At this point, the first concern consists of deciding if the MCPS should change the amount of analgesia according to the data sensed by the bedside heart monitor. To solve the concern, current solutions of MCPS report an alert to the medical staff to place and configure another physical monitor manually. However, this solution is not optimal in a medical scenario where response time is critical. Due to that, our framework analyzes patient-sensed information, detects the anomaly, and creates a new virtual bedside heart monitor in a generic hardware located in the patient's room.

After the previous steps, the *Computational System* detects that data reported by both devices is different. At this point, Fig. 2c shows the second concern, which consists of detecting whether the bedside heart monitor has suffered a cyber-attack or it is just a system failure. To this end, current MCPS have no mechanisms to automatically manage medical devices and detect network attacks or system failures. Yet, our architecture is able to deploy, control, and dismantle VNFs, when needed, focused on the monitoring and detection of malware running in medical devices. Fig. 2d shows the deployment of a given HIDS in the bedside heart monitor (in blue), which will be placed at the patient's room.

IV. Conclusions and Future Work

An autonomic and sustainable framework supported by the fog computing paradigm has been presented along this paper, which makes use of NFV and SDN techniques to enable real-time dynamic management of the security in MCPS. To meet this challenge, the proposed framework controls Physical and Virtual Medical Devices in real time, as well as the network infrastructure that enables interoperability of MCPS.

Based on the theoretical architecture presented above, as future work our plan is to exercise all the proposed elements in a real medical environment, and thereby extracting all possible information to assess and evaluate architecture feasibility.

Acknowledgements

This work has been supported by a postdoctoral INCIBE grant within the "Ayudas para la Excelencia de los Equipos de Investigación Avanzada en Ciberseguridad" Program, with code INCIBEI-2015-27352, a Séneca Foundation grant within the Human Resources Researching Postdoctoral Program 2018, the European Commission Horizon 2020 Programme under grant agreement number H2020-ICT-2014-2/671672 - SELFNET (*Framework for Self-Organized Network Management in Virtualized and Software Defined Networks*), and the European Commission (FEDER/ERDF).

References

[1] A. Huertas Celdrán, M. Gil Pérez, F. J. García Clemente, G. Martínez Pérez: "Sustainable Securing of Medical Cyber-Physical Systems for the Healthcare of the Future," Sustainable Computing: Informatics and Systems, In Press. DOI 10.1016/j.suscom.2018.02.010

Desarrollo de un sistema de trazabilidad en entornos IoT mediante Hyperledger

Jesús Iglesias García
Escuela Politécnica Superior
Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
jesusgiglesias@gmail.com

David Arroyo Guardeño
Escuela Politécnica Superior
Departamento de Ingeniería Informática
Universidad Autónoma de Madrid
david.arroyo@uam.es

***Resumen*—En la actualidad un grueso importante de empresas tienen un interés especial en construir una nueva generación de aplicaciones transaccionales que establezcan confianza, responsabilidad y transparencia en su núcleo junto con una arquitectura abierta y distribuida. Aquí es de especial relevancia el sistema de pago de gestión no centralizado Bitcoin [9], que está basado en la tecnología Blockchain (BC) la cual habilita la creación de un sistema inmutable de registro de eventos. Desde su origen ha propiciado todo un conjunto de iniciativas que, prescindiendo de la existencia de una tercera parte de confianza, puedan proporcionar protocolos para la protección de la integridad de la información.**

Una de estas iniciativas es Hyperledger [2] -*standard open source* de BCs permisionadas-, se centra en apoyar este tipo de transacciones de negocio para mejorar numerosos aspectos de rendimiento y empresariales donde la inserción de información requiere que las entidades sean previamente autenticadas. Este modelo de control de acceso es de alto interés en el contexto de la trazabilidad de recursos y productos de una organización. El presente proyecto aborda el desarrollo de una prueba de concepto (*Proof of Concept* -PoC-) para implementar Hyperledger Fabric [3] en el Internet de las Cosas (*Internet of Things* -IoT-). El objetivo será la recolección de eventos mediante sensores situados en una Raspberry Pi 3 (RPi3), y la inclusión de incidencias en Hyperledger Fabric. Asimismo, la información introducida podrá ser consumida en tiempo real mediante un cliente web.

***Index Terms*—Hyperledger, Blockchain, Seguridad, Privacidad, Internet de las Cosas**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

Internet ha cambiado la forma de vida y la sociedad en general. Nuestra actividad diaria depende cada vez más de la información que obtenemos de Internet, de forma que necesitamos contar con mecanismos que nos permitan dirimir si los datos que obtenemos son fiables. Esta cuestión se ha resuelto tradicionalmente a través de alguna suerte de Tercera Parte Confiable (TPC). Sin embargo, la participación de intermediarios también presenta desventajas, entre las que se encuentra la posible degradación de nuestra privacidad si la TPC accede sin permiso a nuestra información sensible y personal.

Una posible solución es la descentralización de la gestión de la información, de forma que no se necesita una autoridad central intermediaria que tenga acceso a los datos. Aquí, es donde nace el concepto de cadena de bloques o *blockchain*, un libro de registros o transacciones (*ledger*) distribuido, del que todos los que participan en la red almacenan una copia que se actualiza mediante un protocolo *Peer-to-Peer* (P2P) de consenso. En este contexto, el protocolo de consenso distribuido se constituye en garante de la integridad de la información y, por ende, de su veracidad.

En su origen, BC se inventó para sustentar Bitcoin, la primera criptomoneda descentralizada no emitida por un banco central. Sin embargo, su aplicación no queda limitada a las criptomonedas. En efecto, la BC de Bitcoin habilita la escritura de información no vinculada a transacciones (por ejemplo, mediante el campo *OP_RETURN*). Esta información puede ser utilizada como canal de control de trazabilidad de las cadenas de producción (por ejemplo, Everledger), de certificación de documentación (como es el caso de Stampery y MaidSafe), de hipotecas (e.g, Zensar), de títulos o cualquier otro documento oficial (Bitfury, Factom, ChromaWay y Velox.re son ejemplos de *start-ups* que ofrecen este tipo de servicio), así como aplicaciones de control de integridad de información en ámbitos relacionados con la seguridad lógica[1]

Uno de los ámbitos donde BC tiene especial interés es en IoT [1]. La IoT constituye una red de dispositivos físicos que por naturaleza se conectan entre sí e intercambian datos para hacer nuestras vidas más sencillas y eficientes. Sin embargo, cada dispositivo puede ser cualquier *cosa* desde un televisor, un vehículo o hasta un frigorífico y todos con un funcionamiento y niveles de seguridad implementados diferentes. Esta variabilidad en las interfaces de acceso a la información y los mecanismos de intercambios de datos, introduce una incertidumbre en lo referente a las diversas fuentes de datos. Aquí es donde entraría en juego la BC, en específico aquellas evoluciones de la BC de Bitcoin mediante la incorporación de los denominados *smart contracts*[2] y de modelos de control de acceso.

II. Hyperledger

Hyperledger es una iniciativa de carácter colaborativo anunciado en el año 2015 por la fundación Linux para investigar y evolucionar la tecnología BC de uso privado y orientada al ámbito empresarial.

Dentro de los proyectos de este consorcio, el más conocido es la plataforma de BC permisionada: Hyperledger Fabric, proyecto que implementa la tecnología de libro de registros distribuido (*Distributed Ledger Technology* -DLT-) en la BC. En constante evolución[3] y con una hoja de ruta interesante [4],

[1] Aquí cabe destacar el proyecto Guardtime, que cuenta con el patrocinio de DARPA.

[2] Código informático que se ejecuta en la BC y hace cumplir un contrato de manera automática. Este concepto fue introducido por Nick Szabo en 1996 (ver [8], último acceso 01/05/2018).

[3] Hyperledger Fabric v1.1.0 fue liberado en la fecha 15 marzo 2018.

ofrece características [7] (arquitectura modular y escalable, red transaccional de alto rendimiento permisionada, privacidad e identidad, *chaincode -smart contracts* de Hyperledger-, etc.) que tienden a mejorar aspectos de productividad y fiabilidad distinguiéndola de otras alternativas de BC.

III. HYOT

Hyot es la PoC para la trazabilidad de un entorno controlado de IoT mediante la tecnología Hyperledger Fabric. Si bien cabe decir que es fácilmente escalable a ampliar un mayor rango de cobertura en cuanto a IoT se refiere. Esta solución gestiona de forma transparente para el usuario una serie de sucesos -temperatura, humedad, distancia, etc.- que son monitorizados desde sensores conectados a un ordenador de placa reducida (*Single Board Computer*) como es la RPi3. En caso de que se produzca una incidencia, las lecturas de los sensores son almacenados en una base de datos. Una incidencia no es sino una acción no controlada que tiene lugar en el entorno que se está vigilando, y origina la ejecución de un protocolo de alerta que comprende: (1) la captura de un vídeo a través de una cámara conectada a la RPi3; (2) almacenamiento en la nube del vídeo. En la medida que estos servicios de almacenamiento son una TPC que pueden poner en riesgo la privacidad de las evidencias, hemos asumido un modelo de confianza nula en el cual todos la información es cifrada mediante GPG (*GNU Privacy Guard*) antes de ser almacenada en al nube.

El punto central de Hyot es garantizar que el registro de un suceso anómalo no ha sido indebidamente modificado, de forma que una vez registrada una incidencia se tenga total certeza respecto a su integridad y veracidad. Este objetivo se consigue con Hyperledger Fabric, ya que proporciona un protocolo de consenso distribuido para la protección de integridad y un control de acceso que permite identificar a los agentes que introducen datos en la BC. En el dominio de nuestro caso de uso, se establecen las transacciones posibles a ejecutar junto con los activos (*assets*): marca temporal (*timestamp*) en la que ocurrió el suceso, el *hash* del contenido del vídeo encriptado -calculado con el algoritmo criptográfico SHA3- e incluso los valores de los sensores medidos.

Además, en este tipo de sistemas donde se controla un entorno (ya sea por cuestiones de seguridad o por cualquier otro motivo), es importante informar al administrador del sistema de que un suceso no controlado está sucediendo. Esto se consigue con la notificación de la información actual mediante un *email* a la dirección de correo electrónico configurada.

A modo resumen, se puede detallar que Hyot se compone de tres partes:

- Script de monitorización de los sucesos de los sensores de una RPi3.
- Protocolo de registro de incidencias en la BC de Hyperledger Fabric y almacenamiento de evidencias en la nube.
- Sistema web que actúa como cliente y consume la información en tiempo real registrada por la RPi3.

IV. CONCLUSIONES

La rápida evolución del mercado del IoT ha provocado una explosión en el número y la variedad de soluciones IoT, lo que ha creado grandes desafíos a medida que la industria evoluciona, principalmente, la necesidad urgente de un modelo IoT seguro para realizar tareas comunes como detección, almacenamiento y comunicación. Es por ello que la aplicación de la tecnología BC puede favorecer el despliegue de soluciones y arquitecturas más seguras.

Con este objetivo surge Hyot, un proyecto software de código abierto[4] que propone una prueba de concepto simple y novedosa sobre conceptos y tecnologías en auge como la BC. Con ello, se pretende mostrar las ventajas de las arquitecturas basadas en BC a la hora de diseñar e implementar protocolos de control y auditoría en IoT.

Por último, cabe mencionar que se trata de un proyecto en pleno desarrollo al momento de redactar este documento y que será presentado como Trabajo Fin de Máster (TFM) en el *Máster de Ingeniería Informática* de la Escuela Politécnica Superior de la Universidad Autónoma de Madrid (UAM). El ciclo de vida de este proyecto no pretende ser el habitual de un trabajo de universidad, y la idea principal es continuar con su desarrollo una vez presentado dado que se engloba dentro de un área con gran interés tanto para empresas nacionales como internacionales. Como parte del trabajo futuro se incorporarán nuevas funcionalidades relacionadas con la gestión de la identidad y su equilibrio con la protección de la privacidad[6], con la interoperabilidad con BC públicas [5], [11], así como con la incorporación de técnicas de aprendizaje automático que permitan identificar información sensible y relevante como evidencia digital [10].

AGRADECIMIENTOS

Este trabajo ha sido financiado a través de los proyectos CIBERDINE (S2013/ICE-3095) -Comunidad Autónoma de Madrid- y MINECO/FEDER DPI2015-65833-P (Gobierno de España).

REFERENCIAS

[1] Dorri, A., Kanhere, S. S., & Jurdak, R. (2017, April). "Towards an optimized blockchain for IoT." In Proceedings of the Second International Conference on Internet-of-Things Design and Implementation (pp. 173-178). ACM.
[2] Hyperledger. https://www.hyperledger.org
[3] Hyperledger Fabric. http://hyperledger-fabric.readthedocs.io/en/latest/
[4] Hyperledger Fabric Roadmap. https://wiki.hyperledger.org/projects/fabric/roadmap
[5] Hyperledger Sawtooth. https://www.hyperledger.org/projects/sawtooth
[6] Idemix (Identity Mixer) en Hyperledger Fabric. https://jira.hyperledger.org/browse/FAB-2005
[7] Marko Vukolic: "Hyperledger Fabric - An Open-Source Distributed Operating System for Permissioned Blockchains", Swiss Blockchain Summer School Lausann, 2017.
[8] Nick Szabo: "Formalizing and Securing Relationships on Public Networks", First Monday, vol.2, n.9, 1997.
[9] Pedro Franco: "Understanding Bitcoin: Cryptography, Engineering and Economics", Wiley Finance Series, 2014.
[10] Ramachandran, A., & Kantarcioglu, M. (2018, March). "SmartProvenance: A Distributed, Blockchain Based DataProvenance System." In Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy (pp. 35-42). ACM.
[11] Xu, R., Chen, Y., Blasch, E., & Chen, G. (2018). "BlendCAC: A BLockchain-ENabled Decentralized Capability-based Access Control for IoTs." arXiv preprint arXiv:1804.09267.

[4]El código fuente será liberado en Github una vez finalizado y presentado el Trabajo Fin de Máster.

Solución para la Securización de Comunicaciones con Requisitos de Tiempo Real en Infraestructuras Críticas

Jon Matias, Jokin Garay
Keynetic Technologies
Bilbao, España
{jon.matias, jokin.garay}@keynetic.es

Mikel Rodriguez, Armando Astarloa
System-on-Chip Engineering
Erandio, España
{mikel.rodriguez, armando.astarloa}@soc-e.com

Resumen- **La transformación digital del sector industrial permite generar un conocimiento más profundo de los procesos para mejorar su automatización y la toma de decisiones en tiempo real. Esta interconexión de estos sistemas tradicionalmente aislados tiene numerosas ventajas, pero también implica nuevos riesgos. Como resultado, las infraestructuras críticas se han convertido en un objetivo prioritario de los ciber-ataques debido al gran impacto que estos provocan. Para protegerse de estas amenazas, la industria ha desarrollado una serie de normas denominadas IEC 62351. Las empresas SoC-e y Keynetic han desarrollado una solución con requisitos de tiempo real para el sector de Smart Grids que implementa tanto la norma IEC 62351-6 para el cifrado y autenticación de las comunicaciones, como el IEC 62351-9 para la gestión de las claves criptográficas necesarias (generación, distribución y actualización).**

***Index Terms*- Ciberseguridad, Infraestructuras Críticas, Tiempo Real, Smart Grids, IEC 62351**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

La nueva revolución industrial, acuñada como Industria 4.0, está transformando este sector e impulsándolo hacia un enorme progreso centrado en el tratamiento de la información para la extracción de conocimiento y valor de los procesos industriales. Como eje de esta transformación, las comunicaciones se convierten en la columna vertebral que hace posible generar un mejor y mayor conocimiento de dichos procesos. Además, la interconexión de los sistemas y dispositivos permite mejorar la automatización de procesos, así como el control y la gestión de los mismos, permitiendo incrementar la visión y conocimiento de estos procesos en tiempo real para una mejor toma de decisiones.

Numerosas Infraestructuras Críticas (IC) nacionales, tales como el sistema eléctrico o de transporte son consideradas como parte de este sector industrial y, por lo tanto, también están inmersas en dicha transformación. Las ventajas son innumerables, permitiendo tener una visión completa de la infraestructura y su estado, y pudiendo actuar sobre la misma con un control más efectivo con el objetivo de lograr un rendimiento más eficiente.

Sin embargo, todas estas ventajas no vienen sin contraprestaciones, el hecho de interconectar los dispositivos y sistemas industriales hace que se incremente su exposición hacia posibles atacantes, que ya no tienen que estar físicamente en la infraestructura para provocar su mal funcionamiento, sino que remotamente pueden orquestar un ciber-ataque masivo afectando a un conjunto de instalaciones consiguiendo un mayor impacto y logrando su escalado a un coste menor.

II. Problemática de los Sistemas de Control Industriales (ICS)

Tradicionalmente, de forma similar al sector eléctrico, los Sistemas de Control Industriales (Industrial Control Systems, ICS) estaban aislados y usaban protocolos propietarios con hardware y software especializado. Esto hizo que la seguridad en estos sistemas se enfocase principalmente a la seguridad física y la prevención de riesgos, obviando completamente la seguridad lógica de los sistemas y las comunicaciones.

Actualmente, la amplia disponibilidad y el bajo coste de los dispositivos IP (Internet Protocol) está reemplazando a las soluciones propietarias, lo que por otro lado incrementa su exposición ante incidentes y vulnerabilidades frente a los ciber-ataques. Esta tendencia hace que, en cierta forma, la diferencia entre ICS y las tecnologías de información (IT) se estén difuminando. Esta evolución permitiría dotar a los entornos ICS con tecnología de seguridad IT madura para cubrir rápidamente esta carencia. Sin embargo, los primeros pasos dados en ese sentido han puesto en evidencia que es necesario tener en cuenta la especificidad de los entornos ICS y de los requisitos que imponen (e.g., tiempo real, rendimiento, disponibilidad, respuesta crítica en tiempo), lo que obliga a ofrecer soluciones de seguridad adaptadas a estos entornos. En líneas generales, los ICS son críticos en tiempo y no tanto en capacidad de ancho de banda. Según el nivel de criticidad de los mismos, pueden requerir una estricta operativa de tiempo real, exigiendo por tanto un comportamiento determinista de las redes de comunicación difícilmente asegurable con los sistemas IT, orientados a ofrecer throughput alto y con exigencias más laxas en cuanto al cumplimento de tiempos en términos de delay y jitter.

Otro de los aspectos a tener en cuenta, son las consecuencias que un ciber-ataque puede llegar a tener en un entorno ICS, la parada de un sistema en producción o incluso la integridad física de los operarios se puede llegar a ver comprometida al tener capacidad de impacto en el mundo físico. Esta problemática se agrava en una Infraestructura Crítica (e.g. Smart Grid), siendo por su propia naturaleza un foco más atractivo para los posibles atacantes y sus

consecuencias devastadoras. Numerosos han sido ya los ataques a este tipo de infraestructuras (e.g., sistema eléctrico de Ucrania, Israel o Turquía, sistema de aguas de Illinois, aeropuerto de Varsovia, puerto de Maasvlakte) y las previsiones no son nada tranquilizadoras.

III. Securización de las Infraestructuras Críticas con Requisitos de Tiempo Real

El sector eléctrico, que aglutina gran parte de las infraestructuras críticas nacionales (sistemas de generación, transporte y distribución de energía), es una industria fuertemente regulada y estandarizada. En este sentido, el organismo de estandarización internacional IEC que desarrolla la gran parte de los estándares para la industria ha publicado el IEC 62351 [1], definiendo detalladas propuestas de protección en las comunicaciones para el sector. Estas definiciones cubren aspectos específicos tales como la protección de las comunicaciones con requisitos de tiempo real y propuestas avanzadas para la autenticación de equipos y la distribución de claves.

La plataforma desarrollada por SoC-e y Keynetic resuelve la problemática de la securización de tráfico con requisitos estrictos de tiempo real mediante un procesamiento wire-speed hardware (en FPGA) de las tramas de comunicación. Tanto la autenticación automática de los equipos, como el intercambio ágil de claves se realiza mediante una implementación hardware/software ad-hoc de los sistemas involucrados (TPM, PKI, KDC y GDOI) acorde a la solución demandada por el sector eléctrico. Los dos componentes principales son un Servidor IEC 62351-9 [2] y un equipo de comunicación en campo, Plataforma RELY-RB-SAFE.

El Servidor IEC 62351-9 se encarga de la gestión de claves criptográficas (generación, distribución y revocación), así como de la gestión de certificados digitales para la protección de las comunicaciones de la Smart Grid. Por lo tanto, se abarca tanto la gestión de claves asimétricas (para la autenticación y seguridad en la distribución de claves) como simétricas para claves de grupo [3] usadas para las comunicaciones industriales con requisitos de tiempo real.

La Plataforma RELY-RB-SAFE (IEC 62351-6 [4]) se encarga de securizar las comunicaciones de la Smart Grid. Por un lado, este equipo actúa como cliente IEC 62351-9, autenticándose en la red de forma segura utilizando mecanismos hardware y software que le permiten gestionar con el servidor de claves la obtención de claves criptográficas utilizadas para la securización de las comunicaciones propias de la Smart Grid. Por otro lado, apoyándose en un motor criptográfico de desarrollo propio de acuerdo al estándar AES-GCM, el equipo es capaz de securizar (cifrar y autenticar) las comunicaciones críticas de la Smart Grid de forma transparente a los propios equipos de la red que generan y reciben dichas comunicaciones. Así, el equipo se encarga de filtrar el tráfico deseado y realizar las modificaciones necesarias para proporcionar comunicaciones seguras de acuerdo al estándar IEC 62351-6. De esta forma, se consigue proporcionar seguridad a la Smart Grid y a sus equipos sin necesidad de sustituir los equipos ya presentes.

IV. Caso de Uso: Smart Grids

La Figura 1 muestra la arquitectura de un armario de Subestación Digital IEC 61850 para Smart Grids. Cada celda

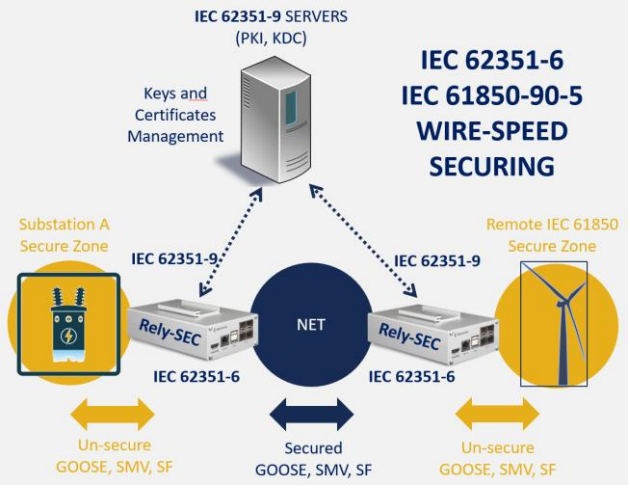


Fig. 1. Caso de uso en Smart Grids: Securización de mensajes GOOSE y SMV en Subestación IEC 61850.

tiene una unidad de comunicaciones dedicada, agrupadas en racks y sincronizadas por IEEE1588 con una precisión inferior al microsegundo.

En la Subestación se introducen mecanismos de ciberseguridad avanzados para securizar los mensajes de control con requisitos estrictos de tiempo real. Estos mensajes son GOOSE y valores de medidas digitalizadas (SMV) que se intercambian entre los diferentes dispositivos, principalmente las Merging Units y los Relays IEDs.

El dispositivo Rely-SEC (desarrollado for SoC-e) es capaz de securizar, mediante cifrado y autenticación, todo el tráfico intercambiado a velocidad de línea. La latencia introducida por este equipo está en el rango de microsegundos. De esta forma, la operativa en tiempo real no se ve afectada. Por un lado, el IEC 62351-6 define el formato de estos paquetes seguros intercambiados, mientras que el IEC 62351-9 establece como se realiza la gestión de las claves de seguridad necesarias (generación, actualización y distribución).

V. Conclusiones

Las infraestructuras críticas no fueron diseñadas con la ciberseguridad como requisito en su diseño, pero para poder seguir operándolas es fundamental protegerlas apoyándose en las normas IEC 62351. Por su parte, las Smart Grids además imponen requisitos estrictos de tiempo real, lo que hace necesario la implementación del plano de datos (IEC 62351-6) en hardware específico para cumplir dichas restricciones. El plano de control (IEC 62351-9) no tiene esa criticidad en tiempo, por lo que se aísla para evitar interferir. Como resultado, se consigue una plataforma estándar IEC que permite securizar las comunicaciones en el sector eléctrico (Smart Grids), en concreto dentro de la Subestación.

Referencias

[1] "Power systems management and associated information exchange - Data and communications security", IEC 62351, 2018.

[2] "Power systems management and associated information exchange - Data and communications security - Part 9: Cyber security key management for power system equipment", IEC 62351-9, 2017.

[3] B. Weis, et al.: "Group Domain of Interpretation (GDOI) Protocol Support for IEC 62351 Security Services", RFC 8052, 2017.

[4] "Power systems management and associated information exchange - Data and communications security - Part 6: Security for IEC 61850", IEC 62351-6, 2007.

Hardening de redes industriales con FlowNAC

Jon Matias, Jokin Garay
Keynetic Technologies
Bilbao, España
{jon.matias, jokin.garay}@keynetic.es

Javier Benito, Beñat Uriarte, Koldo Santisteban
Batz S. Coop.
Igorre, España
{JBenito, BUriarte, KSantisteban}@batz.com

Resumen- **El proceso de digitalización actual del sector industrial hacia la Industria 4.0 plantea nuevos riesgos y amenazas a nivel de ciberseguridad, ya que máquinas tradicionalmente aisladas se empiezan a conectar a las redes IT. Para que la producción no se vea afectada es importante proteger los activos críticos de las fábricas. Enmarcado en el programa BIND4.0, Batz decide mejorar la protección de su red industrial con el objetivo de tener un mayor control sobre los activos conectados y limitar el tráfico lateral. Como respuesta a esa necesidad, Keynetic ha iniciado la implantación de FlowNAC, una solución de seguridad de red apoyada en tecnología SDN, que implementa un modelo de confianza cero con micro-segmentación, lo que permite el hardening de la red industrial.**

Index Terms- Ciberseguridad, Control de Acceso a Red, Hardening de Red, Confianza Cero, Microsegmentación

Tipo de contribución: *Investigación en desarrollo*

I. Problemática de la Seguridad En Redes Industriales

El sector industrial está actualmente inmerso en un proceso de evolución hacia la industria conectada, también conocida como Industria 4.0. La irrupción de las comunicaciones posibilita la innovación de los procesos, proporcionando numerosas ventajas y oportunidades, como el mantenimiento predictivo, el Big Data aplicado a los procesos industriales o la teleasistencia industrial.

Tradicionalmente las redes industriales han estado aisladas, pero el proceso de digitalización ha supuesto que las máquinas se expongan a nuevos riesgos y amenazas, incrementando de esta forma la superficie de ataque expuesta por la industria.

Además, es importante tener en cuenta la diferencia en la ciberseguridad en el mundo IT y el OT, ya que en el mundo OT prevalece el rendimiento y la disponibilidad frente a otros aspectos como la confidencialidad y la integridad, debido a que una parada de la producción impacta directamente en la cuenta de resultados. Por otro lado, existe una gran heterogeneidad de equipamiento, en cuanto a tipología de máquinas, su grado de actualización frente a vulnerabilidades y la coexistencia generacional.

Por estos motivos no siempre es posible aplicar las mismas técnicas para la mitigación de amenazas del mundo IT en la industria.

II. Marco del Proyecto BIND4.0

El programa BIND4.0 (https://bind40.com) tiene como objetivo conectar a grandes empresas tractoras del sector industrial vasco con startups tecnológicas que puedan ofrecer soluciones innovadoras.

En el contexto de este programa, el presente proyecto se orienta a la mejora de la protección de la red industrial de la compañía Batz (http://www.batz.com). No se parte de cero, sino que se construye sobre la capacidad de protección de los activos con acceso a Internet, la segmentación de la red industrial o la capacidad de inspección de tráfico que se han implantado previamente.

Con el compromiso de mejorar la seguridad de la planta, y como consecuencia de una auditoria externa de seguridad, se plantean los siguientes objetivos para el proyecto:

- El **control del tráfico lateral** dentro de la planta industrial. Se considera fundamental limitar este tráfico para evitar tanto accesos no permitidos como la posible propagación incontrolada de malware.
- La **visualización** en todo momento de los dispositivos conectados a la red. Esto permitirá tener un mejor conocimiento de los activos industriales y su estado (e.g., autorizado, cuarentena, bloqueado).
- La **gestión simplificada** de la seguridad de la red industrial. De esta forma los administradores podrán definir y desplegar las políticas que controlan el acceso de los dispositivos, así como su adaptación a las necesidades específicas derivadas de los procesos industriales y el mantenimiento de las máquinas.
- Disponer de una serie de **alertas** que reflejen los estados y situaciones más críticas y que necesiten de una intervención por parte de los administradores.

En este sentido, cabe destacar la relevancia de la correcta gestión del acceso por parte de terceros a la infraestructura para diferentes procesos, como el mantenimiento predictivo de las máquinas, el Big Data en SaaS o el acceso bajo demanda para teleasistencia. Limitar dicho acceso únicamente a los recursos involucrados, así como en la dimensión temporal es crucial para proteger adecuadamente la red industrial.

Como respuesta a esta necesidad, Keynetic (https://keynetic.tech) ofrece una solución que permite controlar de una forma más específica los activos/máquinas y restringir el uso de la red (hardening de red) para minimizar la superficie de ataque expuesta por la planta industrial.

El proyecto se plantea en dos fases, una primera que se centra en la planta en la que se aborda el control en todo momento de quién se conecta a la red industrial y para qué, limitando el tráfico lateral. En una segunda fase, se trata de extender esta misma solución a la red de oficinas.

III. Securización de las Redes Industriales con FlowNAC

Las recomendaciones actuales para la securización de las redes industriales se basan principalmente en las técnicas de

segmentación y defensa en profundidad. Como referencia las normas del IEC 62443 [1], que continuó con el trabajo de la ISA99 [2] que definió las zonas de seguridad y conductos, y la NIST800-82r2 [3] son las más relevantes para la seguridad de los sistemas de control industriales (ICS).

La segmentación se basa por lo tanto en la definición de zonas o segmentos confiables, por lo que, si un dispositivo malicioso o uno previamente confiable se ve comprometido, todo el segmento cae al no haber protección dentro del segmento. La debilidad de este modelo se puso de manifiesto el 12 de mayo de 2017 con WannaCry, por el cual una vez infectado uno de los equipos del segmento, la propagación fue inmediata debido a la falta de protección interna.

Este modelo de confianza ha evolucionado hacia modelos de confianza cero (Zero-Trust), en los que la red interna no se considera más segura que un hotspot público. Estas tesis se ven respaldadas por los planteamientos de proyectos como BeyondCorp [4] en Google. Además, la microsegmentación permite definir específicamente los servicios a los que se tendrá acceso, tal y como se hace en el Data Center con soluciones como VMware NSX [5]. Para ello se hace uso de tecnología NAC (Control de Acceso a Red), lo que permite tanto visualizar quién se conecta a la red, como definir políticas de seguridad que gobiernan a qué servicios se puede conectar.

Con el objetivo de securizar la red industrial de Batz, Keynetic ha iniciado un proyecto para la implantación de FlowNAC [6, 7] en el contexto del programa BIND4.0.

FlowNAC supone un cambio de paradigma apoyado en la tecnología SDN [8] (Redes Definidas por Software) para el control de acceso a red. FlowNAC implementa un modelo de confianza cero, apoyándose en la microsegmentación para controlar de forma específica qué dispositivos pueden acceder a qué servicios (modelo de lista blanca), **limitando el tráfico lateral**. De esta forma, tanto si un dispositivo malicioso como un tráfico no permitido tratan de acceder a la red, se les cortará dicho acceso, protegiendo de esta forma las máquinas conectadas.

FlowNAC permite dotar de la **visibilidad** completa de los dispositivos conectados y el control sobre lo que pueden hacer en la red. El objetivo de este control es el hardening de la red, limitando el movimiento lateral dentro del segmento y restringiendo de esta forma la superficie de ataque expuesta por la red industrial.

Para la **gestión simplificada** de este control, se definen políticas de seguridad en lógica de negocio, lo que permite adaptar el comportamiento de la red a los procesos industriales. La apropiada agrupación tanto de usuarios/dispositivos y servicios como de las zonas y segmentos, permitirá que la gestión de las políticas sea escalable y manejable por los administradores del sistema.

La tecnología SDN permite además que se pueda analizar bajo demanda todo el tráfico que no cumple con las políticas definidas para cada dispositivo y caracterizarlo. Este análisis permitirá generar **alertas** y determinar si un dispositivo se ha visto comprometido.

IV. Caso de Uso: Red Industrial de Batz

El caso de uso que se presenta en la Figura 1 es representativo del tipo de control que se quiere conseguir en la red industrial. Por un lado, se gestionan dispositivos (e.g. máquinas, PLCs), grupos de usuarios (e.g. ingeniería, mantenimiento) y servicios (e.g. teleasistencia, mantenimiento predictivo), y por otro lado, se definen políticas de conexión autorizadas. Finalmente, FlowNAC se encarga de implementar el control del tráfico a nivel de flujo acorde con dichas políticas, así como de visualizar su estado.

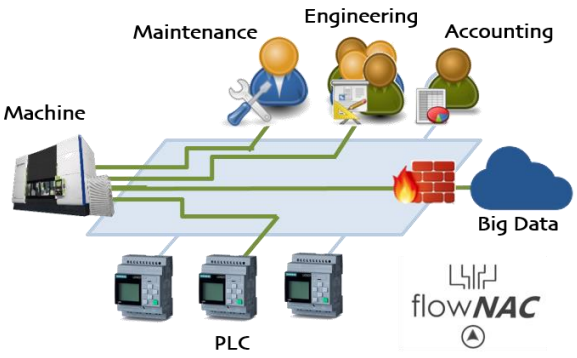


Fig. 1. Caso de uso BIND4.0: Hardening de la red industrial de Batz con FlowNAC.

V. Conclusiones

El proyecto que persigue los objetivos descritos en la sección II, actualmente se encuentra en proceso de implantación enfocado a la protección de la red industrial y los procesos productivos. FlowNAC se adapta perfectamente a las necesidades de securización planteadas, permitiendo tanto visualizar como controlar los dispositivos que acceden a la red sin afectar a la producción. Siendo este último aspecto clave para el éxito de la implantación, ya que en ningún caso los procesos productivos tienen que verse afectados. Además, se ofrece un cuadro de mando que permite definir las políticas de seguridad adaptadas a los procesos industriales.

Agradecimientos

Este trabajo se enmarca en la segunda edición del programa BIND4.0 de aceleración de startups orientada al entorno industrial.

Referencias

[1] "The 62443 Series of Standards: Industrial Automation and Control Systems Security", *International Society of Automation (ISA)*, 2018. http://isa99.isa.org/Public/Information/The-62443-Series-Overview.pdf

[2] "ISA99, Industrial Automation and Control Systems Security", *International Society of Automation (ISA)*. Retrieved 2018 from https://www.isa.org/ISA99/

[3] K. Stouffer, S. Lightman, V. Pillitteri, M. Abrams, A. Hahn: "Guide to Industrial Control Systems (ICS) Security", *NIST SP 800-82 Rev. 2*, May 2015, https://csrc.nist.gov/publications/detail/sp/800-82/rev-2/final

[4] R. Ward, B. Beyer: "Beyondcorp: A new approach to enterprise security", ;*login:*, 2014, vol. 39, no 6, p. 6-11.

[5] W. Holmes: "VMware NSX Micro-segmentation – Day 1", VMware Press, 2017.

[6] J. Matias, J. Garay, et al.: "FlowNAC: Flow-based network access control.", *Third European Workshop on Software Defined Networks (EWSDN)*, IEEE, 2014.

[7] J. Matias, J. Garay, et al.: "FlowNAC: Flexible and granular Network Access Control based on SDN and NFV technologies", *II Jornadas Nacionales de Investigación en Ciberseguridad (JNIC2016)*, 2016.

[8] J. Matias, J. Garay, et al.: "Toward an SDN-enabled NFV architecture", *IEEE Communications Magazine*, vol. 53, no 4, p. 187-193, 2015.

Un resumen de: UGR'16: Un nuevo conjunto de datos para la evaluación de IDS de red basados en cicloestacionariedad

G. Maciá-Fernández[1], J. Camacho[1], R. Magán-Carrión[1], M. Fuentes-García[1], P. García-Teodoro[1], R. Therón[2]
[1]Universidad de Granada - *Network Engineering & Security Group* - CITIC
{gmacia,josecamacho,rmagan,nmfuentes,pgteodor}@ugr.es
[2]Universidad de Salamanca
theron@usal.es

***Resumen*—La evaluación de algoritmos y técnicas para implementar sistemas de detección de intrusiones depende en gran medida de la existencia de conjuntos de datos (*dataset*) bien diseñados. En los últimos años, se ha realizado un gran esfuerzo para construir estos *datasets*. En este trabajo se presenta un nuevo *dataset* que se construye a partir de tráfico real y donde se realizan ataques actualizados. La principal ventaja de este conjunto de datos sobre otros previos es su utilidad para la evaluación de IDSs donde se considera la evolución a largo plazo y la cicloestacionariedad del tráfico. También permite entrenar y evaluar modelos que contemplen las diferencias entre día/noche o entre días laborables/fines de semana.**

***Index Terms*—Seguridad en redes, dataset, IDS, tráfico de red, netflow**

Tipo de contribución: *Investigación previamente publicada en JITEL'17 (Español) y Computers & Security 73 (2018)*

I. Introducción

Los sistemas de detección de intrusiones (IDS) aparecieron en la esfera de la seguridad como una solución al problema de identificar actividades maliciosas en redes y sistemas. En pocas palabras, un IDS consta de un módulo encargado de la obtención de datos, un módulo de pre-procesamiento que adapta esos datos para los siguientes pasos en el sistema, y un módulo de decisión capaz de determinar si un evento debe ser considerado malicioso o no.

Un problema esencial cuando se evalúan las capacidades de los IDS es la necesidad de un conjunto de datos representativo que permita la comparación entre distintas propuestas. A pesar del gran número de esfuerzos realizados para obtener un conjunto de datos para la correcta evaluación de IDS es posible constatar que, hasta el momento, todas ellas son soluciones parciales. En una primera revisión, se puede comprobar que muchos de los conjuntos recientes carecen de tráfico real o estrategias de ataque actualizadas. Otra limitación importante está relacionada con la duración de las capturas de datos. Esto es, para hacer posible la evaluación de algoritmos de detección que consideran la evolución cicloestacionaria del tráfico, es decir, las diferencias en el tráfico entre día/noche o laborables/festivos, se necesita una traza de larga duración.

En este trabajo, se describe un nuevo *dataset* (UGR'16) que contiene trazas reales de *netflow* anonimizadas capturadas en un ISP Tier-3 durante 4 meses. En este conjunto, se han incluido escenarios de ataque realistas y se ha llevado a cabo el etiquetado el tráfico. Todos los datos están disponibles públicamente en: https://nesg.ugr.es/nesg-ugr16/.

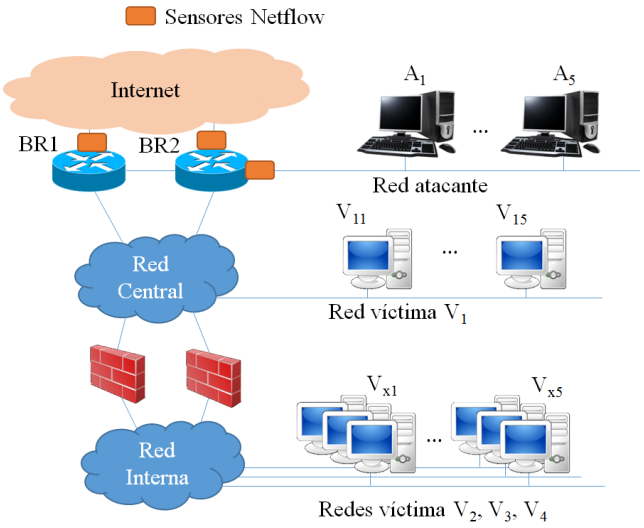


Figura 1. Topología de la red.

II. Generación del conjunto de datos

Los datos se obtienen de una red real de un ISP Tier-3. El ISP es un proveedor de servicios en la nube, por lo que algunos de los servicios típicos implementados en la red están virtualizados. Algunos de los servicios típicos de alojamiento que se encuentran son webs con configuraciones propietarias o estándares, correo electrónico, servidores FTP y DNS, etc. Esta red se utiliza por muchas compañías que tienen tamaños dispares y que se centran en una gran variedad de mercados. Se garantiza así que el tráfico que atraviesa la red sea muy heterogéneo, pues incluye tanto accesos de clientes a Internet como recepción de tráfico por servidores típicos.

La topología esquemática de la red del ISP y la infraestructura usada para la recolección de datos se muestran en la Fig. 1. Se observan:

- Dos encaminadores frontera redundantes, *BR1* y *BR2*, donde se ubican los *sensores de netflow* que permiten el registro de todas las conexiones.

- Una *red de atacantes* con 5 máquinas A_1-A_5, para la generación de ataques controlados.
- Una *red de víctimas* V_1 con 5 máquinas víctima que se utilizan para el registro de los ataques efectuados contra ellas.
- Tres *redes víctima* adicionales con 15 máquinas en total ubicadas en la red protegida por cortafuegos.

Generación de ataques sintéticos. Sobre esta topología se planifica la generación de ataques desde las máquinas atacantes hacia las máquinas víctima. Los ataques están controlados para permitir su etiquetado e identificación posterior por parte de los IDSs. Los tipos de ataque ejecutados son:

- *DoS de baja tasa*: Se envían paquetes TCP SYN a las víctimas utilizando la herramienta `hping3`. Se utilizan diversas variantes de ataque.
- *Escaneo de puertos*: Se ejecuta un escaneo SYN continuo a los puertos comunes de las víctimas durante 3 minutos, utilizando la herramienta `nmap`. También se utilizan diversas variantes.
- *Tráfico de Botnet*: Se incorpora tráfico de *botnet* a la traza con el objeto de incorporar tráfico actualizado de ataques. Debido a las restricciones éticas que supone la instalación de una *botnet* en una red en producción se procede a tomar el tráfico generado por la *botnet* Neris del dataset CTU-13 [1], y se modifican las direcciones IP y *timestamps* para que coincida con la temporización del tráfico generado en el conjunto de datos UGR'16.

El tráfico de ataque se genera en lotes de 2 horas. En cada lote de ataque se ejecutan todas las variantes de ataque. Dado que el tráfico capturado por los sensores incluirá instancias de tráfico relacionado tanto con ataques como con tráfico normal se lanzan lotes de ataque durante 12 días consecutivos, cambiando la hora de inicio para así cubrir todas las horas posibles del día, permitiendo así el estudio del tráfico de *background* para distintas horas del día junto con tráfico de ataque.

Con esta metodología se obtienen dos conjuntos de datos; uno para *calibración* de modelos, obtenido durante el periodo marzo-2016 a junio-2016 en el que no se incluyen ataques sintéticos, y el otro para *prueba*, obtenido principalmente durante agosto-2016, donde se incluyen los ataques sintéticos anteriormente descritos.

III. Análisis del conjunto de datos

Tras la generación del conjunto de datos se ha realizado un análisis de los mismos con el fin de estudiar su aplicabilidad a la evaluación de IDSs basados en cicloestacionariedad. Para ello, se han realizado las siguientes tareas:

- Extracción de estadísticas sobre los flujos y su evolución con el tiempo. En la Fig. 2 se puede observar el número de flujos por aplicación para el conjunto de prueba.
- Detección de anomalías con detectores de referencia del estado del arte. Se han utilizado dos variantes de MSNM [2] y OCSVM [3][4] como tercer detector. Se destaca que se han detectado anomalías que se han confirmado manualmente como ataques. En la Fig. 2 están representadas como puntos rojos. En concreto, se ha detectado una campaña de *spam*, ataques de escaneo SSH y ataques de escaneo UDP.

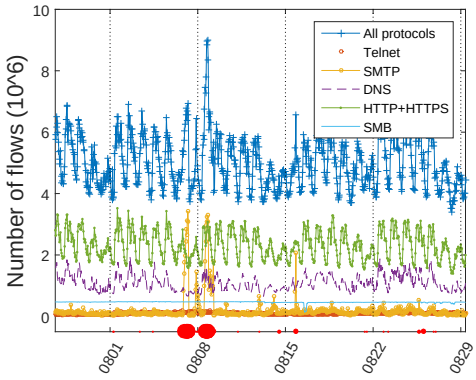


Figura 2. Evolución del número de flujos para el conjunto de prueba, y anomalías detectadas (puntos rojos en la base de la Figura).

- Etiquetado de los flujos. Se han etiquetado los diferentes flujos según el tipo de ataque sintético o detectado mediante análisis de anomalías. El resto de tráfico se ha etiquetado como *background*, dado que a priori no se puede afirmar con toda certeza si es normal o no.

IV. Conclusiones

La principal contribución de este trabajo es la generación de un nuevo *dataset* para la evaluación de algoritmos y sistemas IDS llamado UGR'16. UGR'16 es una colección de trazas *netflow* capturadas durante más de 4 meses de tráfico en una red real de un ISP Tier-3, junto con un conjunto de ataques de red real que se ha diseñado específicamente para entrenar y probar algoritmos IDS.

Las principales ventajas del *dataset* presentado frente a otros ya existentes son: *i)* el tráfico de *background* es muy representativo del tráfico actual de Internet, pues se captura mediante sensores en una red ISP donde existen perfiles muy diferentes de clientes; *ii)* la duración del *dataset* lo hace adecuado para probar algoritmos que consideran la evolución ciclo-estacionaria del tráfico en día/noche y días laborables/fines de semana.

El *dataset* está disponible públicamente en: https://nesg.ugr.es/nesg-ugr16/.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Gobierno Español-MINECO (Ministerio de Economía y Competitividad) y fondos FEDER, a través de los proyectos TIN2014-60346-R y TIN2017-83494-R.

Referencias

[1] CTU-13 dataset. [Online]. Available: https://stratosphereips.org/category/dataset.html

[2] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, and G. Maciá-Fernández, "PCA-based multivariate statistical network monitoring for anomaly detection," *Computers & Security*, vol. 59, pp. 118–137, 2016.

[3] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the Support of a High-Dimensional Distribution," *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[4] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, "Network Anomaly Detection: Methods, Systems and Tools," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, 2014.

Metodología para la generación de conjuntos de datos de ataques basados en URI de HTTP

Jesús Díaz-Verdejo
Universidad de Granada
ETSITT - CITIC
jedv@ugr.es

Rafael Estepa
Universidad de Sevilla
ETS Ingenieros. 41092 Sevilla
rafaestepa@us.es

Antonio Estepa
Universidad de Sevilla
ETS Ingenieros 41092 Sevilla
aestepa@us.es

Germán Madinabeita
Universidad de Sevilla
ETS Ingenieros 41092 Sevilla
german@trajano.us.es

Daniel Rodríguez
Universidad de Sevilla
ETS Ingenieros 41092 Sevilla
danrodleo@alum.us.es

Resumen- **El desarrollo de sistemas de detección de intrusiones basadas en web, o de firewalls de aplicación web, requiere el uso de conjuntos de datos (*datasets*) apropiados para el entrenamiento y evaluación. Una elección inadecuada de los mismos resultará en sesgos e imprecisiones que pueden invalidar la experimentación y, consecuentemente, la evaluación de las capacidades de detección de la/s técnica/s analizada/s. El problema es especialmente relevante en el caso de los sistemas basados en anomalías, ya que se requiere disponer de ataques adecuados al entorno de experimentación.**

En el presente trabajo se propone una metodología para la generación de *datasets* adaptados a las necesidades de la experimentación y del escenario de uso, mediante el uso de la combinación y parametrización de diferentes fuentes de ataques. Además, se ha implementado una herramienta que sigue la metodología propuesta, generando dos *datasets* con 800 y 1.100 instancias de ataque respectivamente, que responden a las necesidades de la experimentación particular de un sistema de detección de anomalías en peticiones HTTP. No obstante, la metodología desarrollada es suficientemente genérica para permitir la generación de *datasets* adecuados al desarrollo de otros sistemas en función de las necesidades del usuario.

Index Terms- **ciberseguridad, dataset ataques, sistemas de detección de intrusiones, detección de anomalías**

Tipo de contribución: *Investigación original*

I. INTRODUCCIÓN

Los conjuntos de datos (*datasets*) son ampliamente utilizados en el ámbito de la ciberseguridad como parte, entre otros, del método de entrenamiento y evaluación de los algoritmos de detección de intrusiones (IDS) [1]. Las herramientas empleadas en estos sistemas de detección (p.ej. redes neuronales, cadenas de Markov, etc.) habitualmente incorporan gran cantidad de parámetros libres que son ajustados gracias a *datasets* validados y relacionados con el aspecto a modelar. El contenido de estos *datasets* incluye típicamente el tráfico generado por algún tipo de ciberataque o el tráfico en ausencia de ataques (denominado habitualmente tráfico normal o de *background*). Una vez fijados los parámetros libres del sistema en la fase de entrenamiento, se utilizará un nuevo *dataset*, disjunto del usado en el entrenamiento, para validar lo certero del sistema ya entrenado (i.e. fase de validación [1]). En cualquier caso, el rendimiento final del IDS propuesto dependerá, en gran medida, de la calidad de los *datasets* empleados en ambas fases: su representatividad del tráfico real, su actualidad, su volumen, etc., por lo que la obtención de *datasets* de calidad presenta una importancia capital en el desarrollo de los sistemas de detección de intrusiones. En [22] se analizan las características que debería tener un buen *dataset* y las buenas prácticas en la experimentación realizada con ellos. Un buen *dataset* debe permitir, además, repetir experimentos, validar y comparar aproximaciones y características del modelo propuesto. Sin embargo, en nuestra opinión, gran parte de los utilizados en la actualidad son privados, sufren de una excesiva tara en su carga útil a fin de anonimizarlos o resultan sesgados en su adquisición.

Nuestro trabajo surge a raíz de la necesidad de obtener *datasets* que pudieran ser utilizados en el contexto del desarrollo de un sistema basado en modelos de Markov del tipo *stochastic segmental modelling* (SSM) para la detección de anomalías en las peticiones (URI) del protocolo HTTP [2]. El sistema en desarrollo establecía un modelo específico para cada servicio web a proteger, para lo que necesitaba ser entrenado mediante un *dataset* con tráfico real libre de ataques recibido por dicho servidor. Además, el modelo empleado calcula la frecuencia de aparición de las palabras del vocabulario encontrado durante la fase de entrenamiento, por lo que era necesario disponer en el *dataset* de entrenamiento de un volumen elevado de peticiones para que el entrenamiento fuese representativo. Tras la fase de entrenamiento, es posible evaluar el grado de anomalía de una nueva URI respecto al tráfico normal. Si dicho grado de anomalía supera cierto umbral, a determinar también experimentalmente, la URI analizada puede considerarse un ataque. Por lo tanto, en el contexto de este artículo, un ataque se referirá a una URI.

A fin de determinar la bondad del sistema SSM desarrollado, es necesario confrontar los resultados con un conjunto disjunto de tráfico normal (aquellas URI detectadas como ataques en este conjunto serán falsos positivos, FP), así como con un *dataset* con tráfico de ataques (aquellas URI maliciosas no detectadas en este conjunto se corresponderán con los falsos negativos, FN). Dado que el *dataset* con tráfico normal puede ser obtenido del tráfico entrante al servicio web (filtrado de posibles ataques), el problema al que nos enfrentábamos consistía en obtener un conjunto de ataques fiable y con un volumen tal que permitiera obtener resultados significativos durante la evaluación del sistema.

Aunque existen algunos *datasets* tanto con tráfico normal como de ataques disponibles para la comunidad científica,

estos adolecen de múltiples problemas que limitan su utilidad, como los identificados en [22] sobre corrección, transparencia o realismo. Además, como se describirá en el Apartado II, estos *datasets* resultan inadecuados para la correcta evaluación en nuestro contexto debido a que sólo estamos interesados en las URI de los métodos GET y POST de HTTP que recibe un servidor web particular.

Este trabajo describe e implementa una metodología para la creación de conjuntos de datos de ataques con URI cuyas características (p.ej. fecha de aparición del ataque, criticidad, vulnerabilidad asociada, etc.) puedan ser definidas por el usuario, y que resuelve las limitaciones de los *datasets* existentes para nuestro escenario. No obstante, la metodología propuesta es suficientemente genérica para inspirar y ser adaptada para crear *datasets* utilizables con otros tipos de ataques. Por último, el sistema desarrollado puede ampliarse fácilmente en el futuro, ya que puede ser alimentado de forma distribuida.

El resto del artículo se estructura como sigue. En el Apartado II se describen trabajos relacionados analizando algunos de los *datasets* existentes, así como las metodologías propuestas para el desarrollo de los mismos. En el Apartado III se introduce la propuesta del presente trabajo. En el Apartado IV se describen las fuentes de información utilizadas para la obtención de ataques, su parametrización y tipología. En el Apartado V se presenta la arquitectura y la implementación del sistema desarrollado para la generación de *datasets*. En el Apartado VI se explican los tipos de ataques incorporados al sistema y algunos resultados en forma de *datasets* de ataques. Finalmente, en el Apartado VII se presentan las conclusiones y líneas de continuación del presente trabajo.

II. Antecedentes

Es posible encontrar varios *datasets* orientados a la seguridad con tráfico normal (*background*) y de ataques, disponibles para la comunidad científica. Sin pretender ser exhaustivos, uno de los más conocidos y usados, a pesar de sus grandes limitaciones [3], es el denominado KDD99 [4]. Este incluye registros de conexiones parametrizadas y etiquetadas como normal o ataque, construido sobre la base de datos DARPA98 [5]. Los ataques que presenta pueden agruparse en 4 categorías: *DoS, Remote to Local, User to Root* y *Probe*, y el tráfico de fondo es simulado. Una variante de esta es la denominada NSL-KDD, que consiste en una selección de registros de KDD que mejora el conjunto de entrenamiento y test. Otro *dataset* disponible es IDEVAL2000 [6], en el que se simularon 2 escenarios de ataque DDoS que se ejecutaron sobre distintos escenarios de red para conseguir variabilidad. Por su parte, DEFCON [7] contiene tráfico capturado durante un ejercicio de *Capture the Flag*, en el que los participantes podían ser atacantes o defensores. Contiene fundamentalmente tráfico de intrusiones, por lo que adolece de falta de tráfico limpio de *'background'*.

Aunque no primariamente orientada a la ciberseguridad, CAIDA [8] recopila numerosas capturas de tráfico real. Entre ellas se incorporan algunas que incluyen ataques, aunque resultan de tamaño muy limitado. Por ejemplo, se incluye 1 hora de tráfico anonimizado con ataques DDoS a víctimas y las respuestas de estas. Sin embargo, la mayoría de las trazas disponibles carecen de carga útil, lo que limita su usabilidad. Análogamente, en LBNL [9] podemos encontrar trazas de tráfico interno de empresas que incluye cabecera, pero sin carga útil y que se encuentra anonimizado. El tráfico de ataque disponible consiste básicamente en escaneos TCP.

También existen *datasets* basados en recolección de tráfico de red durante varios días en un *testbed* (habitualmente en un campus). En esta categoría entrarían, por ejemplo: *UNIBS Dataset* [10], capturado en el *router* del campus de la Universidad de Brescia (Italia) durante 3 días consecutivos; *ISCX-ISD Dataset* [11], donde, sobre un *testbed* real, se incorporaron diversos perfiles de ataques multietapa y tráfico de fondo, elaborados en base a características de tráfico real registrado en el *testbed*; y *Kyoto2006+* [12], recogido en la Universidad de Kyoto mediante *honeypots*.

Aunque los *dataset* descritos anteriormente están disponibles para la comunidad científica, estos adolecen, entre otros problemas, de no representar con fidelidad las características del tráfico real. Además, la mayoría no se encuentran correctamente etiquetados y/o no incluyen información sobre el ataque (KDD99, UNIBS, LBNL). Adicionalmente, el proceso de anonimización impide el uso de datos relevantes para los investigadores (CAIDA, DARPA2000). El resultado es que ninguna de estas bases de datos resulta adecuada para la evaluación en nuestro contexto.

Por otra parte, dado que los comportamientos del tráfico de usuario y del tráfico de ataques van evolucionando a lo largo del tiempo, es preciso cambiar de usar *dataset* estáticos a *dataset* dinámicos que permitan, no solo reflejar la evolución en la composición del tráfico de intrusiones, sino también ser recompuestos, parametrizados y extendidos. En [13] se propone una metodología para conseguir tales objetivos. Aunque en dicho trabajo se definen perfiles para describir escenarios de ataques, la contribución principal se centra en describir perfiles para tráfico de usuario de distintas aplicaciones, como http, ftp, smtp/imap o ssh. En el trabajo se implanta un *testbed* real en el que se genera tráfico con perfil de usuario y sobre el que se realizan 3 tipos básicos de ataques. El tráfico capturado es etiquetado y se permite la generación de *datasets* con perfiles específicos definidos por el usuario. Una aproximación similar se sigue en el reciente trabajo [14], que propone la creación de 3 *datasets* con características de tráfico real (tanto en el tráfico de *background* como en el de ataques): uno dedicado a la intrusión, otro al escaneo y otro dedicado a DDoS. Para el tráfico normal se capturó el tráfico de diversos perfiles sobre la red (estudiantes, profesores, etc.) y para el tráfico de ataque se lanzaron 15 ataques de los tres tipos indicados anteriormente sobre 6 escenarios distintos.

En nuestro escenario, centrado en las URI de HTTP que recibe un servidor web, los *datasets* con el tráfico de usuario deben basarse exclusivamente en el tráfico histórico recibido por dicho servidor, por lo que no sería válido tomar el tráfico normal de un *dataset* público ni de uno generado conforme a las metodologías expuestas. Adicionalmente, el tráfico de ataques debe ser representativo de aquellos presentes en las URI de HTTP, los cuales se encuentran muy residualmente en las bases de datos mencionadas previamente. Además, la base de datos de ataques debería contener ataques no detectados por los sistemas IDS basados en firmas (*signature-based* IDS o SIDS) actuales, permitiendo consecuentemente evaluar las mejoras que aporta el sistema propuesto frente a los basados en firmas. En este mismo contexto, otro grave inconveniente de cualquier conjunto de datos ya existente es que los valores de los campos que forman parte de la URI, p.ej. el '*path'*, se corresponden con un servidor distinto al que se está evaluando, por lo que es más que probable que esto afecte a la valoración

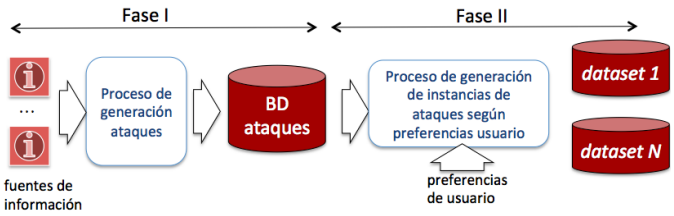


Fig. 1. Esquema general del proceso de generación de *dataset*

del grado de anormalidad de una URI e influya en los resultados obtenidos para los FN y FP.

III. Metodología propuesta

A grandes rasgos, el proceso seguido para la generación de *datasets* se muestra en la Fig. 1. Podemos distinguir en él dos fases:

- Fase I: partiendo de diversas fuentes de información disponibles, las actividades de esta fase tienen por objetivo generar una base de datos de ataques lo más completa posible. Algunos ataques contenidos en esta base de datos pueden contener parámetros, por lo que estos ataques no son necesariamente iguales a los incluidos en los *datasets* finalmente generados.
- Fase II: esta fase toma como entrada las preferencias del usuario del sistema para, tras procesar la base de datos de ataques, generar los *datasets* finales que cumplan las características deseadas en función de su uso. Los *datasets* contienen instancias de ataques (i.e. ataques cuyos parámetros ya han sido sustituidos).

Es importante hacer notar la distinción entre un ataque y una instancia de un ataque. Así, en nuestro contexto, un ataque es un elemento genérico que viene determinado por una vulnerabilidad, mientras que una instancia no es más que una de las posibles implementaciones de dicho ataque. Siempre que sea posible, los ataques serán identificados según el identificador de la vulnerabilidad asociada según Mitre (CVE, *Common Vulnerabilities and Exposures*) [15]. En nuestro caso, una instancia de un ataque será un URI al que se asignarán los diferentes identificadores en función del ataque al que corresponda. La base de datos de ataques puede contener tanto ataques como instancias de ataques, mientras que los *datasets* generados sólo contendrán instancias de ataques.

A continuación, se detallan los pasos seguidos en cada una de las fases descritas anteriormente.

IV. Generación de la base de datos de ataques

Se desea generar un repositorio de ataques partiendo de fuentes de información conocidas. En este trabajo se considerarán tres mecanismos de generación en función de la fuente de información utilizada (Fig. 2):

a) *Generación directa a partir de la base de datos de CVE*. En este caso, el listado de códigos CVE es la fuente de información para determinar los ataques y los pasos a seguir en la generación de instancias. Esta fuente presenta propiedades útiles, ya que permite filtrar los ataques de interés (p.ej. aquellos recientes o de alta criticidad) y que no sean necesariamente capturados por los sistemas SIDS mediante las firmas existentes.

El proceso seguido para generar ataques partiendo de esta base de datos es el siguiente:

1. Filtrar los CVE de interés. En nuestro caso, sólo son de interés aquellos que afectan únicamente a la URI de peticiones HTTP. Para ello se pueden emplear distintos filtros sobre su descripción (p.e. “HTTP”, “HTTP GET”), así como filtros por CWE (p.e. “*Crossite scripting*”). También resulta de interés la selección de CVE mediante otros criterios, como su fecha de aparición (p.e. considerando primero los más recientes), o su repercusión, a partir de su CVSS [16] (p.e. seleccionando únicamente los críticos, esto es, con

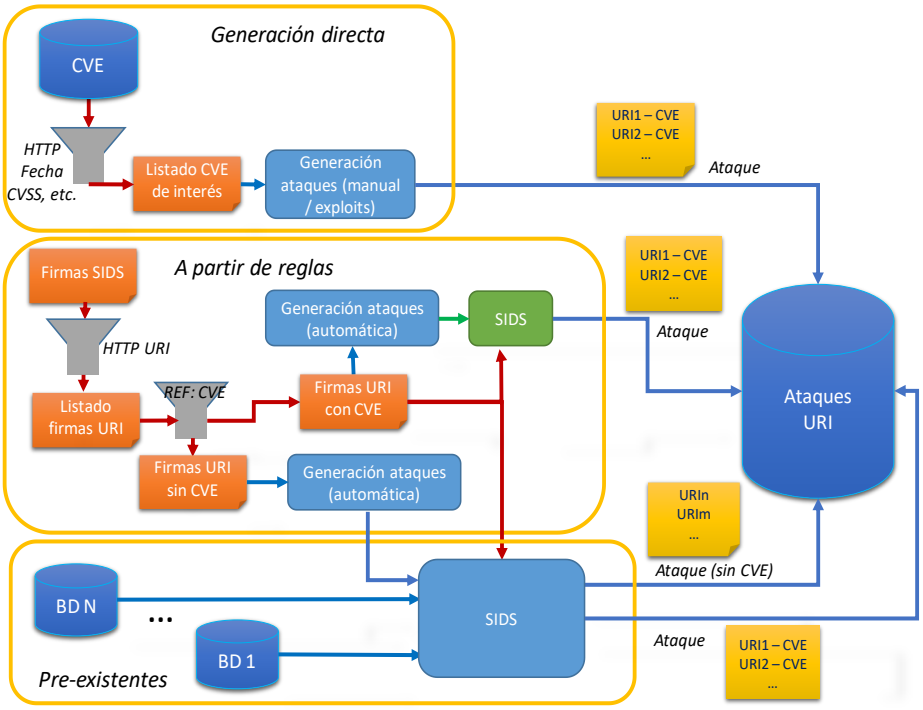


Fig. 2: Esquema de los métodos de obtención de ataques según la fuente de información.

CVSS > 9.0). Tras la preselección con los criterios anteriores, se ha leído la descripción de cada CVE para asegurar que el ataque está relacionado directamente con el interés de nuestra experimentación (i.e. ataques con URI) y descartar aquellos que no lo estén. En nuestro caso, esto se ha traducido en el filtrado del 40% de aquellos preseleccionados inicialmente tal y como se comentará en el Apartado VI.

2. Para cada CVE seleccionado, se recrea el correspondiente escenario del ataque (i.e. la URI). Para ello se analizan la descripción y referencias incluidas en el CVE. En caso de existir, se usan los *exploits* identificados en el CVE. En caso contrario, se recrea manualmente el URI de ataque. Evidentemente, este mecanismo de generación de ataques plantea la dificultad de requerir trabajo manual, por lo que su uso se ha orientado principalmente a la obtención de ataques recientes presumiblemente no incorporados aún en los repertorios de firmas para SIDS. En nuestro caso, se han reproducido finalmente la mitad de los ataques filtrados en el paso anterior tal y como se comentará en el Apartado VI.

b) *Generación automática a partir de las reglas de SIDS.* En este caso, la fuente de información usada para la generación de ataques serán las firmas existentes para los SIDS. Su principal ventaja estriba en que permite generar gran cantidad de ataques de forma automatizada a partir de las reglas seleccionadas. Lamentablemente, esto implica que todos los ataques así generados serán, obviamente, detectados por los sistemas SIDS, lo que los invalida para evaluar las capacidades de los detectores frente a la aparición de nuevos ataques (*0-day*).

Para generar ataques a partir de esta fuente se realizan los siguientes pasos:

1. Seleccionar las reglas de interés. En nuestro caso, aquellas que afectan exclusivamente a la URI de peticiones HTTP. Para ello, es necesario determinar los campos de las reglas que deben ser analizados tanto para incluir como para excluir algunas de ellas.
2. Generar las URIs a partir de las reglas seleccionadas en el paso anterior. Esto se ha realizado con una herramienta desarrollada a tal efecto de manera que esta generación sea automática.
3. Validar la nueva URI creada. Para garantizar que no se introducen distorsiones, se comprueba que la URI es detectada por la regla con la que se generó y, en caso de existir información suficiente por incluirse en la propia regla, se asigna a la URI el CVE correspondiente.

c) *Reutilización de bases de datos pre-existentes.* En este caso se consideran bases de datos pre-existentes con instancias de ataque, bien procedentes de fuentes contrastadas, o bien seleccionadas mediante las reglas de un SIDS. En ambos casos esta fuente de información presenta la ventaja de que los ataques están validados por algún procedimiento, aunque la información disponible sobre los diferentes ataques puede ser más limitada.

La incorporación de estos ataques es directa una vez identificada la base de datos, si bien resulta conveniente su procesamiento mediante un SIDS a fin de poder emparejar los ataques con el CVE correspondiente cuando las reglas incluyan esta información (Fig. 2).

A. *Parametrización de los ataques*

Gran parte de los ataques basados en URI son susceptibles de ser parametrizados, lo que permite la posterior generación de distintas instancias de ataques (i.e. distintas URI) que respondan al mismo patrón base. A modo de ejemplo, un caso simple de parametrización se correspondería con la posibilidad de usar diferentes rutas (*path*) en un ataque sin que cambie su naturaleza de ataque. Se podría definir, por tanto, un parámetro *[PATH]*, que puede ser adaptado según el caso del servidor que esté siendo modelado. En principio, se pueden identificar multitud de parámetros que pueden ser adaptados. Para el presente trabajo los parámetros considerados son: *IP, PATH, PORT, FILENAME, SQL-PAYLOAD, CODE-INJECTION, PATH-TRAVERSAL, RANDOM-STRING* y *COMMAND-EXE-ATTEMPT*.

De esta forma, en base a la parametrización, un ataque puede representarse mediante plantillas compuestas por una combinación de caracteres fijos, que son los específicos del ataque, y campos asociados a los parámetros que, consecuentemente, pueden tomar múltiples valores. Así, a partir de una plantilla de ataque pueden generarse múltiples instancias de ataques sin más que asignar valores según los criterios que se consideren oportunos y que pueden adaptarse al servidor concreto que esté siendo modelado.

Para la utilización de la parametrización es necesario definir un formato. En nuestro caso se utilizan los *[ ]* para delimitar los parámetros. Así, a modo de ejemplo, el ataque *[PATH]/?var=[FILENAME]* quedaría caracterizado por la inclusión de los caracteres */?var=* precedidos de un valor válido de ruta (*path*) y seguidos de un nombre de archivo (*filename*). Este ejemplo permitiría generar múltiples instancias de ataque adaptadas un servicio particular usando las rutas y archivos realmente existentes.

La generación de plantillas debe realizarse manualmente durante el proceso de incorporación de los ataques o a posteriori. La disponibilidad o no de plantilla para los ataques se utilizará también para definir la tipología que se detallará a continuación.

B. *Clasificación de los tipos de ataques*

Realizamos una clasificación de los ataques generados en función de su origen y características de parametrización. Esta clasificación permitirá dotar de propiedades a cada ataque en la base de datos donde se almacenan. Estas propiedades serán utilizadas por el generador de instancias de ataques a fin de poder construir las que cumplan con las preferencias indicadas por el usuario.

Se ha establecido la siguiente tipología (Fig. 3):

- *Tipo 1*: *Generación directa a partir de CVE*. Ataques generados manualmente o mediante los correspondientes *exploits* a partir de la información contenida en la base de datos CVE, tras la selección de los CVE de interés. Todos los ataques en esta categoría se encuentran parametrizados.
- *Tipo 2: Generación automática a partir de reglas de SIDS con CVE*. Ataques generados mediante una aplicación desarrollada al efecto para obtener URI a partir de la información existente en reglas de SIDS seleccionadas. Se consideran en este caso las reglas que afectan únicamente al URI y que incluyen el código CVE correspondiente. Todos los ataques se encuentran parcialmente parametrizados (sólo *[PATH]*).

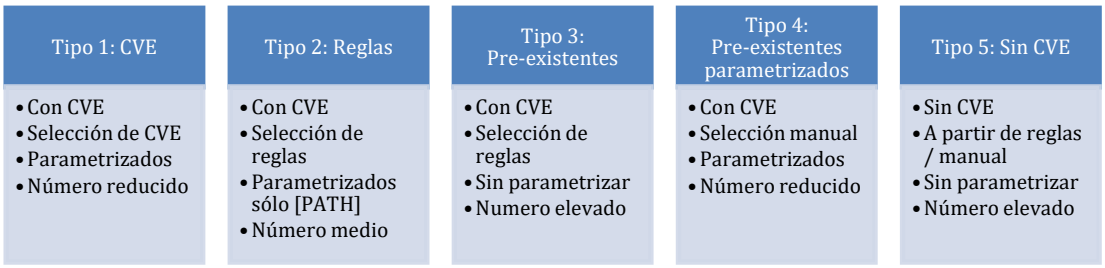


Fig. 3: Tipología de ataques y propiedades más relevantes.

- *Tipo 3: Ataques pre-existentes con CVE conocido*. En esta categoría se incluyen los ataques extraídos de bases de datos preexistentes para los que ha sido posible determinar el CVE asociado. No se ha procedido a la parametrización de los mismos.
- *Tipo 4: Ataques pre-existentes parametrizados*. Con la finalidad de disponer de un mayor número de ataques parametrizados, en esta categoría se incluye una fracción de los ataques Tipo 3 tras ser convenientemente parametrizados. Estos ataques permitirán la generación de un gran número de URI de ataques gracias al uso de múltiples parámetros, proporcionando una gran flexibilidad y capacidad de adaptación a un escenario determinado. Evidentemente, todos los ataques clasificados como Tipo 4 se encuentran incorporados también en la categoría Tipo 3.
- *Tipo 5: Ataques sin CVE asociado*. En esta categoría se incluyen ataques generados a partir de las reglas de SIDS carentes de CVE, así como los ataques procedentes de bases de datos pre-existentes a los que no haya sido posible asociarles CVE mediante un SIDS y las reglas correspondientes. Es relevante que, en este último caso, el uso del SIDS (Fig. 2) tiene como única finalidad el posible emparejamiento entre el URI y el CVE, ya que todos los URI en las bases de datos pre-existentes se consideran de ataque.

V. Generación de *datasets* a partir del repositorio de ataques y las preferencias del usuario.

Una vez recopilados los ataques, se sigue un procedimiento para generar instancias de ataques que reúnan las características preferidas por el usuario.

En la presente propuesta cada ataque o instancia de ataque considerado en este trabajo se clasificará, siempre que sea posible, de acuerdo a la vulnerabilidad asociada según la clasificación CVE [15], aunque esto mismo puede extenderse a otras bases de datos donde se clasifican ataques como OWASP [17] o Bugtraq [18]. Por otra parte, la base de datos de CVE asocia a cada vulnerabilidad un valor de su impacto potencial (métrica CVSS 3.0 *Common Vulnerability Scoring System* [16]), el software al que afecta (métrica CPE, *Common Platform Enumeration*, [19]), el tipo de vulnerabilidad (métrica CWE, *Common Weakness Enumeration*, [20]) y su fecha de aparición. Todos estos valores podrían ser utilizados como filtros a la hora de generar los distintos *datasets* de ataques. Así mismo, cada ataque podrá (o no) activar alguna regla en un SIDS, hecho que resulta relevante para este trabajo, por lo cual también se relacionará la regla o reglas que se activan en un SIDS determinado con el CVE correspondiente a la URI utilizada en el ataque.

A. *Procedimiento para la generación de datasets*

Las preferencias del usuario respecto a las características del *dataset* a generar incluyen parámetros como:

- Número total de instancias de ataques (número de URI en nuestro caso).
- Porcentaje de cada tipo de ataque según su tipología (p.ej. 30% de ataques tipo 1 y 70% de ataques tipo 2).
- Filtros de propiedades incluidas en CVE (p.ej. fecha de creación del CVE, criticidad CVSS, tipología CWE, etc.).
- Parametrización: para aquellos ataques parametrizables, la lista de parámetros para sustituir, junto con la probabilidad de aparición de los mismos (por ejemplo: *[PATH]* debe sustituirse por */inicio/hola* con un 20% de probabilidad y por */inicio/adios* con un 80%)
- Permitir o no ataques con el mismo CVE (siempre que su parametrización difiera).

Con estos parámetros de entrada, la salida del procedimiento sería un nuevo *dataset* que cumpliera, en la medida de lo posible, las características solicitadas. Este *dataset* debe incluir al menos la siguiente información:

- identificación del *dataset*,
- URI con ataques y sus características (CVE asociado, SID asociados, criticidad, fecha, etc.), y
- resumen, donde conste: total de URI generadas, porcentaje de URI generadas por tipo y porcentaje de aparición de cada parámetro en las URI generadas.

Para asegurar que los ataques del *dataset* se distribuyen por tipos conforme a la especificación del usuario se tomarán valores siguiendo una distribución uniforme dentro de cada categoría. Así, dentro de un tipo de datos, la selección de la URI es aleatoria y si el ataque resulta parametrizable y la opción de permitir ataques duplicados está activa, se sustituirán los parámetros por los indicados en el *dataset* del ataque correspondiente, tomando nuevamente una variable aleatoria uniforme para que la frecuencia de dichas sustituciones se ajuste a lo seleccionado por el usuario. En caso de que no se quisiera permitir ataques duplicados, o el ataque no fuera parametrizado, se tomarán los valores por defecto para dicho ataque.

Finalmente, antes de insertar una URI en el *dataset*, se verifica que no esté ya incluida. Si tras intentar tomar un ataque de cierto tipo resulta que está repetido en el *dataset*, la aplicación lo intentará nuevamente hasta un número de veces (100 por defecto) buscando un nuevo ataque o una nueva parametrización si fuera posible, tras lo cual considerará imposible la generación de dicho ataque del tipo especificado.

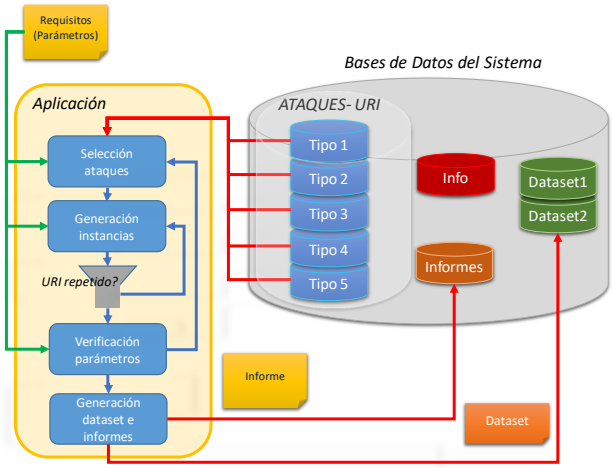


Fig. 4: Diagrama de operación de la aplicación desarrollada.

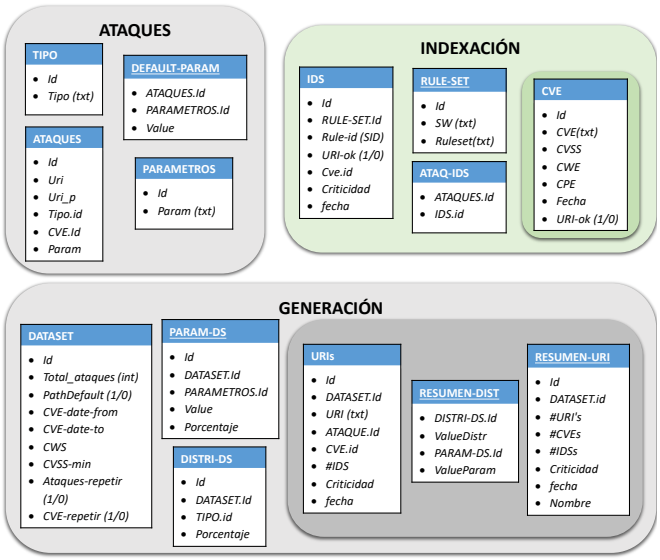


Fig. 5: Estructura de la base de datos usada.

B. *Aplicación desarrollada y modelo de datos*

La Fig. 4 muestra un esquema de la aplicación desarrollada. En esta figura se pueden distinguir los pasos seguidos en el procedimiento anterior: selección de ataques, generación de instancias que cumplan los requisitos, verificación de parámetros y generación del *dataset* final. El esquema también muestra la interacción de este proceso con diferentes repositorios de datos.

La base de datos diseñada para el sistema consta de 14 tablas que pueden agruparse, según su finalidad, en 3 repositorios (Fig. 5): ataques, indexación y generación.

- En el repositorio de ataques destacan las tablas asociadas a la base de datos de ataques, su tipología y sus parámetros por defecto. La tabla de ataques tiene como principales campos la *URI* del ataque original sin parametrizar, un indicador booleano sobre si se aplica parametrización y, en tal caso, la URI parametrizada. Cada ataque podrá tener asociada una vulnerabilidad (*CVE.Id*) y puede tener ninguna, una o varias reglas de SIDS asociadas.
- En el repositorio de indexación tendremos las tablas relacionadas con la información que puede ser asociada a un ataque; básicamente los datos asociados a vulnerabilidades (*CVE*): *CWE*, *CPE*, *CVSS*, *fecha*, etc. y los asociados a las posibles reglas de SIDS que activen. Dado que existen distintos SIDS, y que cada cual puede emplear un conjunto de reglas distinto, se ha normalizado en una tabla *RULE-SET* esta relación, por lo que en la tabla *SIDS* encontraremos el rule-set empleado (que incluye el programa y conjunto de reglas utilizado), y la identificación de la regla (*SID*), así como sus propiedades (por ejemplo, *CVE* asociado, si lo tuviere).
- Por último, en el repositorio de generación, podemos observar las tablas correspondientes a las condiciones de generación de un *dataset*: *DATASET* (con las características generales del *dataset*), *PARAM-DS* (parametrización para este *dataset*), *DISTRI-DS* (distribución de los tipos de ataques para este *dataset*). Los resultados tras la ejecución de la aplicación pueblan las tablas: *URIs* (con las URI que conforman las instancias de ataque y sus características) y resumen del *dataset* generado (*RESUMEN-URI*), así como de las distribuciones conseguidas en los parámetros y tipo de datos (*RESUMEN-DIST*).

Finalmente, la aplicación generará, además del *dataset* compuesto por las URI que conforman los ataques y sus características, dos tipos de informes: uno con los detalles del *dataset* generado (número total de URI, CVE, SID, etc.) y otro con el grado de cumplimiento conseguido para cada tipo de ataques y el porcentaje de uso de las parametrizaciones

VI. Caso práctico: población de la base de datos de ataques y generación de datasets.

En esta sección se describe cómo la aplicación descrita anteriormente ha sido utilizada para generar dos *datasets* de utilidad en el contexto descrito en el Apartado I. Para poblar la base de datos de ataques (fase I) se han utilizado las distintas fuentes de datos descritas en el Apartado III. El procedimiento para la obtención de los diferentes ataques a incorporar en la base de datos depende, obviamente, de su tipología.

A continuación, se detallan los pasos seguidos para la generación de la base de datos de ataques (fase I) en función del tipo de ataque generado:

- *Tipo 1*. En este caso se han de seleccionar los CVE de interés y, a partir de ellos, generar de alguna forma las plantillas de los ataques. Los pasos seguidos son:
  a) *Aplicación de filtros de interés al listado de CVE*: Se han preseleccionado como candidatos únicamente los CVE que cumplen los siguientes criterios:
     - Filtros de importancia: sólo se consideran los años 2016 y 2017 y CVE asociados a CVSS 3.0 crítica (>9.0).
     - Filtros de búsqueda de ataques tipo URI. Se han utilizado filtros de texto como "HTTP GET", “HTTP POST”, "HTTP", así como filtros asociados a los tipos de CVE como *SQL-INJECTION, DIRECTORY TRAVERSAL*, etc.

     Mediante este procedimiento se han preseleccionado 480 CVE.
  b) *Análisis de la vulnerabilidad correspondiente*: A partir de la inspección del CVE preseleccionado (p.ej. lectura de la descripción y referencias del CVE) se

toma una decisión sobre si es de interés y si el ataque puede ser implementado adecuadamente. En caso contrario se descarta el CVE. Finalmente se han determinado de interés 280 (i.e. ataques de URI) de los 480 preseleccionados en el paso anterior. De los 280, se decidió implementar la mitad (140) en base al esfuerzo requerido para implementar los ataques.

c) *Generación de instancia/s del ataque*: A continuación, se procede a la generación de una URI de ataque mediante el uso del *exploit* correspondiente, si está disponible, o manualmente en caso contrario. En caso de usar el *exploit* se procederá a la captura y selección de los paquetes generados por el mismo en un escenario adecuado. Adicionalmente, se evalúa la URI mediante un SIDS para determinar la SID correspondiente, si la tuviese.

d) *Parametrización*. Finalmente se procede de forma manual a la parametrización de la URI generada.

Por lo tanto, mediante este procedimiento manual se han generado y parametrizado 140 ataques. El esfuerzo empleado en la selección y generación de estos 140 ataques fue de un mes de una persona a tiempo completo.

- *Tipo 2*. La generación de ataques de este tipo requiere, por una parte, de la selección de las reglas del SIDS de interés y, por otra, del desarrollo de una aplicación que genere las instancias de ataque a partir de la información contenida en la regla. Por tanto, los pasos seguidos son:

  a) *Selección de reglas de URI con CVE*: El primer paso consiste en la extracción de las reglas que afectan únicamente al URI. Para ello se han filtrado las reglas oficiales de Snort y Suricata seleccionando las que afectan a los puertos HTTP y que incluyen los campos relacionados con el contenido de la URI (*http_uri, uricontent, http_raw_uri* o *content* -en última instancia-), pero que en ningún caso incluyen campos relacionados con otros elementos de las cabeceras de HTTP.

  De las reglas resultantes, se han seleccionado aquellas que tienen un CVE asociado, para lo cual se ha inspeccionado de forma automática el texto de la regla buscando *CVE* en un campo de tipo *reference*. Asimismo, a partir de los CVE de nuestro interés, se ha buscado la regla correspondiente haciendo uso de la aplicación *vFeed*. Como resultado de este primer paso tendremos una lista de SID (reglas) de interés con sus correspondientes CVE asociados.

  En el presente trabajo se han considerado las reglas de pago de Suricata de julio de 2017 (*su201707* en adelante) y las oficiales de Snort (Talos) de número de serie 2990 (*sn2990* en adelante). En el primer caso se detectan 543 reglas que afectan únicamente al URI y que tienen CVE asociado, mientras que en el caso, se identifican 1.182 reglas.

  b) *Generación automatizada de URI*. Se ha desarrollado un programa que genera URI válidos a partir de las reglas. Sin embargo, en su estado actual, tan sólo genera una cadena de texto aleatoria de una longitud determinada e inserta el valor del/los campo/s *CONTENT* definido/s en la regla correspondiente. Por tanto, en la versión inicial de esta herramienta no se recoge la generación de contenidos a partir de patrones de expresiones regulares de tipo *PCRE*.

  De las reglas seleccionadas en el paso a), únicamente se han podido generar ataques para 403 de ellas, incluyéndose su SID y su CVE asociado. Aunque estos ataques podrían ser parametrizados a mano, la enorme cantidad de trabajo que conlleva y la pérdida de automatización han originado que tan sólo se haya parametrizado de forma automática el *[Path]*).

  c) *Comprobación de los ataques generados*. A fin de validar cada una de las instancias de ataque generadas, se ha desarrollado un software que, dada una URI, compone un paquete HTTP que incorpore la URI utilizando el programa *Scapy* y lo emite a través de la red para su procesamiento por el SIDS pertinente. Se verifica así que se active la regla asociada y, consecuentemente, se valida la instancia de ataque generada. El resultado obtenido ha sido la validación correcta del 100% de las instancias de ataque.

- *Tipo 3*. En este caso no es necesario generar los ataques, por estar ya disponibles, sino únicamente asociarles su correspondiente CVE. Para ello se utiliza un SIDS y el conjunto de reglas con CVE asociado determinado previamente para los ataques de Tipo 2. De esta forma, los ataques detectados como tales con las reglas seleccionadas y a los que, consecuentemente, se les podrá asociar el CVE de la regla que se active, se incorporan sin más en la base de datos dentro de esta categoría.

  En nuestro caso se contaba con dos bases de datos previas generadas a partir de información sobre vulnerabilidades en el año 2009 gracias a un proyecto de investigación previo. En particular, se disponía de dos archivos, denominados *RDB* y *OSVDB*, conteniendo 933 y 6.896 ataques basados en URI respectivamente. Los ataques incorporados en *RDB* fueron generados manualmente a partir de las vulnerabilidades correspondientes a URI incluidas en aquel momento en la base de datos *Bugtraq* [17]. Por su parte, *OSVDB* incluye ataques generados también manualmente o mediante los correspondientes *exploits* a partir de la información en *OWASP* [16]. De estos ataques, fue posible asignarles CVE mediante el uso de las reglas *sn2990* y *su201707* a 716 de los contenidos en *RDB* y 4423 de *OSVDB*.

- *Tipo 4*. La generación de ataques Tipo 4 se ha realizado a partir de la inspección y parametrización manual de ataques de Tipo 3.

  Concretamente, se seleccionaron y parametrizaron 129 URI, con sus correspondientes reglas (SID) y vulnerabilidad (CVE) asociada.

- *Tipo 5*. Esta categoría incorpora todos los ataques e instancias de ataque disponibles o generados automáticamente para los que no ha sido posible determinar su CVE. En particular, incorpora las instancias de ataque de las bases de datos pre-existentes que no han generado alertas en los SIDS y las generadas automáticamente a partir de reglas que no incluyen CVE (Fig. 2). Por tanto, se incluyen 217 procedentes de *RDB*, y 2473 de *OSVDB*.

La Fig. 6 muestra la distribución de los URI incorporados en la base de datos según su tipología. Como resumen, podemos señalar que se ha poblado la base de datos anterior con 278 ataques completamente parametrizados (tipos 1 y 4),

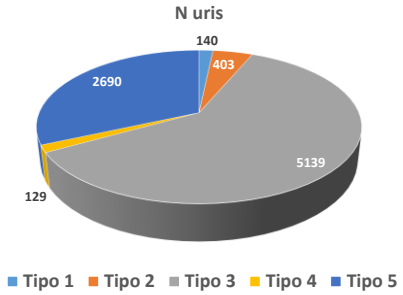


Fig. 6: Distribución de ataques/instancias por tipo.

403 ataques generados automáticamente a partir de reglas de SIDS que pueden ser adaptados a un *[PATH]* determinado, y 5139 ataques de bases de datos preexistentes. A modo de ejemplo, esta base de datos ha sido utilizada para generar distintos *dataset* de instancias de ataques entre los que se encuentran:

- DS800: contiene 800 instancias de ataques de Tipo 3.
- DS1100: contiene 1100 instancias de ataque, el 10% de Tipo 1, 10% de Tipo 4 y 80% de Tipo 3, donde no se permite repetir el patrón de ataques duplicados (parametrización única).

VII. Conclusiones y trabajos futuros

En el presente trabajo se sugiere un método para la generación de *datasets* basado en la creación de una base de datos de ataques parametrizables y su posterior procesamiento para, teniendo en cuenta las preferencias del usuario, generar los *datasets* correspondientes. La base de datos de ataques utiliza fuentes de información diversas (i.e. ha sido poblada por ataques de diversa procedencia), por lo que representa en la actualidad una de las más completas en este contexto. Aunque nuestro interés se ciñe a ataques basados en URI de peticiones HTTP, la metodología seguida puede ser adaptada a otros tipos de ataques.

Además, se ha desarrollado una aplicación que implementa la metodología recomendada. Su mayor potencial reside en la capacidad de generar conjuntos de instancias de ataque adaptados a las necesidades de la experimentación y el escenario, mediante el uso de una parametrización adecuada, y la combinación de diferentes fuentes de ataques. Esta es una característica muy relevante frente a la situación habitual de considerar *datasets* de ataques estáticos, esto es, capturados o generados en escenarios específicos y sin capacidad para modificar ningún elemento. En particular, nuestra herramienta ha sido utilizada para la creación de dos *datasets*, DS800 y DS1100, empleados en el entrenamiento y evaluación de un sistema SSM.

En el futuro se deberían incorporar nuevos ataques, especialmente de Tipos 1 y 4, ya que gracias a su parametrización permiten un buen ajuste al escenario real de pruebas, así como la posibilidad de generar tantas instancias de ataque como permita la parametrización.

Agradecimientos

Este proyecto ha sido parcialmente financiado por la CTA: "Sistema Integral para Vigilancia y Auditoría de Ciberseguridad Corporativa (SIVA)" (PI-1669/22/2017).

Referencias

[1] M. Bermúdez-Edo, R. Salazar-Hernández, J. Díaz-Verdejo, P. García-Teodoro, *Proposals on Assessment Environments for Anomaly-Based Network Intrusion Detection Systems*, LNCS (4347) 210-221, 2006.
[2] J. M. Estévez-Tapiador, P. García-Teodoro, J. E. Díaz-Verdejo, *Detection of Web-Based Attacks Through Markovian Protocol Parsing*, Proc. IEEE Symposium on Computers and Communications (ISCC'05), pp. 457-462 (2005).
[3] McHugh, J.: *Testing Intrusion Detection Systems: a Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory*, ACM Trans. on Information and System Security 3(4), 262–294, 2000.
[4] KDD Cup 1999 Data - UCI KDD Archive (http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html)
[5] 1998 DARPA Intrusion Detection Evaluation Data Set (https://www.ll.mit.edu/ideval/data/1998data.html)
[6] Information Systems Technology Group MIT Lincoln Lab, DARPA Intrusion Detection Data Sets, 2000. (http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/2000data.html)
[7] DEFCON, The SHMOO Group, 2011. (http://cctf.shmoo.com/)
[8] CAIDA, The Cooperative Analysis for Internet Data Analysis, 2011. (http://www.caida.org)
[9] Lawrence Berkeley National Laboratory (LBNL), ICSI, LBNL/ICSI Enterprise Tracing Project, 2005. (http://www.icir.org/enterprise-tracing/)
[10] UNIBS, University of Brescia Dataset, 2009. (http://www.ing.unibs.it/ntw/tools/traces/)
[11] Intrusion detection evaluation dataset (ISCXIDS2012) (http://www.unb.ca/cic/datasets/ids.html)
[12] J. Song, H. Takakura, Y. Okabe, D. Inoue, K. Nakao, Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation, Proc. First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security, pp. 29-36, 2010.
[13] A. Shiravi, H. Shiravi, M. Tavallaee, A. Ghorbani; Toward developing a systematic approach to generate benchmark datasets for intrusion detection, Computers & Security 31(3) 357-374 (2012).
[14] Monowar H. Bhuyan, Dhruba K. Bhattacharyya, and Jugal K. Kalita, Towards Generating Real-life Datasets for Network Intrusion Detection, International Journal of Network Security, (17)6, 683-701, 2015.
[15] CVE - Common Vulnerabilities and Exposures, https://cve.mitre.org.
[16] Common Vulnerability Scoring System SIG, https://www.first.org/cvss
[17] OWASP (Open Web Application Security Project), https://owasp.org
[18] Vulnerabilities – SecurityFocus, https://www.securityfocus.com/bid
[19] Official Common Platform Enumeration (CPE) Dictionary, https://nvd.nist.gov/products/cpe
[20] Common Weakness Enumeration: CWE, https://cwe.mitre.org/
[21] Rossow, Christian, et al. "Prudent practices for designing malware experiments: Status quo and outlook." Security and Privacy (SP), 2012 IEEE Symposium on. IEEE, 2012.

Modelo para la clasificación y el análisis de ataques *cross-platform*

Antonio Acien, Ana Nieto y Javier Lopez
Network, Information and Computer Security (NICS) Lab
Lenguajes y Ciencias de la Computación
Universidad de Málaga, España
Email:{acien,nieto,jlm}@lcc.uma.es

***Resumen*—Los ataques *cross-platform* suponen un serio desafío para los mecanismos de seguridad cuando los portadores de un ataque dirigido no son conscientes de su participación en el mismo. Es por ello que, con dispositivos y tecnologías cada vez más entrelazadas, en constante comunicación, numerosos ataques pasan desapercibidos hasta que alcanzan su objetivo final. Estos nuevos escenarios hacen posible una vía de transmisión a tener en cuenta, y que se debe abordar cuanto antes, ya que sus consecuencias, especialmente en el panorama de telecomunicaciones actual, podrían ser desoladoras. La rápida transmisión de estos ataques, y la dificultad que supone su prevención, detección y mitigación antes de que se hagan efectivos, hacen que el problema sea particularmente preocupante. En este artículo se presentará un modelo para el análisis de los ataques cross-platform silenciosos, cuyo objetivo es ayudar a comprender mejor este tipo de amenazas y ofrecer soluciones que permitan mitigarlas y rastrearlas.**

***Index Terms*—Cross-platform, architecture, attack, security.**

I. Introducción

Los ataques *cross-platform* son aquellos que afectan a múltiples plataformas y servicios, difíciles de detectar y rastrear, ya que la mayoría de medidas de seguridad se restringen a la seguridad de una plataforma. En este artículo, nos centramos en ataques cross-platform en los que el funcionamiento de los portadores intermedios no se ve directamente alterado y por ello son difíciles de detectar hasta que alcanzan su objetivo final. En un entorno en el que las distintas plataformas, tecnologías y protocolos están cada vez más interconectadas entre sí, este tipo de ataques representan un serio riesgo.

A lo largo de este artículo se verá cómo una plataforma puede ser comprometida desde otra muy distinta, empleando elementos intermediarios en la comunicación para pivotar. Dado que, a nuestro juicio, no hay modelos para el análisis de este tipo de ataques que permita entender su progresión en tiempo real, en este artículo se propone un modelo para el análisis de ataques cross-platform, con el objetivo de ayudar a su detección, prevención y mitigación.

Cabe destacar que la intercomunicación de distintas plataformas, o distintas capas dentro de una misma (p.ej. el caso de los *network slices* en 5G), es un escenario cada vez más recurrente en las comunicaciones modernas. No es raro tener un smartphone que se conecte a un ordenador, y éste a su vez a varios dispositivos más (routers, *gadgets* USB, o incluso otros smartphones), o incluso a un coche que también cuenta con una conexión a una red inalámbrica. También se halla un panorama similar en entornos virtualizados (conectándose el huésped al anfitrión) o en sistemas compuestos, a su vez, de varios sistemas empotrados que se comunican. La idea de un ataque infectando una de estas plataformas y propagando su actuación a las que se comunican con ella es cada vez más una realidad, que no sólo se ha dado en pruebas de concepto o despliegues controlados, sino en entornos de producción, y a usuarios medios (sección II). Con la aparición de nuevas tecnologías que fomentan estas interacciones (5G, mmWave, SDN...), es necesario establecer herramientas de análisis que entiendan este cambio de contexto.

El artículo se estructura como sigue. En la sección II se presenta un estudio del estado del arte en lo referente a ataques Cross-platform, y tras ello se presentará el modelo propuesto, llamado BTV (*Bearer, Transmitter, Victim*), en la sección III, que se formalizará en la sección IV. La sección V describe cómo se aplicaría el modelo propuesto en un caso de uso simplificado. Por último, se lleva a cabo una discusión de los resultados y unas conclusiones derivadas del trabajo realizado en las secciones VI y VII, respectivamente.

II. Análisis del Estado del arte

La definición de Cross-platform es relativamente amplia, pudiendo referirse a varias interpretaciones. Desde ataques que, con el mismo código, pueden afectar a varias plataformas (por ejemplo, aprovechando la naturaleza multi-plataforma de Java [1]), hasta aquellos que modifican su código dependiendo del sistema al que se dirijan (ataques metamórficos) [2]. Además, algunos trabajos sobre Cross-platform optan por centrarse en permisos y políticas para escenarios específicos, o procedimientos, métodos y librerías que tienen partes correspondientes en diferentes sistemas operativos o arquitecturas [3].

La mayoría de las contribuciones en este área se ha enfocado en redes en general: transmisión de paquetes, autenticación y autorización (control de acceso) entre otros temas [4]. Además, las redes que se han tenido en cuenta han sido casos concretos, como 3G [5] [6]. También se estudian casos muy específicos como en redes vehiculares, donde los vehículos son plataformas que integran otras plataformas. Por ejemplo, en [7] se emplea un CD preparado para causar alteraciones en el motor o la dirección, y en [8] un dispositivo vinculado envía mensajes a otras partes del vehículo, como los nodos

multimedia o los encargados de gestionar las comunicaciones externas.

También se han dado casos en escenarios de virtualización, donde el equipo anfitrión (host) es una plataforma que se sobre-entendía aislada del equipo virtualizado. Sin embargo, de sobra es sabida la existencia de ataques diseñados para escapar la barrera invitado-anfitrión, permitiendo que desde la máquina virtual se pueda escribir en zonas de memoria reservadas para la máquina que la hospeda. Estos ataques van más allá de las pruebas de concepto[9], habiendo exploits disponibles comercialmente para llevar a cabo el ataque en un entorno real [10].

El *Internet de las Cosas* (IoT) tampoco es una excepción. Por ejemplo, la versión para Windows de Mirai ataca en primera instancia a ordenadores con este sistema operativo, pero una vez llega a ellos, escanea su red local en búsqueda de dispositivos IoT vulnerables, los cuales pueda hacer parte de una botnet para llevar a cabo ataques de denegación de servicio distribuidos hacia otras infraestructuras [11].

Asimismo, se han documentado ataques pasando de smartphone a ordenador, y viceversa. En el caso de los primeros, normalmente se transmiten cuando el dispositivo móvil se conecta por USB, pinchando el micrófono del ordenador y enviando el audio grabado a un servidor externo, espiando conversaciones [12]. Siguiendo el flujo contrario, se encuentran ataques que se pasan del ordenador a un dispositivo Android, suplantando las apps que tiene instaladas por versiones aparentemente idénticas, que envían los datos de identificación introducidos a un servidor del atacante [13]. Esto es especialmente delicado cuando se trata, por ejemplo, de datos bancarios. También cabe destacar el caso de XcodeGhost, una versión modificada del entorno de desarrollo para iOS que infecta las aplicaciones elaboradas con el mismo y transmite este malware a los dispositivos donde se instalen [14].

Cabe destacar que los ataques Cross-plaform, aunque más preocupantes y comunes en los últimos tiempos, principalmente motivados por las amenazas persistentes avanzadas, no son nada nuevo, aunque la terminología y el enfoque que se le da en este artículo, más general, sí lo sea. La propagación de ataques a través de múltiples plataformas es algo que ocurre desde varias generaciones de comunicaciones atrás, y queda patente que aunque se lleven a cabo mejoras en la seguridad, sin soluciones que se habitúen a los cambios de contexto será muy difícil frenar la propagación de los ataques en los elementos portadores. Cualquier modelo de seguridad debería considerar este tipo de ataques y afrontarlos desde una perspectiva global. A medida que se desarrollan nuevas plataformas y tecnologías, también se desarrolla malware específico para ellas [15], lo que pone de manifiesto la urgencia de afrontar este problema. Por lo tanto, definir una estrategia y medidas frente a estos ataques es prioritario.

Este artículo proporcionará un enfoque general a este problema con el objetivo de ofrecer soluciones que permitan dar respuesta con independencia de la plataforma.

III. Modelo para el análisis de ataques Cross-platform

En esta sección se presenta el modelo BTV (*Bearer, Transmitter, Victim*). Su objetivo es cubrir los escenarios de ataques Cross-platform de una manera simple y general, con flexibilidad para adaptarse a cada uno de los diferentes escenarios y flujos de ataque. La idea principal detrás del concepto de BTV es la de proponer un esquema simple que consiste en un agente que alberga el ataque, conocido como *portador o bearer*. Este ataque se dirige a una víctima, a la cual no infecta directamente. En su lugar, el ataque pasa a través de otro agente, llamado *transmisor o transmitter*, que contiene el ataque, pero no ve su funcionamiento alterado (al menos no hasta el punto de poder detectar el ataque). Por último, el transmisor enviará inconscientemente el ataque a la víctima, que será quien sufrirá las consecuencias del mismo. Tras mostrar esta idea de una forma gráfica y detallada (sección III-A), se explicará cómo se emplearía el modelo para identificar ataques crosss-platform (sección III-B).

III-A. Componentes

El enfoque clásico llevado a cabo cuando se analizan ataques (sección II) es un escenario simplificado de dos actores. En dicho escenario, tanto el atacante como la víctima son agentes bien definidos: el atacante se esfuerza en propagar la infección a la víctima, la cual se ve comprometida y dañada. Sin embargo, este punto de vista puede quedarse corto en los ataques Cross-platform, los cuales involucran más sistemas con roles diferentes. Los sistemas normalmente están preparados para detectar el malware que se dirige a ellos, pero no a otros. Cuando surge una situación en la que un dispositivo se infecta pero su funcionamiento no cambia, detectar tal infección y evitar su propagación no es nada trivial. El rol de este transmisor inconsciente es esencial en muchos casos, porque permite a los ataques afectar a dispositivos a los que, de otro modo, no tendrían acceso.

El modelo BTV (Figura 1) permite estudiar la posibilidad de un ataque originado en una plataforma extendiéndose a otras distintas, aplicándose desde escenarios novedosos como 5G y SDN, o vehículos con conexiones de datos, hasta situaciones más cotidianas como conexiones USB o Bluetooth entre smartphones y ordenadores.

En el esquema hay tres actores principales, que se describen a continuación:

1. **Portador:** este participante en el esquema es el que comienza la propagación del ataque. Puede ser un atacante malicioso, con el objetivo de causar daño, o un usuario inconsciente que cree estar operando con normalidad.
2. **Transmisor:** es el agente que recibe el ataque, pero no se ve afectado directamente por él.
3. **Víctima:** el objetivo del ataque. Normalmente se comunica con un servidor CnC (*Command and Control*) para recibir instrucciones o al que enviar datos, aunque no siempre es el caso, ya que podría ser víctima de un ataque que no necesite comunicación externa.

Como se aprecia en Figura 1, hay más partes involucradas. Una de ellas es la tecnología de acceso, la cual, dependiendo

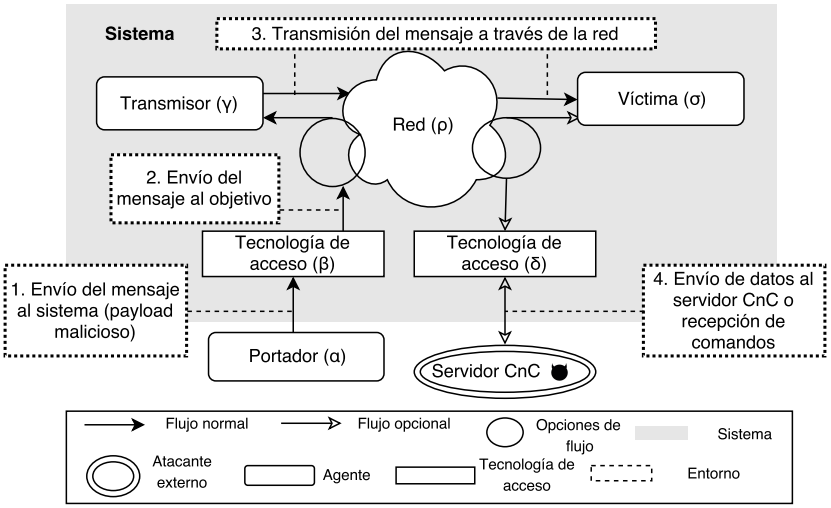


Figura 1. Diagrama del modelo BTV

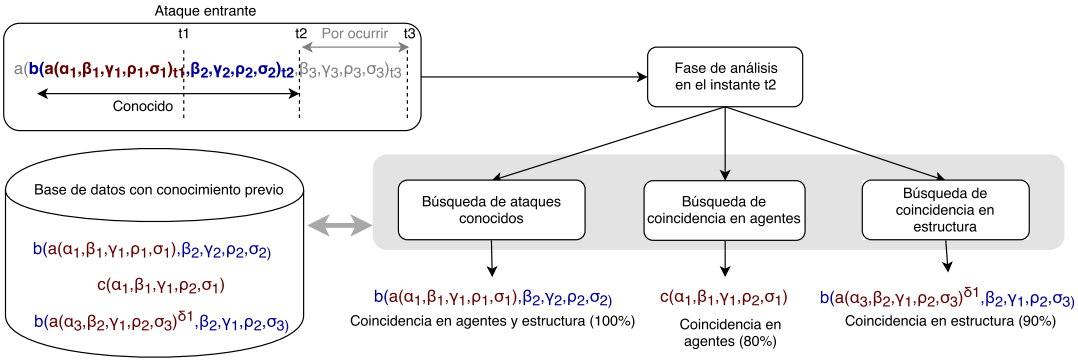


Figura 2. Implementación de un sistema de detección de amenazas basado en el modelo BTV

del tipo de ataque, variará. Si el portador es un servidor de Internet con malware, podría acceder al transmisor simplemente a través de un router normal. Si en cambio, se trata de un vehículo, este acceso será una puerta de enlace para comunicaciones externas o una ECU (*Electronic Control Unit*, unidad de control electrónica) en un sistema SCADA (*Supervisory Control and Data Acquisition*, control de supervisión y adquisición de datos).

También presenta una red que expresa la conexión entre el transmisor y el agente afectado, sea via WiFi, USB, Bluetooth, CAN o cualquier otro protocolo. Los mensajes pueden ser enviados a través de ella a cada uno de los agentes, o directamente a través de la tecnología de acceso.

Cabe destacar que, mientras que en algunos escenarios los agentes son dispositivos independientes que tienen conexiones momentáneas (como un smartphone y un ordenador), en otros son parte de un sistema completo e indivisible, donde comparten recursos o mensajes, tales como las distintas partes de un vehículo, o una máquina virtual y su anfitrión. Además de estos sistemas, se pueden definir entornos, siendo estos escenarios donde normalmente varios agentes interactúan entre ellos.

III-B. Despliegue

Al tener la posibilidad de analizar los ataques de manera sistemática mediante este modelo, surgen aplicaciones de la misma que, mediante una implementación, ayudan a prevenir, detectar y mitigar estas amenazas.

Una vez analizados los ataques Cross-platform conocidos, ya sean aquellos presentes en bibliografía, o los que se detecten en entornos de producción y sean analizados, se pueden usar sus patrones de propagación como conocimiento previo. Si se incluyen en una base de datos, comparar sus participantes o su estructura con las amenazas detectadas, puede ayudar a predecir su evolución, y prevenir su propagación o tomar las medidas necesarias para mitigarlas.

La comparación con los ataques conocidos se puede realizar de varias maneras. Se puede comparar de manera estructural, viendo así si los agentes implicados siguen un cierto patrón y predecir si el ataque se extenderá a otra plataforma. También se puede comprobar si los agentes implicados coinciden con los de algún caso conocido, sabiéndose así si se trata de algún ataque visto previamente, o quizás una variación del mismo.

Al llevar a cabo estas comprobaciones con los ataques de la base de datos, se ha de establecer un criterio que permita indicar el índice de similitud. Esto ayudará a decidir qué

medidas tomar, ya que si el parecido a una cierta amenaza es alto, servirá de referencia para hacer frente a la actual.

Este criterio de similitud se puede basar en distintas observaciones. Una de las comparaciones a realizar es estructural: si el ataque sigue la misma estructura que alguno conocido, aunque mediante distintos agentes, se puede predecir si se va a producir una propagación del mismo y cómo. En cambio, si la estructura es distinta, pero coinciden los agentes, se puede prever a qué agentes más podría afectar si sigue extendiéndose con dicha estructura. Por supuesto, pueden coincidir ambas situaciones, y corresponderse totalmente el ataque con uno ya conocido. Estos casos se ilustran en la Figura 2. Cabe destacar, que las flechas que marcan la correspondencia de los ataques con otros vectores similares de la base de datos cambiarían conforme se tenga más información del ataque, expresado a través de la barra de tiempo t.

Como ya se ha indicado, una de las mayores dificultades que presentan los ataques Cross-platform es su detección, ya que las plataformas normalmente están preparadas para detectar y mitigar ataques dirigidos específicamente a ellas, y esta tarea se dificulta si para ellas el ataque es inocuo y sólo colaboran en su expansión. Mediante este conocimiento previo expresado con el modelo se lograría hacer frente a este problema, ya que al detectar un ataque y compararlo con los datos, se conocerían las posibilidades de expansión que posee y el posible funcionamiento de la misma.

IV. Formalización

Con objetivo de ayudar a la definición del modelo, y hacer que tenga una aplicación más directa y comprensible para una máquina, se ha decidido formalizar su estructura con un lenguaje sencillo pero versátil, de manera que se pueda instanciar en cada uno de los casos necesarios. Al hacer que se pueda expresar de una manera uniforme, también se posibilita que sea fácilmente comprensible por máquinas, ayudando a su procesado automático. Esto ayudará a llevar a cabo la detección que se detalla en la sección V, ya que las comparaciones comentadas se pueden hacer de forma sistemática comparando variables o expresiones.

IV-A. Vector de ataque

La expresión general por la que se define el vector de ataque (V) es la siguiente:

$$V = f\,x\,{}_t^{\delta},\ t \in \mathbb{N} \tag{1}$$

donde $f\,x$ es la función de flujo (sección IV-D), $delta$ es la tecnología de acceso (entre la víctima y el servidor CnC si lo hubiera, sección IV-C), y t es el instante de tiempo, que nos permite definir momentos para observar cómo profundiza o avanza el ataque. Las variables del vector x se detallan en la sección IV-C.

Mediante este lenguaje, que se sirve de distintas variables se puede cubrir cada caso contemplado por el modelo BTV. Esta expresión describirá los *vectores de ataque* (V) del modelo BTV. A continuación describimos de forma detallada los conjuntos, variables y funciones de flujo y delimitadores que proporcionan significado a esta fórmula.

IV-B. Conjuntos

Se definen dos conjuntos, A y T, para expresar los posibles agentes y tecnologías, respectivamente. Son conjuntos compuestos por cadenas de caracteres. Dado que no es posible incluir todas las posibilidades en estos conjuntos, se instanciarán en cada caso de uso, incluyendo los miembros necesarios.

Ejemplo:

$$A = \{Servidor, Smartphone, NodoCAN...\} \tag{2}$$

$$T = \{USB, WiFi, Bluetooth, ZigBee, CAN...\} \tag{3}$$

IV-C. Variables del modelo BTV

En el modelo se emplean varias variables, que se pueden ver de forma gráfica en la Figura 1. Aquí procedemos a describirlas.

$$\alpha, \gamma, \sigma \in \{V, A\} \tag{4}$$

$$\beta, \rho, \delta \in T \tag{5}$$

$$t \in \mathbb{N} \tag{6}$$

- α**:** Portador del ataque
- β**:** Tecnología de acceso al transmisor
- γ**:** Transmisor del ataque
- ρ**:** Red que comunica al transmisor con la víctima
- σ**:** Víctima del ataque
- δ**:** Tecnología de comunicación con el servidor CnC (si hay)

Como se aprecia en la exprs. (4), α puede ser a su vez otro vector de ataque, aportando así recursividad. Lo mismo sucede con el transmisor y la víctima del ataque. El instante de tiempo en el que se produce el ataque es representado por medio de t, pudiendo variar a medida que se conoce más información sobre el mismo. Por último, señalar que δ es opcional, ya que no en todos los escenarios se establece comunicación con servidores CnC.

IV-D. Funciones de flujo

Para representar el flujo del ataque en el vector, se pueden indicar cada uno de los componentes del modelo por los que pasa el ataque en su camino de manera secuencial. Sin embargo, esto haría que los vectores tuvieran un número de miembros distinto dependiendo de la situación, y que en varios de ellos se repitiera. Para evitar esta repetición, la función de flujo representa el camino del ataque por los distintos agentes a los que afecta con un solo símbolo. La función de flujo se expresa de forma general como f, pero se instanciará a funciones de flujo específicas que describen el flujo específico del ataque (véase la Figura 3): a, b, c o d. Estas funciones se usarán extensivamente en el caso de uso (Sección V), para expresar de manera simple el flujo de los ataques mostrados.

$$f\,x = \begin{cases} a\,x & \text{si}\, x = \alpha \to \beta \to \gamma \to \rho \to \sigma \\ b\,x & \text{si}\, x = \alpha \to \beta \to \boldsymbol{\rho} \to \gamma \to \rho \to \sigma \\ c\,x & \text{si}\, x = \alpha \to \beta \to \gamma \to \rho \to \sigma \to \boldsymbol{\rho} \\ d\,x & \text{si}\, x = \alpha \to \beta \to \boldsymbol{\rho} \to \gamma \to \rho \to \sigma \to \boldsymbol{\rho} \end{cases} \tag{7}$$

Hay que destacar que las funciones a y c son muy similares, y el único cambio que presentan es que, en caso de ocurrir una propagación del ataque (ya sea a otro entorno, o a un servidor de CnC), ésta se hará a través de la red. Por lo tanto, que un ataque *mute* de una función a otro en su funcionamiento es muy posible, ocurriendo lo mismo entre las funciones b y d.

IV-E. Delimitadores

Además de las variables previamente indicadas, también se incluyen los símbolos $[\,]$ y $\{\,\}$. Éstos se usan para delimitar la extensión de un sistema y un entorno, respectivamente. Tener unos agentes comprendidos entre los límites de un sistema significa que son indivisibles y tienen un funcionamiento conjunto. Por otra parte, los entornos representan conjuntos de dispositivos que normalmente actúan entre ellos para llevar a cabo ciertas funciones, pero pueden operar de manera independiente. Hay ciertas reglas a seguir a la hora de usar estos símbolos:

1. Tanto un entorno como un sistema pueden existir el uno sin el otro. Sin embargo, un entorno no puede estar comprendido dentro de un sistema. Un sistema sí puede estar comprendido dentro de un entorno.
2. En caso de que haya un sistema cuyo límite comience dentro de un entorno, su fin también tiene que estar dentro del entorno.
3. Tanto los delimitadores de sistema como los de entorno actúan sobre las variables $\alpha, \beta, \gamma, \rho$ y ϵ, no pudiendo aplicarse fuera de ellas.

Si, por ejemplo, hay un entorno que comprende la tecnología de acceso, el transmisor, y la red, y un sistema conformado por estos dos últimos, se expresará como $f(\alpha, \{\beta, [\gamma, \rho]\}, \sigma)$.

Una vez definido el contexto a analizar empleando esta formalización, llevar a cabo la implementación de un sistema de mitigación de ataques basado en el modelo BTV es más sencillo, ya que todos los ataques se pueden expresar mediante fórmulas con una sintaxis común.

V. Caso de uso

En esta sección se estudiará un caso de uso que implica varias tecnologías, observando así cómo el modelo BTV se puede aplicar a diferentes escenarios y protocolos, y mostrando su utilidad práctica en situaciones reales. Aunque en el caso de uso se mencionarán tecnologías y dispositivos específicos para explicarlo, cabe puntualizar que habría infinitas variaciones. Asimismo, se formalizará, poniendo en práctica las funciones y variables vistas en la sección anterior aplicadas tanto a dispositivos como tecnologías concretas.

V-A. Primera fase

En la primera fase del ataque, un servidor de internet que alberga malware infecta un ordenador a través de un *exploit* que se aprovecha de una vulnerabilidad en el navegador para instalar un archivo ejecutable con código malicioso (pasos 1 y 2 de Figura4). El usuario del ordenador no es consciente de que el malware se instala en su máquina porque no afecta a su actividad normal, pero éste se encuentra en segundo plano monitorizando las conexiones USB hasta encontrar un terminal Android (paso 3). Cuando esto ocurra, el malware buscará una app que se conecte a un vehículo por Bluetooth, desinstalándola y sustituyéndola por una versión idéntica a ojos del usuario (paso 4), pero que se conecta con el servidor CnC del atacante (paso 5) y tiene propósitos maliciosos. Hay casos de malware que siguen este patrón de operación con otro tipo de aplicaciones, como por ejemplo, bancarias.

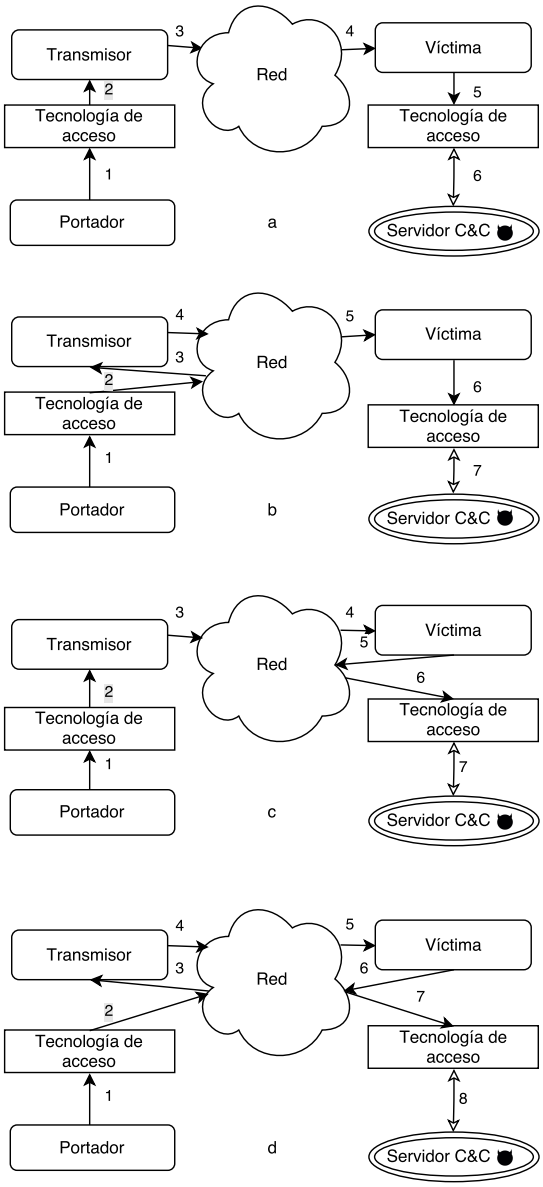


Figura 3. Posibles flujos del ataque que representan las distintas funciones

Así, el portador en esta fase sería el servidor externo, mientras que el ordenador y el dispositivo Android tendrían los roles de portador y víctima, respectivamente. A su vez, el smartphone podría ser portador si la cadena del ataque continúa.

Para expresar esta parte del caso de uso mediante la formalización presentada en la sección IV, se usarán las abreviaturas presentadas en la tabla I, para no extender la longitud de la

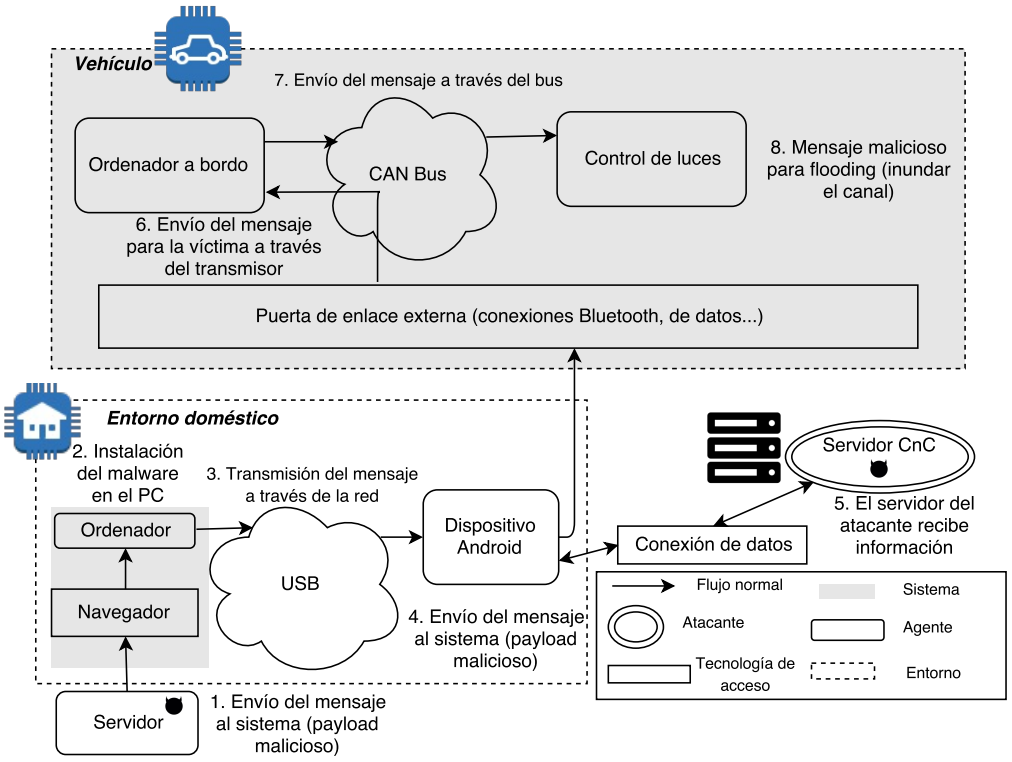


Figura 4. Aplicación del modelo BTV en un caso de uso

Tabla I
TABLA DE ABREVIATURAS PARA EL CASO DE USO

Acrónimo	Significado
SRV	Servidor
PC	Ordenador
AND	Dispositivo Android
OBC	Ordenador a bordo
GW	Puerta de enlace
LC	Control de luces
NVG	Navegador
DAT	Conexión de datos
NODE	Nodo
ENG	Motor

fórmula de manera innecesaria. La elección de estas abreviaturas es totalmente arbitraria y se puede extender añadiendo tantas como sea necesario, pero debe haber coherencia en usar siempre las mismas dentro de una implementación si se lleva una a cabo.

La aplicación de la fórmula que formaliza el ataque es bastante directa, dado que solamente hay que sustituir variables y funciones. Por simplicidad, se usarán abreviaturas para expresar tanto tecnologías como agentes, encontrándose éstas detalladas en la tabla I. En este caso, quedaría de la siguiente manera:

$$V = f(\alpha_1, \beta_1, \gamma_1, \rho_1, \sigma_1)_{\epsilon}^{\delta} \quad (8)$$

$$V_{CU} = a(SRV, NVG, PC, USB, AND)_{t_1}^{DAT} \quad (9)$$

Al no haber un servidor CnC en este escenario, no se incluye en la expresión previa. También se puede apreciar que, al estar analizando este escenario de manera estática, y no a lo largo del tiempo, tampoco se incluye el instante como subíndice.

V-B. Segunda fase

Una vez el smartphone haya sido infectado como se ha visto, la app instalada puede monitorizar conexiones Bluetooth con vehículos. Esto significa que al vincularse con un vehículo, puede intentar comprometer su funcionamiento. Las conexiones en el interior de los vehículos siguen el protocolo CAN (*Controller Area Network*, red de área de controladores), que rige la comunicación a través de un bus al que se conectan los distintos nodos, como pueden ser cosas tan diversas como control de ventanillas o luces, hasta dirección o motor. Debido al diseño de este protocolo, es muy sencillo realizar ataques de denegación de servicio desde cualquiera de los nodos, inundando el canal con un envío de mensajes masivo.

Hay otra clase de ataques más sofisticados, que pueden ir dirigidos a alguno de los nodos que componen la red interna del vehículo, originándose desde cualquiera de ellos. Estos ataques, sin embargo, requieren más sofisticación y un conocimiento amplio del firmware y modelo de vehículo concretos.

Tanto haciendo un ataque más sencillo como uno más complejo, se tendría un escenario en el que un smartphone Android se conecta al sistema de un automóvil mediante Bluetooth. Estas comunicaciones, en un vehículo moderno, se suelen hacer a través de una puerta de enlace para comunicaciones externas. Una vez vinculado, el smartphone podría comenzar un envío masivo de mensajes a través de esta puerta de enlace (paso 6), impidiendo así la comunicación de otros nodos con órdenes importantes, causando una denegación de servicio. Aunque algunos vehículos modernos cuentan con buses a distintos niveles para las comunicaciones críticas o con *firewalls*,

pero ha habido ataques en los que se ha logrado tumbar estos mecanismos de defensa. Otro tipo de ataque podría ser uno en el que el smartphone envía un mensaje al ordenador central para que lo redirija a otro nodo, que sería su víctima (paso 7). Este mensaje contendría un payload malicioso que afecta a la víctima, pero no al transmisor, debido a la codificación del mismo. Este nodo podría ser prácticamente cualquiera (ventanillas, cuentakilómetros, indicador de gasolina...), pero por instanciar el caso de uso, se usará el control de luces (paso 8).

Como se ha visto, el dispositivo Android pasa de ser la víctima de un ataque a ser el portador del mismo, ya que se puede transmitir a otras plataformas. En este caso, el transmisor sería la puerta de enlace de comunicaciones externas del vehículo, a la que se accede por Bluetooth, que hace las veces de tecnología de acceso. Una vez el trasmisor se vea infectado por el malware, continuará su operación con normalidad (transmitir los mensajes de los dispositivos externos que se hayan conectado), pero los demás nodos del vehículo, que tienen el papel de víctimas, y a los cuales se accede mediante el bus CAN, no podrán realizar su comunicación, ya que la inundación de paquetes por parte del smartphone habrá causado una denegación de servicio.

Como el smartphone que hace de portador, a su vez fue infectado como víctima, se puede detallar este escenario como una recursión en la fórmula. Además, al pasar los mensajes por el bus CAN desde la tecnología de acceso hasta el ordenador de a bordo, habría un cambio de función, por lo que la expresión resultante, sería la siguiente:

$$V = f(f(\alpha_1, \beta_1, \gamma_1, \rho_1, \sigma_1)^{\delta}, \beta_2, \gamma_2, \rho_2, \sigma_2)_{\epsilon} \quad (10)$$

$$V_{aCU} = b(a(SRV, NVG, PC, USB, AND)^{DAT}, GW, OBC, CAN, LC))_{t_2} \quad (11)$$

Este ataque se vale de que el estándar CAN sigue un diseño muy simple pensado para microcontroladores con poca capacidad de procesamiento, y que se centra más en comunicaciones a tiempo real. Por tanto, los mecanismos para evitar inyección de mensajes o ataques de denegación de servicio deteriorarían este tiempo de respuesta, ya que suponen operaciones más complejas.

Ya que se ha llevado a cabo la formalización del ataque observado en el caso de uso, podemos compararlo mediante la implementación detallada en la sección III-B con ataques registrados. Estos ataques son, tanto amenazas reales registradas en bases de datos de malware[13], como pruebas de concepto académicas [16].

Se puede comprobar que el ataque tiene similitudes con el troyano TROJ˙DROIDPAK.A [13], ya que usa los mismos agentes (un ordenador personal que se infecta desde una conexión externa y se transmite a un smartphone conectado por USB), y con el ataque a automóviles remoto descrito por Miller y Valasek [16], ya que la estructura es similar (mediante una conexión externa, se envían paquetes para causar que un nodo del vehículo tenga un funcionamiento distinto). Gracias a estas similitudes se puede saber qué medidas tomar para evitar su propagación o mitigar su actuación, ya que éstas están detalladas en los artículos y en las bases de datos de malware [17], tales como modificar ciertos registros de Windows o cerrar las conexiones externas a los vehículos.

Como se ha visto, el modelo BTV propuesto es extensible a varias situaciones, distintas plataformas y protocolos, y lo suficientemente versátil como para adaptarse a ataques más complejos.

VI. Discusión

Hay varios detalles a comentar en relación a los ataques Cross-platform y al enfoque seguido al desarrollar el modelo BTV. Por una parte, habría que discutir cuánto de factible es realizar un ataque de la magnitud comentada en este artículo en los sistemas actuales. Por ejemplo, para ataques sofisticados, desarrollar un firmware modificado (p. ej. para un automóvil) no es algo trivial bajo ningún concepto. Si bien en algunas pruebas de concepto, cambiar unas pocas líneas de código ha cambiado el comportamiento del vehículo de manera crucial [8], se necesitaría tener antes acceso al firmware del vehículo, lo cual no es fácil de obtener. Aunque esto se puede hacer mediante ingeniería inversa, se trata de una tarea larga y ardua, cuyo esfuerzo es difícil ver compensado. Sin embargo, la preocupación en este ámbito es creciente, como demuestran las noticias y estudios recientes al respecto [18]. Además, en relación a los ataques en plataformas de virtualización, no se debe olvidar que para tener éxito, necesitan configuraciones o versiones concretas, las cuales no se suelen tener en entornos de producción, ya que con un mínimo de mantemiento se deben llevar los parches al día.

Sin embargo, incluso teniendo en cuenta los detalles de estas situaciones y lo específicos que son los escenarios que se necesitan a veces para llevar a cabo ataques Cross-platform, no significa que estos ataques no sean factibles. Cada parte de los casos de usos mostrados se basa en casos reales de ataques vistos, ya sea en pruebas de concepto, o en la práctica, por lo que existe un riesgo real de que vayan a más y tengan consecuencias fatales para las potenciales víctimas. El modelo BTV pretende cubrir tanto estos escenarios como los nuevos que surjan. Para ello se han de definir funciones que hagan la comparación, ya sea variable a variable, o que comparen la similitud de la estructura general. En base a estas operaciones, se puede definir un porcentaje en similitud a cada uno de los ataques conocidos y saber con cuál encaja más para establecer las medidas de seguridad necesarias.

Los métodos a seguir para la búsqueda de ataques Cross-platform en bases de datos de ataques o bibliografía sobre el tema también son un punto a tener en cuenta, ya que son cruciales para la elaboración de dicha base de datos con conocimiento previo.

Por último, consideramos que este tipo de enfoque podría ser muy beneficioso para la quinta generación de redes celulares (5G), por dos motivos. Por una parte, por la gran sinergia de tecnologías software que cooperan para ofrecer servicios, empleándose multitud de plataformas diversas e integrando al usuario final (y sus dispositivos) más que nunca como parte del ecosistema. Por otra parte, por motivos operacionales y de despliegue. Este modelo y su implementación necesitan gran capacidad de recursos para su correcto funcionamiento. El procesamiento en tiempo real de los vectores de ataque requiere

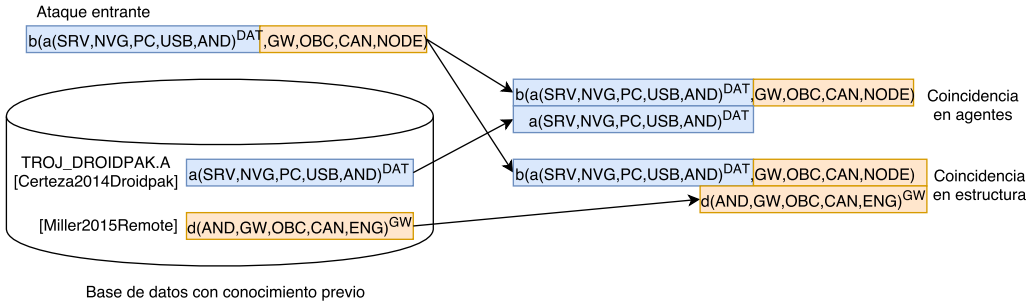


Figura 5. Comparación del ataque del caso de uso con malware conocido, utilizando la formalización propuesta

de clasificación avanzadas y manejar gran volumen de datos de forma eficiente. Estos servicios podría proporcionarlos la infraestructura 5G desde el core.

VII. Conclusiones y trabajo futuro

Los ataques Cross-platform son una realidad que se extiende cada vez más, y, conforme mejoran las infraestructuras de comunicaciones, también las posibilidades de propagación serán mayores. En este artículo se define y formaliza un modelo capaz de expresar las variaciones de los ataques cross-platform, proporcionando un enfoque especializado a cada caso, pero dentro de un mismo lenguaje para expresar los vectores de ataque y facilitar su comparación. El objetivo es ayudar en la prevención y mitigación de estas amenazas, permitiendo que los componentes intermedios puedan predecir si algo que está ocurriendo podría afectar a otros sistemas con los que se interrelacionan.

En lo relativo al trabajo futuro y mejoras, la aplicación directa y práctica de este trabajo pasa por modelar, empleando BTV, un conjunto de arquitecturas y sistemas que hayan sido objeto de ataques Cross-platform, y analizar sus relaciones para encontrar posibles similitudes y observar si se puede obtener información sobre vulnerabilidades de los mismos en base a ellas.

Además, se analizarán los ataques Cross-platform conocidos, y se procederá a su formalización con el modelo propuesto, para tener una base de conocimiento lo suficientemente representativa con la que elaborar un sistema de detección y prevención que ayude a mitigar el efecto de ataques de esta naturaleza. Un entorno en el que puede ser particularmente útil este sistema es 5G, debido a la naturaleza software y multi-capa en la que se basan sus comunicaciones.

VIII. Agradecimientos

Este trabajo ha sido financiado por el Ministerio de Economía y Competitividad a través de los proyectos IoTest (TIN2015-72634-EXP) y SMOG (TIN2016-79095-C2-1-R). El segundo autor ha sido financiado por INCIBE a través del programa de ayudas para la excelencia de los equipos de investigación avanzada en ciberseguridad.

Referencias

[1] M. Lindorfer, M. Neumayr, J. Caballero, and C. Platzer, "Poster: Cross-platform malware: write once, infect everywhere," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 1425–1428.

[2] Eugene, "Architecture spanning shellcode," 2000.

[3] K. Chen, X. Wang, Y. Chen, P. Wang, Y. Lee, X. Wang, B. Ma, A. Wang, Y. Zhang, and W. Zou, "Following devil's footprints: Cross-platform analysis of potentially harmful libraries on android and ios," in *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 2016, pp. 357–376.

[4] C. R. Reeves Jr, "Cross platform network authentication and authorization model," Feb. 13 2007, uS Patent 7,178,163.

[5] W. Jia, D. Bin, and L. Liao, "Architecture of secure cross-platform and network communications," in *Proceedings of the 2nd international conference on Ubiquitous information management and communication*. ACM, 2008, pp. 321–328.

[6] K. Kotapati, P. Liu, Y. Sun, and T. LaPorta, "A taxonomy of cyber attacks on 3g networks," *Intelligence and Security Informatics*, pp. 129–138, 2005.

[7] S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham, S. Savage, K. Koscher, A. Czeskis, F. Roesner, T. Kohno *et al.*, "Comprehensive experimental analyses of automotive attack surfaces." in *USENIX Security Symposium*. San Francisco, 2011.

[8] K. Koscher, A. Czeskis, F. Roesner, S. Patel, T. Kohno, S. Checkoway, D. McCoy, B. Kantor, D. Anderson, H. Shacham *et al.*, "Experimental security analysis of a modern automobile," in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 447–462.

[9] N. Elhage, "Virtunoid: Breaking out of kvm," *Black Hat USA*, 2011.

[10] K. Kortchinsky, "Cloudburst: A vmware guest to host escape story," *Black Hat USA*, 2009.

[11] M. Mimoso, "Windows botnet spreading mirai variant," 2017.

[12] "Android.ciaco," 2013.

[13] R. Certeza, "Cross-platform mobile threats: A multi-pronged attack," 2014.

[14] X. Gui, J. Liu, M. Chi, C. Li, and Z. Lei, "Analysis of malware application based on massive network traffic," *China Communications*, vol. 13, no. 8, pp. 209–221, 2016.

[15] M. Sargent, "Malware trends: The rise of cross-platform malware," 2012.

[16] C. Miller and C. Valasek, "Remote exploitation of an unaltered passenger vehicle," *Black Hat USA*, vol. 2015, 2015.

[17] G. R. Joi, "Troj˙droidpak.a," 2014.

[18] F. Maggi, "The crisis of connected cars: When vulnerabilites affect the can standard," 2017.

This Is My Private Business! Privacy Risks on Adult Websites

Pelayo Vallina[†◇], Julien Gamba[†◇], Alvaro Feal[†◇], Narseo Vallina-Rodriguez[† ‡], Antonio Fernández Anta[†]
† IMDEA Networks Institute, ◇Universidad Carlos III de Madrid, ‡ ICSI

***Abstract*—Millions of users from all over the world access pornographic content on the web. However, as of today, there's a dearth of knowledge about the porn web ecosystem and, in particular, about the privacy risks to which porn web users get exposed. As a matter of fact, porn websites can infer sensitive private information from users, including their sexual orientation and preferences, which can be later linked to their identity. As a result, many porn web consumers resort to ad blockers and private browsing modes to protect their identity when browsing adult websites. In this work in progress, we aim to expose the porn web ecosystem hoping to analyze aspects like their age verification mechanisms — which are required in certain regulatory frameworks as the British one —, their user tracking and data collection practices, and their privacy policies. By leveraging simple methodologies, we have already obtained relevant preliminary results: (1) we have identified a significant presence of third-party tracking services on porn websites; (2) we have discovered the use of both first-party and third-party cookies, most times installed without the user consent; and (3) we have found a concerning lack of age verification mechanism to prevent the access of minors to porn websites. In the future, we plan to study in depth the many stakeholders in this industry as well as the effectiveness of anti-tracking mechanisms such as ad blockers, VPN services, and safe browsing modes.**

***Index Terms*—Privacy, Adult Content Websites, Porn Web**

Tipo de contribución: Investigación en desarrollo

I. Introduction

Porn sites are among the most visited websites globally. According to the Alexa Rank (as of March 6th, 2018) two porn sites are among the top 40 most popular websites. Despite their importance in terms of the number of on-line users, the porn web ecosystem remains mostly opaque and unexplored. Only a number of isolated efforts, mostly executed by ciber-security firms, have revealed concerning aspects of this industry. For instance, Kaspersky labs revealed that PornHub.com, one of the most popular porn websites, was responsible for distributing malware [1]. Moreover, porn websites are considered sensitive websites, due to their capacity to obtain and infer private information, such as a user's sexual orientation, habits, and behavior.

Many web users want to remain anonymous when browsing pornographic content, and resort to anti-tracking solutions or safe browsing modes to protect their privacy. However, these mechanisms may not offer them total protection. That is the case of anti-tracking solutions based on incomplete blacklists [2], or on-line services using advanced user and IP fingerprinting. Despite these issues, the academic efforts have ignored so far the tangible privacy risks of porn websites for the end-user. Instead, they have focused on measuring and characterizing how users browse and interact with those services [3], and their integration with social networks [4].

Policy makers have made important steps for regulating adult websites and protecting consumer's privacy. According to the European regulation and the upcoming one (GDPR and ePrivacy directives), it is illegal to obtain and process sensitive data without explicit consent from the users. UK legislators are putting efforts to prevent minors from accessing porn websites [5]. In their new regulatory framework, online porn services will have to verify the user's age, for instance requiring the users to provide their credit card or their national id card. Many users may consider such mechanisms intrusive, and a mean for de-anonymizing their activites. Consequently, it is unclear whether the pornographic industry will effectively implement age verification mechanisms that could negatively impact their number of daily visitors.

In this project, we will develop methodologies and techniques to uncover the online porn ecosystem. Specifically, we aim to answer the following questions:

- Can porn websites harm user's privacy?
- Do they use third-party tracking services?
- Do they interact with each other via affiliate programs?
- Are safe browsing mode and anti-trackers effective to guarantee user's anonymity?
- Do porn websites use age verification mechanisms?

II. Methodology

The first step in our methodology is selecting a large and representative corpus of porn websites. In our case we have selected 86 relevant free porn websites obtained from four different sources, specialized in recommending and classifying pornographic content[1]. Then, we use a Selenium-based crawler using a Firefox browser equiped with two add-ons: a VPN [6] for accessing the content from vantage points located in four countries (Spain, Russia, USA, and UK), to analyze geographical differences and Lightbeam [7], a tool to analyze the interactions between first and third-party services present on a given website as Falahrastegar *et al.* implemented in their previous work [8]. The crawler fetches the HTML content of the landing page and gets a screenshot of the main page. We complement our headless browser with ZBrowse [9], a command line tool based on the headless Chromium browser that allows analyzing dependencies on websites such as network

[1]https://www.alexa.com/topsites/category/Top/Adult, http://toppornsites.com, http://mypornbible.com, http://www.only4adults.net.

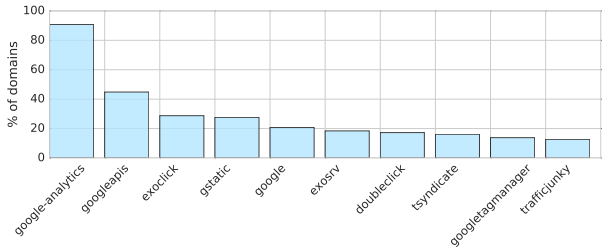


Fig. 1. Top 10 most common third-party services across porn websites.

TABLE I
PRESENCE OF COOKIES IN OUR CORPUS OF PORN WEBSITES. WE DISTINGUISH BETWEEN FIRST- AND THIRD-PARTY COOKIES.

Feature	#Porn Sites ($N = 86$)	Cookie consent
First-party Cookies	42	4
Third-party Cookies	36	0
First and Third-party Cookies	20	0

request and object dependencies. We only used ZBrowse from a vantage point located in Spain. Both crawlers create fresh browsing sessions between each page crawl to minimize bias.

III. PRELIMINARY RESULTS

The online porn network: The analysis of the HTML code of the porn websites revealed interesting links between various services. This suggests the presence of a large number of affiliated or subsidiary websites, all of them associated with the company MindGeek, which has a dominating position in the online porn industry.

Third-party services: Figure 1 presents the 10 most common third-party services over a total of 168 different ones identified in our web crawls. 19 of them are associated with advertising and tracking services, which we found on 79 porn websites. Just Google Analytics is present in 90% of the adult-content domains. We can also identify the presence of Exoclick in 30% of the porn websites. This is a Spanish ad network who, according to the BBC, specializes in distributing ads for sites and products for adults such as sexual content and gambling [10], a market intentionally ignored by Google and Facebook. This is a response to the restrictions set by countries like Germany, Russia, and China among many others who ban the distribution of adult and sensitive content through ad networks [11]. Surprisingly, Google is still present and capable of tracking porn web users through their analytics products.

HTTP Cookies: According to the European legislation, it is mandatory for all websites operating in the EU to inform users about the use of third-party cookies. In Table I we represent the presence of cookies in our set of porn sites. We found 36 porn webs installing third-party cookies without user's consent. Also, we found 42 of these domains installing first-party cookies. Only 4 sites inform users but in these cases there were only of first-party cookies. However, according to the European legislation, it is not necessary to inform and request permission from users to install first party cookies as long as it is required to deliver the expected service [12].

Age verification: We leverage our screenshots to manually inspect the presence of age-verification mechanisms in our corpus of porn sites. For this specific study, we used the VPN service to identify and report differential behavior of those porn sites due to regulatory requirements. In the USA, Spain and the UK there is only one porn site that has implemented age verification mechanism. However, in Russia we have identified 6 of them. The mechanisms to prevent access to minors are far from effective: they are simple pop ups informing that the content is restricted to adults and a checkbox for verification. We didn't find any variation in other EU countries. We can conclude that, as of today, the majority of the websites lack of effective age verification mechanisms to comply with current law requirements (as in Russia) and future ones (as in the UK).

IV. CONCLUSIONS

The online porn ecosystem remains opaque due to the lack of systematic and comprehensive studies focused on this ecosystem. In this work, we have seen that by applying simple techniques, it is possible to audit the regulatory compliance of these websites and obtain striking results. We have found the use of cookies by those websites without the user's consents and also the lack of age verification mechanism, which in some countries is already mandatory to prevent minor's access to such sensitive content. This extended abstract only presents our preliminary results in our larger efforts towards illuminating comprehensively the online porn web ecosystem and its associated privacy risks for end users. We plan to extend this work by extending the scale of our porn site corpus, the geographic differential analysis, their privacy policies, the use of advanced user-profiling and user-tracking mechanisms and the effectiveness of anti-tracking software to protect user's anonymity.

REFERENCES

[1] Kaspersky, "Malware spread through pornhub," https://www.kaspersky.com/blog/pornhub-malvertising/19698/.
[2] A. Razaghpanah, R. Nithyanand, N. Vallina-Rodriguez, S. Sundaresan, M. Allman, C. Kreibich, and P. Gill, "Apps, trackers, privacy, and regulators: A global study of the mobile tracking ecosystem," *NDSS*, 2018.
[3] G. Tyson, Y. Elkhatib, N. Sastry, and S. Uhlig, "Demystifying porn 2.0: A look into a major adult video streaming website," in *Proc of ACM IMC*, 2013.
[4] G. Tyson, Y. Elkhatib, Nishanth, and S. Uhlig, "Are people really social in porn 2.0?" in *Proc of ICWSM*, 2015.
[5] The Guardian, "Pornography sites face uk block under enhanced age controls," https://www.theguardian.com/culture/2016/nov/19/pornography-sites-face-uk-block-under-enhanced-age-controls.
[6] Firefox, "Hoxx-vpn," https://addons.mozilla.org/es/firefox/addon/hoxx-vpn-proxy/.
[7] Firefox, "Lightbeam," https://addons.mozilla.org/en-US/firefox/addon/lightbeam/.
[8] M. Falahrastegar, H. Haddadi, S. Uhlig, and R. Mortier, "Tracking personal identifiers across the web," in *PAM*, 2016.
[9] ZMap, "ZBrowse," https://github.com/zmap/zbrowse/.
[10] BBC, "Exoclick, la empresa que gano mas de 100M USD con los clientes que Google no quiso," http://www.bbc.com/mundo/noticias-39723776.
[11] Google, "Advertising policies help. adult content," https://support.google.com/adwordspolicy/answer/6023699.
[12] EU, "Cookies," http://ec.europa.eu/ipg/basics/legal/cookies/.

Privacidad en redes sociales libres. Impacto en entornos corporativos

Miguel Hernández Boza
Security Researcher
Innovation 4 Security Lab - i4s/BBVA
Av. Monforte de Lemos S/N 28029, Madrid
miguel.hernandez@innovation4security.com

Alfonso Muñoz Muñoz
Head of Cybersecurity Lab
Innovation 4 Security Lab - i4s/BBVA
Av. Monforte de Lemos S/N 28029, Madrid
alfonso.munoz@innovation4security.com

***Resumen*—Las redes sociales siempre han sido un potente altavoz para la comunicación donde se pueden mostrar opiniones de todo tipo, informar de manera rápida, conocer personas con las mismas aficiones y crear idealmente una comunidad. Esto ha ido cambiando a lo largo de los años en un modelo de negocio en el cual los datos son el principal foco de atención y punto central de toda la infraestructura. Un caso significativo son grandes compañías como *Facebook* [1] o *Twitter* [2] que venden los datos generados y facilitados por los usuarios a terceros, sugiriendo los anuncios personalizados que mejor se ajusten a cada uno. Hoy día, no es posible no compartir esta información con estas empresas. Si no se acepta las condiciones de uso las aplicaciones directamente no funcionarán. Dada esta situación han aparecido diversas alternativas libres que no tienen ninguna corporación detrás que las financien pero sirven para el mismo propósito, la comunicación. En la presente investigación, se analiza el ecosistema que forman estas redes y se profundiza en la seguridad de sus plataformas a la hora de proteger la información que se comparte, la privacidad de las comunicaciones y su capacidad frente a ataques externos. Estos factores son de especial relevancia en el uso de estas redes en entornos corporativos.**

***Index Terms*—Privacidad, Seguridad, Redes Sociales, OSINT, DoS, Fediverse, The Federation, The Activity web**

Tipo de contribución: *Investigación original*

I. Introducción

En sentido amplio, una red social es una estructura social formada por personas o entidades conectadas y unidas entre sí por algún tipo de relación o interés común. El término se atribuye a los antropólogos británicos *Alfred Radcliffe-Brown* y *Jhon Barnes* [3]. Las redes sociales son parte de nuestra vida, son la forma en la que se estructuran las relaciones personales, estamos conectados mucho antes de tener conexión a Internet. Sin embargo, el análisis de las redes sociales también ha sido llevado a cabo por otras especialidades que no pertenecen a las ciencias sociales.

Las redes sociales se pueden clasificar de diversas maneras, algunas de ellas no tienen una clasificación bien definida y se describen como horizontales. En este grupo entrarían las grandes como *Facebook* o *Google+*. Por otro lado, también están las verticales que dependen de su temática, como redes profesionales o viajes, su actividad como *blogging* o *microblogging* o su contenido, si son vídeos, fotos, texto o una combinación de ellos.

Su ideal de las 3Cs (Comunicación, nos ayudan a poner en común conocimientos; Comunidad, nos permiten encontrar e integrar comunidades; y Cooperación, nos ayudan a hacer cosas juntos, compartir y encontrar puntos de unión) se ve en ocasiones discutido por sus problemas relacionados con la privacidad, la censura de información o el negocio por encima de los derechos de los usuarios. Estos problemas son los que llevan a la realización de este estudio, al detectar diversas alternativas que cumplen con el propósito de una red social pero sin el negocio como fuente de financiación. Su seguridad será un punto importante a tener en cuenta si estas redes continúan creciendo o su uso en entornos corporativos se ve incrementado.

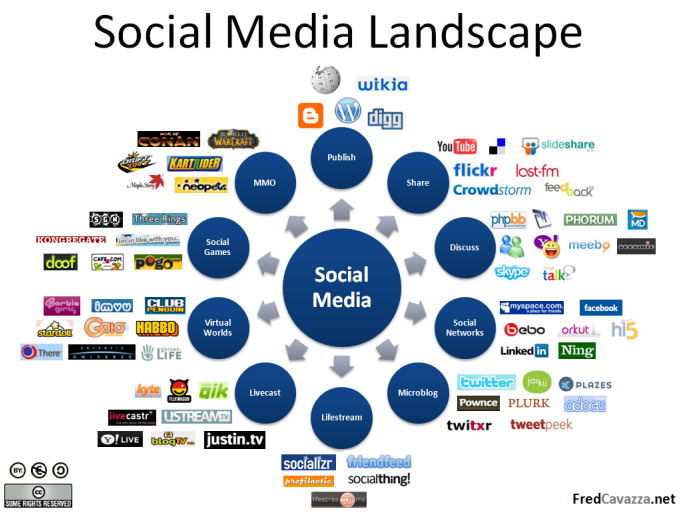


Figura 1. Tipos de redes sociales [4].

II. Redes sociales libres

Las redes sociales libres se basan en la misma premisa de las redes sociales convencionales pero a partir de software libre. Este tipo de redes se apoya en una arquitectura distribuida de servidores conectados y basada en la transparencia para el usuario.

Los servidores son llamados *pods*, *federación* o *instancia* dependiendo de la red y es la plataforma donde los usuarios se podrán registrar y conectarse para relacionarse. Estos servidores se pueden configurar con total libertad por parte del administrador. El único que controla los datos de registro es el administrador y no es necesario más que una dirección

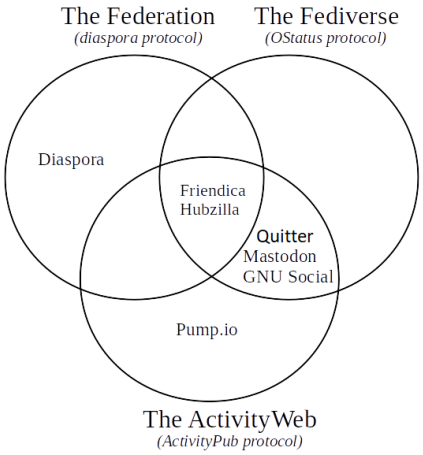


Figura 2. Protocolos y redes sociales libres [5].

de correo para acceder.

Existen tres grandes grupos donde se pueden catalogar las redes sociales libres. En la Figura 2 se agrupa las diversas redes sociales libres de mayor uso. En los siguientes apartados se analizarán en detalle desde el punto de vista de seguridad cada una. Antes de ello, seguidamente se introduce brevemente la descripción de los tipos y como se agrupan.

II-A. Fediverse

Fediverse es un acrónimo de *federación* y *universo*. Es un nombre común e informal para una federación de servidores de redes sociales cuyo objetivo principal es el **microblogging**, el intercambio de mensajes cortos y públicos. Al ejecutar un software libre de red social que admite un conjunto estándar de protocolos llamado **OStatus**, los servidores de ejecución independiente pueden conectarse a Fediverse [6], lo que permite a sus usuarios seguir y recibir mensajes cortos de otros usuarios en cualquier servidor conectado.

OStatus es un estándar abierto para actualizaciones de estado distribuidas [7], que hace referencia a un conjunto de protocolos abiertos que incluyen *Atom, Activity Streams, WebSub, Salmon y WebFinger* [8]. Este estándar permite a los diferentes centros de mensajería de microblogging conectarse a las actualizaciones de estado entre los usuarios casi en tiempo real.

La federación de OStatus fue posible primero entre las instalaciones de StatusNet [9], como *Status.net e Identi.ca*, aunque *Identi.ca* más tarde cambió a *pump.io*. A partir de junio de 2013, otras aplicaciones de microblogging y sistemas de administración de contenido anunciaron que tenían la intención de implementar el estándar, como son el caso de *Mastodon y Quitter*.

II-B. Federation

The Federation [10] se refiere a una red social global compuesta por nodos que hablan entre sí. Cada uno de ellos es una instalación de un software que admite el protocolo de diáspora. Este sitio web enumera todos los servidores registrados en el mundo para dibujar estadísticas.

Tabla I
RESUMEN DE NODOS Y USUARIOS POR PROYECTO [6] [11].

Proyecto	Nodos	Usuarios
Gnu-Social/Quitter	291	100.000
Mastodon	2400	800.000
Hubzilla	47	2.000
Pumpio	25	200
Diaspora*	150	660.000

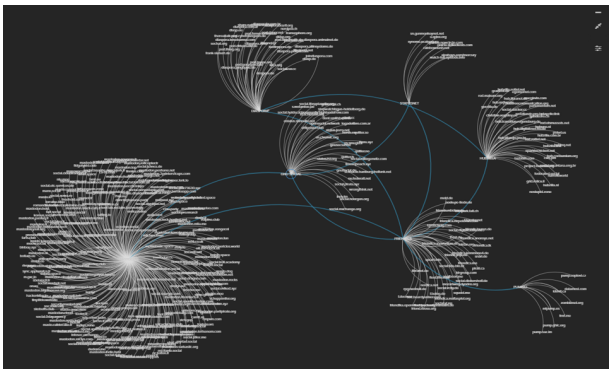


Figura 3. Mapa de redes sociales libres. Véase detallado en [12].

En este caso encontramos cuatro redes siendo *Diaspora** la más grande en cuanto a usuarios y número de comentarios, aunque también existen otras alternativas como *Friendica, Hubzilla o SocialHome*.

II-C. The Activity Web

Este grupo de redes sociales está basado en *ActivityPub*, un protocolo de red social descentralizado fundamentado en el formato de datos *ActivityStreams* 2.0 [13], desarrollándose como parte del grupo de trabajo de web social del W3C. Proporciona una API de cliente a servidor para crear, actualizar y eliminar contenido, así como una API de servidor federado a servidor para entregar notificaciones y subscribirse al contenido. Dentro de este protocolo encontramos a *Pumpio, Quitter y Mastodon*, aunque también a *Friendica y Hubzilla.*

II-D. Mapa de redes sociales libres

En la Figura 3 es posible observar como cada una de las redes esta federada con otra ya que comparten protocolo. Por lo tanto, muchos de los problemas de seguridad que se verán más adelante los comparten los diferentes proyectos y afecta a todos ellos.

En la Tabla I se puede observar que el alcance a nivel de usuarios no es tan grande como una red social convencional, pero hay que tener en cuenta que algunas de ellas como *Mastodon* son recientes y el crecimiento a nivel de usuarios ha sido exponencial, por lo que es un factor a tener en cuenta en esta investigación.

III. PROBLEMAS DE SEGURIDAD EN REDES SOCIALES LIBRES

En la presente investigación se analiza en cada una de ellas los siguientes aspectos de seguridad:

- **Protección frente a automatización en registro:** En este apartado se comprobará si se establecen medidas de seguridad para evitar la creación masiva de usuarios.
- **Privacidad de la cuenta:** Adquisición de información de un usuario dentro de un servidor y si se puede acceder a información de registro.
- **Atribución de la cuenta:** Se comprueba si es posible enlazar una cuenta con una persona real o si existen protecciones frente a la suplantación de identidad.
- **Protección del contenido de la red:** Se analiza si es posible acceder a los mensajes que se comparten dentro de la red social.
- **Protección frente a denegación de servicio:** Se analiza si es posible dejar sin acceso la plataforma para nuevos usuarios o si es posible dejar sin servicio a los usuarios ya registrados.

En la siguiente sección se analizarán todos estos aspectos para ver en que estado de seguridad se encuentran todas ellas.

IV. Auditoria de las redes sociales libres

En las líneas anteriores se han identificado las redes sociales libres de mayor uso. En el presente estudio se centrará el interés en las principales por importancia: *GNU Social - Quitter, Mastodon, Friendica - Hubzilla, Pump.io y Diaspora** (véase Tabla I). Todo el análisis de seguridad que viene a continuación está basado en el estándar de OWASP.

IV-A. GNU Social - Quitter

GNU social [14] es una continuación del proyecto *StatusNet*. Es un software de comunicación social para comunicaciones públicas y privadas. Es ampliamente compatible y tiene una gran base de usuarios, de hecho es utilizado por la Free Software Foundation [15]. Se basa en el microblogging y por un *feed* puramente cronológico de mensajes.

GNU social se inició en 2010 como un spin-off del proyecto GNU FM que se creó para impulsar comunidades de música como *Libre.fm*. En 2013, *Evan Prodromou* amablemente donó todo su trabajo en el proyecto *StatusNet* a la Free Software Foundation, por lo que se fusionó con StatusNet y otro proyecto llamado FreeSocial. *Evan* continúa desarrollando software social para la web con su nuevo proyecto *Pump.io*.

Quitter es una serie de servidores *GNU Social* sin una temática definida, aunque sí el idioma. Algunos ejemplos son:

- Quitter.es
- Quitter.cat
- Quitter.se
- Quitter.de
- Quitter.is

En este punto parece razonable que los problemas de seguridad que afecten a *GNU Social* lo harán del mismo modo a *Quitter*. En las siguientes líneas se irán analizando punto por punto los vectores de ataque destacados en el apartado anterior.

Figura 4. Ejemplo de creación de cuenta en gnusocial.de.

El primer problema de seguridad detectado en esta red está relacionado con la posibilidad de automatización de creación de usuarios. En el registro de un usuario válido se solicita una dirección de correo válida, pero no existe captcha [16] ni ninguna protección frente a la automatización, se muestra el ejemplo de registro en la Figura 4. Este hecho facilita la creación de tantos usuarios como se desee incluso utilizando servicios de correos temporales para la creación de cada cuenta/usuario, como por ejemplo el servicio YOPmail [17]. Un ejemplo de creación de una cuenta a través de curl sería la siguiente:

```
curl "https://x:x@<instancia>/api/account/register.json"
--data "nickname=<NickName>&email=<Email>
&fullname=<Fullname>^&password=<pass>
^&confirm=<Confirmpass>" --compressed
```

Una vez creada la cuenta, el correo utilizado está oculto para otros usuarios pero a partir de tu nombre de usuario se puede saber si estás registrado en la plataforma o no.

No existen las cuentas validadas dentro del proyecto, esto provoca que se pueda suplantar la identidad de otros usuarios utilizando los datos de otras redes sociales y creándolas dentro de estos proyectos.

Todo el contenido del proyecto es accesible, no es necesario estar registrado para ver los comentarios o usuarios. Si se quisieran listar todos ellos se podría realizar en orden cronológico ya que se utilizan identificadores secuenciales tanto para usuarios como para comentarios. Además es sencillo identificar a los administradores del servidor ya que tienen un identificador bajo. Unos ejemplos de peticiones serían las siguientes:

- **Primer usuario registrado:** https://quitter.es/user/1
- **Primer mensaje:** https://quitter.es/notice/1
- **Petición curl:** curl https://quitter.es/notice/1

Finalmente, al no existir protección contra la automatización, es posible denegar el acceso a nuevos usuarios al crear tantos que el servidor no soporte más. Adicionalmente, la creación automatica de usuarios facilitaría la creación de canales encubiertos de información entre usuarios y la

utilización de usuarios creados a medida para difundir código o enlaces maliciosos.

IV-B. Mastodon

Mastodon [18] es un servidor de red social de microblogging gratuito y de código abierto basado en protocolos web abiertos como *ActivityPub y OStatus*. El enfoque social del proyecto es una alternativa descentralizada viable, a los silos de medios sociales comerciales, que devuelve el control de los canales de distribución de contenido a las personas. El enfoque técnico del proyecto es una buena interfaz de usuario, una API REST limpia para aplicaciones de terceros y herramientas sólidas contra el abuso. Su servidor principal es *mastodon.social*.

En la Tabla I se verifica que esta red social es la que más impacto ha tenido por número de usuarios aún siendo un proyecto joven. Esto es debido a que hay un gran número de usuarios japoneses que lo usan ya que son flexibles con la censura [19]. Existen listados públicos de instancias que dependiendo de tus intereses recomiendan en que instancia debes registrarte [20]. A la hora de hacerlo, requiere de un correo válido donde será necesario confirmar la cuenta para que sea efectiva. Aparte de estas instancias públicas, es posible encontrar instancias que no están registradas y que no se federan con otras, ya sea porque buscan ser una comunidad más cerrada o porque los mensajes que se intercambian son de dudosa legalidad. Se realizó una búsqueda con la herramienta Shodan encontrando servidores ocultos, 1200 de ellos, algunos de prueba y otros algo más interesantes como los siguientes:

- greylog.ru
- lesbianschool.com
- anticapitalist.party

No existe ninguna protección frente a la automatización en la creación de usuarios por lo que se desarrollo un programa para verificar que es posible crear cuentas de usuario sin limitación. En la Figura 5 se pueden comprobar los pasos a seguir y una demostración del proceso de creación de cuentas se puede ver en el vídeo referenciado [21]. El proceso de creación automática de usuarios se describe en el Algoritmo 1.

Al igual que en la red social anterior analizada, no se puede ver la información privada de un usuario pero si se puede detectar si está dentro de la red social, al igual que listar todos los usuarios, ya que se numeran de manera secuencial.

- **Primer Mensaje:** https://mastodon.social/web/statuses/1
- **Primer usuario:** https://mastodon.social/web/accounts/1
- **Primer mensaje (vía API):** https://mastodon.social/api/v1/statuses/1
- **Primer usuario (vía API, requiere autenticación):** https://mastodon.social/api/v1/accounts/1

En las Figuras 6 y 7 se muestran ejemplos escritos en código Java para realizar las peticiones a las direcciones anteriormente citadas.

Algorithm 1: Creación de cuentas *Mastodon*

Entrada:
- Dirección de la instancia.
- Número de cuentas a crear.

begin
  while *Contador* $<$ *Nº cuentas* **do**
    Enviar petición GET para recoger el token de session.
    Generar nueva cuenta en Yopmail.
    Enviar petición POST del formulario con el correo generado y el token de session.
    if *Correo de confirmación* **then**
      Envío petición GET al enlace.
      Generar el token de autenticación.
    else
      Esperar a que se llegue o enviar petición para reenvio del mail.

Salida : Lista de *tokens* de acceso

No existe protección frente a la suplantación de identidad, el proyecto ya avisa en sus preguntas frecuentes que a diferencia de *Twitter* [22], donde existe un icono para validar por parte de la aplicación que ese usuario es realmente quien dice ser, en este caso se puede colocar el icono pero es simplemente eso, un icono, y no se hacen responsables. Esto puede llevar a que sea sencillo el suplantar la identidad y hacerla lo más real posible y afectar a más usuarios. En la Figura 8 se muestra una simulación.

Todos los comentarios de la red pueden ser descargados, tienen un identificador secuencial y a través de su API la descarga en sencilla y rápida (descarga en grupos de 40 usuarios). Para esta investigación se utilizó un equipo con un procesador Intel Core i7-4510U CPU 2.00GHz y en 24H fue capaz de descargar con una conexión de red de 50MB más de 14 millones de mensajes de la red social, utilizando un simple programa que hacia peticiones de manera secuencial. Se realizó el mismo proceso para usuarios y se lograron recoger unos 220.000 usuarios, una cuarta parte de todos los usuarios activos de toda la red mundial en un día. Si se aplicara un proceso más exhaustivo se podrían descargar todos los mensajes que hayan aparecido en toda la red de instancias en pocas semanas, teniendo una gran base de datos para futuros análisis, técnicas OSINT, con los problemas de privacidad que eso conlleva.

Como consecuencia del problema en la creación masiva de usuarios derivan otros problemas de seguridad. Para probar el acceso a posibles víctimas se realizó una prueba controlada y limitada de envío masivo de un enlace malicioso en todas las instancias en las que se tenía acceso. Debido a que no se pueden fijar los mensajes, son siempre cronológicos, el enlace solo estuvo unos segundos en la parte visible de las instancias grandes, pero aún así logramos una aceptación de 6.000 usuarios que accedieron en el enlace de los 800.000 posibles. Si se elaborara una campaña dirigida utilizando una

Figura 5. Esquema registro automático en red social *Mastodon*.

```java
public class GetUserInfo implements Callable<JSONObject> {
    private String Instance;
    private long id;
    private String token;
    private boolean Follow;

    public GetUserInfo(String instance2, long n, String token2,boolean follow2) {

    public JSONObject call() throws Exception {
        HttpResponse<JsonNode> jsonResponse = null;
        JSONObject infoUser = new JSONObject();
        try {
            jsonResponse = Unirest.get("https://"+Instance+"/api/v1/accounts/"+id)
                  .header("authorization", "Bearer "+ token)
                  .header("cache-control", "no-cache")
                  .asJson();

            } catch (UnirestException e) {
                System.out.println("Error en la petición");
                e.printStackTrace();
            } catch (Exception e){
                System.out.println("No existe ese usuario");
            }try{
                infoUser = jsonResponse.getBody().getObject();
            }catch (Exception e){
                System.out.println("Error");
                infoUser.append("error", "error");
            }

        return infoUser;
    }
}
```

Figura 6. Código para descargar la información de usuarios.

gran cantidad de usuarios el impacto sería mayor.

También fue posible crear un canal encubierto dentro de una instancia entre dos usuarios, utilizando las condiciones que se daban en ella, Es necesario que los dos usuarios se conozcan con anterioridad y compartan el método de cifrado pero fue posible la comunicación codificada entre dos usuarios. En nuestro ejemplo se utilizó la siguiente instancia:

- https://dolphin.town

Se realizó en esta instancia como prueba de concepto ya que utilizaba un lenguaje que forzaba la red a usar dos caracteres únicamente “e” y “E” y que no afectaría a la actividad normal de la misma. El escenario era el clásico de *Alice* y *Bob*, y en la Figura 9 es posible ver un ejemplo de esos mensajes. La codificación era simple, las “e” representaban un 0 y las “E” un 1 y los mensajes se enviaban en binario. Al tener 500 caracteres por mensaje fácilmente se enviaban enlaces, direcciones o datos sensibles.

Finalmente, el mayor problema que tiene esta red social es la escalabilidad, no soporta cuando aparece un crecimiento de usuarios y se cierran los registros para nuevos usuarios. Es posible denegar el acceso creando un gran número de usuarios generando mensajes que afecten al comportamiento normal de la red, afectando también a otras instancias que estén conectadas. En el siguiente enlace tenemos una instancia con los registros cerrados:

- https://social.targaryen.house/about

IV-C. Friendica - Hubzilla

Friendica es una red social distribuida basada en software libre y de código abierto. Enfatiza los extensos ajustes de privacidad, y la sencilla instalación. Su objetivo es ser capaz de federar con la mayor cantidad posible de redes sociales, pudiendo integrar sus contactos de *Facebook, Twitter, Diaspora, StatusNet* y otros servicios [23].

En 2012, *Macgirvin* dejó el proyecto para desarrollar una bifurcación llamada primero Redmatrix, luego *Hubzilla* [24].

```java
public class GetTimeline implements Callable<JSONArray> {
    private String Instance;
    private long id;
    private String token;

    public GetTimeline(String instance2, long n, String token2) {

    public JSONArray call() throws Exception {
        HttpResponse<JsonNode> jsonResponse = null;
        JSONArray timeline = new JSONArray();
        try {
            jsonResponse = Unirest.get("https://"+Instance+"/api/v1/timelines/public?limit=40&max_id="+id)
                    .header("authorization", "Bearer "+ token)
                    .header("cache-control", "no-cache")
                    .asJson();

            } catch (UnirestException e) {
                e.printStackTrace();
            } catch (Exception e){
                e.printStackTrace();
            }
        try{
            timeline = jsonResponse.getBody().getArray();
        }catch (Exception e){
            System.out.println("Error");
            JSONObject error = new JSONObject();
            error.put("error", "error");
            timeline.put(error);
        }

        return timeline;
    }
}
```

Figura 7. Código para descargar la información de mensajes.

Figura 8. Simulación de una suplantación.

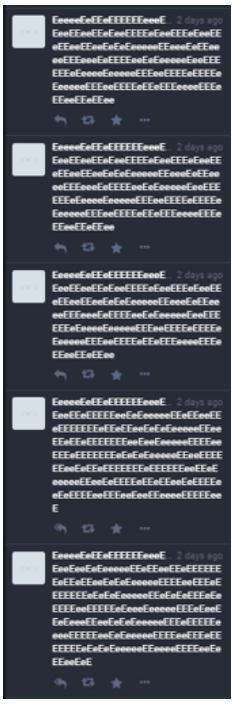


Figura 9. Mensajes codificados en *Mastodon*.

A diferencia de *Friendica*, *Hubzilla* no solamente permite crear una red social distribuida, sino que también actúa como sistema de gestión de contenidos para distintos tipos de sitios y comunidades web cuyos usuarios y canales se pueden conectar entre sí. Actualmente Friendica se desarrolla por la comunidad por lo que se centra el interés en *Hubzilla*.

En este caso, es un proyecto en crecimiento y no existe demasiada actividad, en la parte de registro es necesario confirmar la dirección de correo pero no existe ninguna protección frente a la automatización. Es posible acceder a un servidor a través de otro cediendo la autenticación y con ello accediendo a cualquiera de ellos aunque no estén los registros abiertos. El contenido de los mensajes sigue siendo público para todos los usuarios sin necesidad de registro y en el caso de la privacidad de usuario no se ve comprometida en ningún momento.

Al no tener protecciones frente a la automatización, es vulnerable a denegación de servicio por la creación masiva de usuarios o mensajes. Un listado de los usuarios y mensajes de una instancia sería las siguientes consultas:

- GET https://hub.bodalenz.de/directory
- GET https://hub.bodalenz.de/display/1

*IV-D. Diaspora**

*Diaspora** es un software libre para servidores web que permite implementar redes sociales descentralizadas y privadas, como alternativa a Facebook entre otras. Desarrollada por *Dan Grippi, Max Salzberg, Raphael Sofaer y Ilya Zhitomirskiy* entre otros a través de listas de correo y financiado por Kickstarter donde incluso *Marck Zuckerberg* aportó una importante cantidad de dinero [25]. Actualmente,

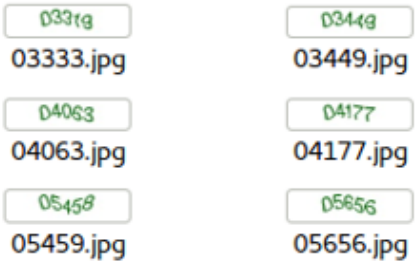


Figura 10. Captcha generado por la red Diaspora y texto detectado.

el código se desarrolla por la comunidad. La versión del código esta en la 5 y se pretendía que tuviera capacidad para juegos, muros y chat pero todo ello alojado en un servidor web del usuario, para que sea el usuario el que pueda controlar la información que se comparte.

Es el primer proyecto que pone una protección frente a la automatización utilizando un captcha, el cuál se analizó para ver su robustez. Está basado en la generación de una imagen con un número que hay que escribir en el formulario. Este número está dentro de un rango, del 00000 al 99999. Toda la combinación de captchas posibles es posible almacenarla realizando peticiones a la siguiente dirección:

- Dirección: https://joindiaspora.com/simple_captcha?
- Identificador de la imagen: code=acfc9453f03a27320e9d1e45cd6ee735663b5410
- Ruido: time=1511392460 (Es posible eliminarlo)

Es posible generar todo el dataset de imágenes y siempre una misma imagen corresponde con un número en concreto (la imagen no cambia en cada petición para el mismo número a proteger). Para demostrar la mala decisión en la elección de este tipo de captcha se utilizó un sistema de reconocimineto OCR muy común en Linux. Con el software Tesseract OCR [26] se consiguió sin ningún esfuerzo romper el 50 % de los captchas.

En la Figura 10 se muestra un ejemplo de la salida que se obtuvo. Con esto era posible generar todas las cuentas que se desearan, saltándose la protección. No hay limitación en el envío de peticiones con lo que simplemente se repetía la petición con un nuevo captcha en el caso de que el OCR no acertara. Por tanto, es posible automatizar el registro facilitando incluso la denegación al acceso a nuevos usuarios (saturación de base de datos).

No existe protección frente a la suplantación de identidad ya que prima la privacidad de las cuentas, no es posible ver los datos de registro de los usuarios, solo el nombre de usuario. Es un problema genérico en todas estas redes sociales libres.

El contenido es accesible por todos los usuarios sin necesidad de registro, los mensajes tienen un identificador secuencial por lo que se puede ir monitorizando todos los mensajes desde el inicio del servidor.

Un listado de los usuarios y mensajes de una instancia se obtendría las siguientes consultas:

- curl https://joindiaspora.com/people/{hash}
- curl https://joindiaspora.com/posts/{id-acumulativo}

IV-E. Pump.io

Pump.io es un motor de secuencias de actividad de propósito general que se puede utilizar como un protocolo de red social federado que “hace la mayor parte de lo que la gente realmente quiere de una red social” [27]. Iniciado por *Evan Prodromou*, es un seguimiento de *StatusNet*; *Identi.ca*, que era el servicio más grande de *StatusNet*, cambió a *Pump.io* en junio de 2013 [28]. Sin embargo, aunque *StatusNet* ofrece una funcionalidad similar a la de Twitter, *Pump.io* ofrece redes sociales mucho más generales y está siendo adoptada por otros tipos de aplicaciones web, como *MediaGoblin* [29].

En el caso de pump.io el registro de usuarios depende del servidor, es necesario una dirección de correo válida o no, aunque en cualquier caso no existe ninguna protección frente a la automatización.

No es posible sacar la información de los usuarios pero si existe un listado con qué usuarios están registrados en esta red social [30], por lo que se podrían listar de manera sencilla. El contenido también es accesible para usuarios no registrados al igual que ver la información que muestre el usuario como pública.

Finalmente hay redes donde ya no es posible el acceso por la gran cantidad de usuarios que hay registrados por lo que te obligan a registrarte a través de otros servidores, estos son los que se atacarían con la creación masiva de usuarios para que no sea posible de acceder por parte de los usuarios.

V. Resumen del impacto en su seguridad

En la tabla II se ha resumido todos los problemas que se han ido enumerando para cada uno de los proyectos.

Por definición, los proyectos se basan en la privacidad del usuario y no hay forma de atribuir una identidad física a la de un usuario registrado en la red, el único que puede ver el correo de registro es el administrador y no hay limitaciones a la hora de usar servidores temporales de correo. Esto produce también un problema derivado de seguridad como es la suplantación de identidad, ya que es posible crear una cuenta de un usuario y no existe forma de detectar quién hay detrás.

El mayor problema es la inexistente protección frente a la automatización, pudiendo causar denegaciones de servicio en toda una red debido a la generación masiva de usuarios o por el envío de mensajes basura dentro de todas las federaciones.

Finalmente, al tener acceso a múltiples usuarios, se analizó la creación de un canal encubierto dentro de una instancia de *Mastodon*, utilizando un lenguaje que conocían las dos partes.

Tabla II
RESUMEN DE LOS PROBLEMAS ENCONTRADOS Y PLATAFORMAS AFECTADAS.

Proyecto/Problema	Automatización	Privacidad	Atribución	DoS
Gnu-Social/Quitter	v	v	x	v
Mastodon	v	v	x	v
Hubzilla	v	v	x	v
Pumpio	v	v	x	v
Diaspora*	v	v	x	v

V-A. Contramedidas y recomendaciones

El mayor problema de las redes sociales libres está relacionado con un deficiente proceso de registro de usuarios desde el punto de vista de seguridad. No existen mecanismos robustos de antiautomatización de alta de usuarios, ya sea mediante captchas más robustos o la detección de uso de cuentas de correo de uso temporal.

Por otro lado, las APIs disponibles dentro de cada proyecto no tienen ningún control de su uso, no hay limitación. Existe la posibilidad de realizar peticiones todo el tiempo hasta que el servidor no pueda soportar más, pudiendo espiar a los usuarios de manera global en la red, así como pudiendo dejar sin servicio al servidor, y por lo tanto a la red social. Es imperativo el poner límites al uso de estas APIs para evitar este tipo de problemas asociado, por ejemplo, a un token de uso.

Estos dos puntos son los más importantes. Los autores de esta investigación notificaron estos descubrimientos a los creadores de estos proyectos para que implementaran estas protecciones, dado que en total estaría afectados alrededor de un millón y medio de usuarios únicos.

VI. CONCLUSIONES

En este trabajo se presenta la primera investigación sobre la seguridad de las principales redes sociales libres.

A la vista de los resultados expuestos, se puede observar como muchos problemas de seguridad clásicos afectan a estas nuevas redes sociales libres, problemas que se superaron hace tiempo en las redes sociales comerciales. Principalmente entre sus defectos destaca la inexistente protección frente a la automatización, tanto en la monitorización de todo el contenido como a la hora de registrar usuarios. Solo en una red social libre de gran difusión, como es el caso de Diaspora*, se ha implementado un captcha, que además no es robusto y se ha podido verificar como evadirlo con un simple software de reconocimiento de imágenes (OCR).

Los detalles de seguridad analizados no solo deben ser considerados a nivel particular como usuarios de estas redes, sino especialmente si se decide desplegar estas redes sociales (código libre) de manera interna en organizaciones, dado que son una herramienta muy potente de compartición y colaboración. Como se demuestra en este trabajo elegir este tipo de tecnología tiene implicaciones en privacidad, monitorización, disponibilidad del servicio y distribución de código malicioso, factores que no pueden pasar desapercibidos.
Independientemente de todo lo justificado en este documento, las redes sociales libres son una interesante alternativa buscando la privacidad y la pluralidad de opiniones a las redes sociales convencionales. No son perfectas, todavía están lejos de tener estándares de seguridad o escalabilidad similares a las opciones comerciales. No obstante, este trabajo intenta proporcionar un grano de ayuda a mejorar la tecnología facilitando su llegada a un mayor número de usuarios.

REFERENCIAS

[1] TEKNAUTAS, El confidencial, https://www.elconfidencial.com/tecnologia/2017-09-11/datos-facebook-multa-espana-ilegal_1441670/. Última actualización 11.09.2017 – 05:00 H.
[2] Configuración Twitter https://twitter.com/personalization Último acceso 15.02.2018
[3] Isabel Ponce. «Monográfico: Redes sociales». Ministerio de Educación, Cultura y Deporte. Consultado el 28 de octubre de 2017.
[4] Imagen de Fredcavazza; @flickr, https://www.flickr.com/photos/fredcavazza/2564571564/ Subida el 09.06.2008
[5] Sean Tilley, https://medium.com/we-distribute/a-quick-guide-to-the-free-network-c069309f334 - Fecha publicación 23.09.2017
[6] Red Fediverse 2017 https://fediverse.kranglabs.com/ Último acceso 15.02.2018
[7] Ostatus 1.0 August 30, 2010 https://www.w3.org/community/ostatus/wiki/images/9/93/OStatus_1.0_Draft_2.pdf
[8] Rahsheen, https://www.blackweb20.com/2010/03/09/statusnet-cloud-service/ - Publicado el 03.09.10
[9] Autor Nathan Willis https://lwn.net/Articles/544347/ - Fecha publicación 27.03.2013
[10] What is The Federation. https://the-federation.info/
[11] Red Federation 2017 https://the-federation.info/ - Último acceso 15.02.2018
[12] @wakest@mastodon.social, https://kumu.io/wakest/fediverse - Último acceso 15.02.2018
[13] Activity Streams 2.0 W3C Recommendation 23 May 2017 https://www.w3.org/TR/activitystreams-core/
[14] Página principal GNU - Social https://gnu.io/social/ Último acceso 15.02.2018
[15] Matt Lee, https://www.fsf.org/es - Publicado el 03.12.2010 12:38
[16] von Ahn, Luis; Blum, Manuel; Hopper, Nicholas J.; Langford, John (May 2003). CAPTCHA: Using Hard AI Problems for Security. EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques.von Ahn, Luis; Blum, Manuel; Hopper, Nicholas J.; Langford, John (May 2003). CAPTCHA: Using Hard AI Problems for Security. EUROCRYPT 2003: International Conference on the Theory and Applications of Cryptographic Techniques.
[17] Página oficial servidor temporal de correo Yopmail http://www.yopmail.com/es/ Último acceso 15.02.2018
[18] «What Is Mastodon and Will It Kill Twitter?» . PCMAG (en inglés). Consultado el 6 de abril de 2017.
[19] Gabriela González, https://www.genbeta.com/a-fondo/ como-mastodon-el-ultimo-clon-de-twitter- ha-triunfado-en-japon-gracias-al-lolicon 22 Agosto 2017 - Actualizado 23 Agosto 2017, 13:37, Último acceso 15.02.2018
[20] Listado público instancias, https://instances.social/
[21] Miguel Hernández, Creación de cuentas en Mastodon, Youtube https://www.youtube.com/watch?v=7tBkh_lPd1Y - Último acceso 15.02.2018
[22] Página ayuda Twitter https://help.twitter.com/es/managing-your-account/about-twitter-verified-accounts Último acceso 15.02.2018
[23] Friendica (ed.) Archivado desde el original el 11 de abril de 2012. Consultado 15.02.2018.
[24] Página principal, https://project.hubzilla.org/page/hubzilla/hubzilla-project - Último acceso 16.02.2018
[25] Decentralize the web with Diaspora, http://www.kickstarter.com/projects/196017994/diaspora-the-personally-controlled-do-it-all-distr - Último acceso 15.02.2018
[26] Código disponible, https://github.com/tesseract-ocr/tesseract - Último acceso 16.02.2018
[27] Página principal pumpio, http://pump.io/ - Último acceso 15.02.2018
[28] Nathan Willis, (March 27, 2013). "StatusNet, Identi.ca, and transitioning to pump.io" LWN.net. Retrieved 2014-03-20
[29] Christopher Allan Webber, (October 24, 2013). "Pump API progress video" mediagoblin.org. Retrieved 2014-03-22.
[30] Listado usuarios, https://www.inventati.org/ppump/usuarios/ - Último acceso 15.02.2018

A summary of "Real-time Electrocardiogram Streams for Continuous Authentication"

Carmen Camara
U. Carlos III de Madrid
macamara@pa.uc3m.es

Pedro Peris-Lopez
U. Carlos III de Madrid
pperis@inf.uc3m.es

Lorena Gonzalez-Manzano
U. Carlos III de Madrid
lgmanzan@inf.uc3m.es

Juan Tapiador
U. Carlos III de Madrid
jestevez@inf.uc3m.es

***Abstract*—Security issues are becoming critical in modern smart systems. Particularly, ensuring that only legitimate users get access to them is essential. New access control systems must rely on Continuous Authentication (CA) to provide higher security level. To achive this, recent research has shown how biological signals, such as Electroencephalograms (EEGs) or Electrocardiograms (ECGs), can be useful for this purpose. In this paper we introduce a new CA scheme that, contrarily to previous works in this area, considers ECG signals as continuous data streams. The data stream paradigm is suitable for this scenario since algorithms tailored for data streams can cope with continuous data of a theoretical infinite length and with a certain variability. The proposed ECG-based CA system is intended for real-time applications and is able to offer an accuracy up to 96%, with an almost perfect system performance (kappa statistic $>$ 80%).**

***Index Terms*—Biometry, Electrocardiography, Datastreams, Cybersecurity in e-health**

Type of contribution: *Published article (max. 2 pages)*

I. Introduction

Although some authors have already explored the problem of continuous authentication using cardiac signals (e.g., ECG [1] and PPG [2]), the used datasets are made up of records with length of only a few minutes). In our opinion, a data stream approach fits better the problem of CA, particularly in the case of ECG signals–and, more generally, physiological signals with a slight variability and a theoretical infinite length. In this paper, we introduce a new CA scheme using this paradigm.

We consider the typical assumptions for classification in the Data Stream Mining (DSM) setting [3]: 1) each sample has a fixed number of attributes that are less than several hundreds; 2) the number of classes is limited and small (in our experiments, ten classes are considered at maximum); 3) we assume that the learner has a small memory; the size of the training dataset is larger than the available memory; and finally, 4) the speed rate of processing each sample is moderate high (the precise value is conditioned to the device that supports on-board the learner). Physiological signals recordings were taken during a maximum period of 24 hours in the best case [4]. The execution time of the algorithm used scales linearly with the number of examples.

II. Experimentation & Results

Although important variations on ECG streams only occur after 5 years observation period [5], we can find slight variations from time to time —the nature of ECGs is non stationary. This is often referred as concept drift. To dealt with this, old instances should become irrelevant to characterize the current state of the system and this information would have to be forgotten by the learner. The interested reader can consult [6] for a detailed explanation of the main existing approaches in the literature. In our particular case, we keep only the most recent samples in memory and the memory size is fixed —sliding window strategy.

Aside from using a limited memory, we can benefit from drift detection by resetying the learner model and triggering the learning when a significant change is detected. We have tested two well-known methods: Drift Detection Method (DDM) and Early Drift Dection Method (EDDM) [7]. In a nutshell, DDM is based on monitoring the number of errors produced by the learner during prediction —errors are modelled by a binomial distribution. DDM performs well to detect abrupt changes and not very slow gradual changes. EEDM was proposed with the aim of improving the detection of gradual changes and keeping a good performance with abrupt changes. Instead of considering only the number of errors as in DDM, it also takes into account the distance between two classification errors.

The performance of the two aforesaid methods has been evaluated with one of the subjects of the CA (unbuffered approach – instances are sent to the learner in real time) setting which is our more demanding scenario. The subject 9 has been selected for this experimentation without prejudice to the generality in the results. More precisely, DDM and EDDM algorithms are used as a wrapper on the KNN learner. We have tested two scenarios: 1) the original data stream; 2) artificial noise has been added to the original data —10% and 5% are the fractions of attributes values and class labels that have been disturbed, respectively. Figure 1 displays the obtained results and Table I summarizes the average values. In terms of performance, the KNN with drift detection marginally improves our previous results of only using a KNN with sliding window. In addition, drift detection methods work well even when the data streams are quite noisy —the performance only suffers a brief dip. Note that we have overstated the used example since the noise remains during the whole data stream and often it is intermittent.

Finally, a key-aspect in the processing of cardiac signals is the time period during which the ECG is observed. In the buffered approach, each stream is linked with the observation of the ECG during a moderate long time period with the extra benefit of achieving a very high performance. In the unbuffered approach, the sending of the samples to the learner is almost instantaneous with the penalty of a slightly degradation of the performance in comparison with the buffered

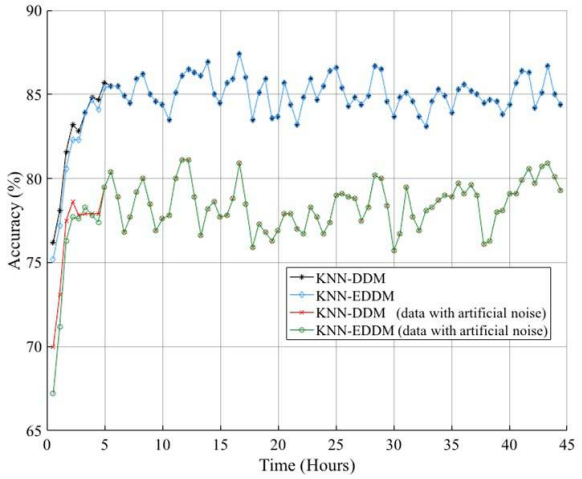


Figure 1. System performance: CA (unbuffered approach) with drift detection.

Table I
AVERAGE PERFORMANCE: CA (UNBUFFERED APPROACH) WITH DRIFT DETECTION

Approach	**Accuracy (average value)**
KNN	84.1200 ± 1.5095
KNN-DDM	84.8000 ±1.6290
KNN-EDDM	84.7300 ±1.7921
KNN-DDM (with artificial noise)	78.2500 ± 1.7060
KNN-EDDM (with artificial noise)	78.1600 ± 1.9831

approach. The choice of one approach or another would be conditioned by the processing speed rate demanded by the learner. In our particular case (a CA system), we have the possibility to check the credentials of an individual almost instantaneously (each two seconds) or just remain patient and proceed with the verification once every three minutes. In Table II we show a comparison of both approaches.

III. CONCLUSIONS

We are currently in an era in which our surrounding devices generate and transmit data in a continuous way. An example of these devices are those belonging to the Internet-of-Things (IoT) or the new generations of Implantable Medical Devices (IMDs) with wireless connectivity. These devices receive data continuously and very frequently in a non-orderly fashion. One use of such data is user authentication. In particular, the use of biological signals has been previously studied for authentication purposes. Cardiac signals (PPG or ECG) and brain signals (EEG) collected from IMDs or body sensors, are widely used for authentication and some authors have applied them to the CA scenario [2], [8]. However, given the continuous nature of the authentication process, the system has to be able to adapt itself to changes; for example, ECG signal may slightly change over time. Thus, DSM emerges as a promising technique to face this sort of problems. To the best of our knowledge, none of the existing solutions use ECG signals as data streams.

We exploit the full potential of DSM for designing a CA system using ECG streams. The proposed real-time system has been evaluated using records of 10 individuals monitored during approximately half a day. Our results show the potential of ECG streams for security purposes. In fact, the

Table II
CA: UNBUFFERED AND BUFFERED APPROACH.

	Unbuffered Approach		Buffered Approach	
Subject	Average Accuracy	Average Kappa	Average Accuracy	Average Kappa
S1	81.98	63.95	96.80	93.59
S2	81.39	62.79	95.10	90.19
S3	79.28	58.57	91.32	82.50
S4	76.00	52.05	97.34	94.68
S5	77.81	55.58	94.06	88.13
S6	72.75	45.49	92.09	84.09
S7	81.02	62.04	94.80	89.62
S8	77.17	54.35	94.35	88.69
S9	84.12	68.18	97.95	95.90
S10	73.51	47.12	94.04	88.05
Average	78.50	57.01	94.79	89.54

behaviour of the classifier, which is the core of the CA system, is almost perfect. Moreover, we have tested the buffered and unbuffered approaches in the CA setting to show how the use of one or another is driven by the requirements of the real time application (e.g., credentials/second that must be checked by the CA system). Finally, we have studied how drift detection techniques (e.g., DDM or EDDM) may help to deal with the existing changes in the ECG data streams —a wrapper approach has been tested. The results clearly indicate that drift detection techniques are effective to build robust CA schemes even under very noisy conditions.

As a future work, we plan to check whether the concept of ECG streams can be extended to other physiological signals. We hope this contribution can serve as seed to many other works that explore the use of biological signals for continuous authentication.

REFERENCES

[1] M. Guennoun, N. Abbad, J. Talom, S. M. M. Rahman, and K. El-Khatib, "Continuous authentication by electrocardiogram data," in *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto international conference*. IEEE, 2009, pp. 40–42.

[2] A. Bonissi, R. D. Labati, L. Perico, R. Sassi, F. Scotti, and L. Sparagino, "A preliminary study on continuous authentication methods for photoplethysmographic biometrics," in *IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications (BioMS)*, 2013, pp. 28–33.

[3] G. Bifet, A. Holmes, R. Kirkby, and B. Pfahringer, "Data stream mining: A practical approach. Technical report. University of Waikato," http://www.cs.waikato.ac.nz/~abifet/MOA/StreamMining.pdf, 2012. [Online]. Available: \url{http://www.cs.waikato.ac.nz/~abifet/MOA/StreamMining.pdf}

[4] PhysioNet, "Physiobank. National Institute of General Medical Sciences (NIGMS) and the National Institute of Biomedical Imaging and Bioengineering (NIBIB)," https://physionet.org/physiobank/, 2017.

[5] C. Camara, P. Peris-Lopez, and J. E. Tapiador, "Human identification using compressed ecg signals," *Journal of Medical Systems*, vol. 39, no. 11, pp. 1–10, 2015.

[6] J. Gama, *Knowledge Discovery from Data Streams*. Chapman and Hall/CRC, 2010.

[7] P. M. Gonçalves, S. G. de Carvalho Santos, R. S. Barros, and D. C. L. Vieira, "A comparative study on concept drift detectors," *Expert Systems with Applications*, vol. 41, no. 18, pp. 8144 – 8156, 2014.

[8] D. P. Coutinho, A. L. N. Fred, and M. A. T. Figueiredo, "Ecg-based continuous authentication system using adaptive string matching." in *Biosignals*, 2011, pp. 354–359.

Un resumen de: Federated system-to-service authentication and authorization combining PUFs and tokens

Publicado en el 12th International Symposium on Reconfigurable Communication-centric

Systems-on-Chip (ReCoSoC 2017)

Marta Beltrán, Miguel Calvo and Sergio González
Universidad Rey Juan Carlos, Madrid
marta.beltran@urjc.es, miguel.calvo@urjc.es and s.gonzalezmej@alumnos.urjc.es

***Abstract*—Different application domains are challenging the still immature access control mechanisms currently used to authenticate and to authorize system-on-chip architectures to services deployed locally or in the cloud. These domains include Internet of Things, Smart Places or Industry 4.0 where different kinds of devices and objects, often poorly physically protected, low-cost and energy-constrained, interact with different kinds of services through lightweight communication protocols. These protocols usually guarantee basic data confidentiality and integrity, securing communication channels using cryptography, but there are still important challenges related to authentication and authorization. This work proposes a new system-to-service authentication and authorization mechanism based on the combination of a Physical Unclonable Function (PUF) and two tokens (one devoted to authentication and the other devoted to authorization), capable of working over HTTP or COAP relying on federated schemes and adapted to the specific requirements of this kind of environments.**

***Index Terms*—Authentication; Authorization; Federated access control; Physical Unclonable Function (PUF); Tokens**

Tipo de contribución: *Investigación ya publicada*

I. Introduction

Physical Unclonable Functions (PUFs) are a promising alternative to solve authentication of integrated circuits thanks to the analogy between the circuit's "fingerprint" caused by manufacturing processes variation and human biometrics. A PUF is, basically, a binary function with an input-output behaviour determined by unique manufacturing features which generates challenge-response pairs [1]. This enables the secure confirmation of the authenticity of different kinds of tags, chips, cards and boards and the detection of tampering, substitution or spoofing of such systems.

PUF-based authentication mechanisms provide implicit or explicit identification depending on their underlying protocols and they usually implement implicit naive authorization: once a system is authenticated to a service, it is supposed to have full privileges. But, what if fine-grained authorization decisions are needed? What if privileges dynamically change or access policies need to be managed centrally? Current PUF-based mechanisms do not solve all these challenges.

The main contribution of this work is the definition of a new federated system-to-service authentication and authorization mechanism combining PUFs and tokens. Specifically, we (a) select a federated token-based scheme, OpenID Connect/Mobile Connect and analyse its applicability to the considered context (b) propose a new scheme, based on this analysis, to combine the underlying concepts of current token-based mechanisms and PUF-based protocols (c) consider energy and resource constraints, making this scheme compatible with COAP (Constrained Application Protocol). A real implementation of the proposed mechanism in a real healthcare use case can be found in the original paper published in the ReCoSoc 2017 proceedings. This implementation allowed us to validate and to assess security and performance of our proposal.

II. Proposed system-to-service authentication and authorization mechanism

A. Problem formulation

The proposed mechanism defines three roles. First, the System that needs to access a service with verified identity, it is supposed to have energy and resource constraints. Second, the Service i.e. the Relying Party (RP) which is the entity needing to verify the identity of the System before granting an authorization to perform some task. This Service can also have energy and resource constraints or, on the contrary, it can be running on a traditional resource-rich server. Third, the Identity Provider (IdP), which is the entity that is able to verify the System's identity. This role is supposed to be always running on full IT infrastructure and it could be adopted by the system manufacturer (but not exclusively, other third parties could work as identity providers such as security services providers or telecommunication operators).

Before a Service can be involved in an authentication/authorization flow it must first register with the Identity Provider associated with the System. The Registration process can be performed statically or dynamically (with the same mechanisms proposed by the OpenID Connect specification) and it assigns the Service a Service Identifier and a Service Secret (a password).

In the same way, the System must be registered with the Identity Provider. During this Registration process, the Identity Provider registers a System's physical identifier (explained later, in the next subsection), assigns an internal unique Object Identifier, and, if using a strong PUF, stores a set of challenge-response pairs in a database. Note that this process

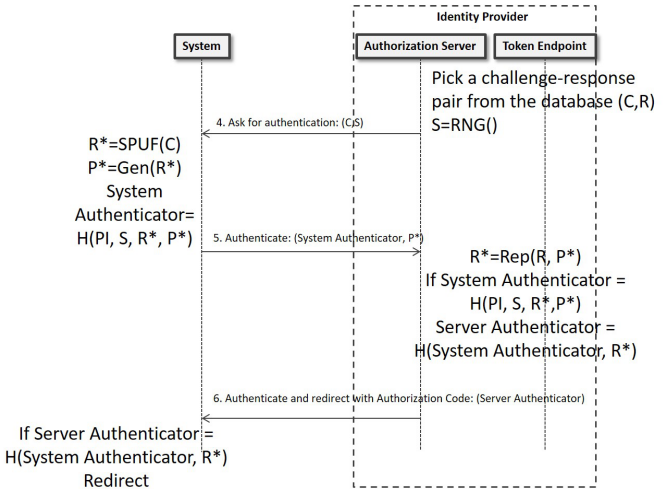


Fig. 1. Detail of PUF-based identification and authentication using Reverse Fuzzy Extractor with strong PUF (SPUF). C=challenge, R=Response, PI=physical identifier stored in a non volatile memory, S=seed, Gen()=generation function, Rep()=reproduction function, H()=hash function, RNG()=random number generation function

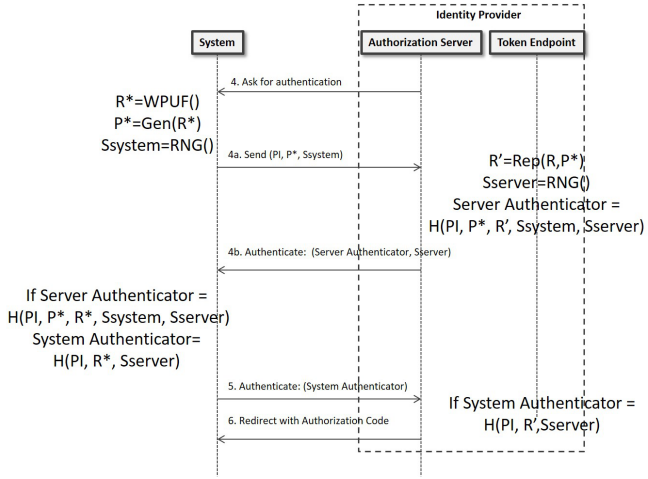


Fig. 2. Detail of PUF-based identification and authentication using Reverse Fuzzy Extractor with weak PUF (WPUF). R=Unique Response, PI=physical identifier stored in a non volatile memory, S=seed, Gen()=generation function, Rep()=reproduction function, H()=hash function, RNG()=random number generation function

corresponds to the first phase of any PUF-based authentication mechanism, the one-time enrolment in a secure environment performed right after manufacturing.

B. Full flow (over HTTP)

This flow is equivalent to the flow followed when using OpenID Connect 1.0 [2] to authenticate an end user to a traditional cloud service. The main differences with the original flow of OpenID Connect 1.0 are the Authorization Request and tokens structures, the mechanisms used to authenticate the System to the Identity Provider (based on a PUF and providing mutual authentication to avoid Identity provider spoofing) and our attempt to keep the flow simple in order to deal with potential energy and resource constraints. The Full flow works over HTTP (and communication with the Authorization Server and the Token endpoint must rely on TLS), therefore it is assumed that these constraints are not too severe. The Lightweight flow, defined in the next subsection, will be able to deal with severe constraints.

In this work we propose to perform the authentication of the System to the Authorization Server (in fact, mutual authentication) using a PUF-based mechanism, specifically we recommend to use Reverse Fuzzy Extractors with a strong PUF [3] (during validation experiments different PDL arbiter PUFs were tested). The detailed PUF-based authentication flow is shown in Figure 1. To avoid the Authorization Server polling to establish a match among all the enrolled systems, each system must store a physical identifier in an insecure non volatile memory (this physical identifier can be public), enabling explicit identification before authentication to decrease undesired latencies. The physical identifier is revealed to the Identity Provider and registered during the already explained Registration process.

C. Lightweight flow (over COAP)

In this case the application protocol is COAP instead of HTTP and it is recommended to use a weak PUF [4], more suitable for resource-constrained environments. First of all the System is, and the Service may be, a constrained device, and resource and/or energy constraints may limit the communication security protocols they support. Both roles must specify during the Registration or enrolment process with the Identity Provider which protocols they are able to support in order to secure their communications with the Authorization Server and Token Endpoint: TLS when using HTTP (for the Full flow or for communications between the Service and the Identity Provider in the Lightweight flow if the Service is executing on resource-rich environments), DTLS (PreSharedKey, RawPublicKey and/or Certificate) or Object Security (OSCOAP and/or OSCON) when using COAP.

The detailed PUF-based authentication flow in this case is shown in Figure 2, based on the selected weak PUF (in the original paper details about this alternative, a butterfly PUF, can be found) and using COAP as application-level communication protocol (and DTLS or Object Security to protect the communication channel between the System and the Authorization Server).

III. Conclusion

The proposed mechanism is general (it can be used with different kinds of systems and services in different environments) and flexible (capable of working with HTTP and COAP and of relying on different kinds of PUFs) enough to solve authentication and authorization in almost all system-to-service Internet of Things, Smart Places or Industry 4.0 scenarios.

References

[1] U. Rührmair and D. E. Holcomb, "PUFs at a glance," in *Proceedings of the Conference on Design, Automation & Test in Europe*, 2014, pp. 347:1–347:6.

[2] "OpenID Connect 1.0," http://openid.net/connect/.

[3] A. V. Herrewege *et al.*, "Reverse fuzzy extractors: Enabling lightweight mutual authentication for PUF-enabled RFIDs," in *16th International Conference on Financial Cryptography and Data Security*, 2012, pp. 374–389.

[4] R. Maes, "Physically unclonable functions: Constructions, properties and applications," Ph.D. dissertation, KU Leuven, 2012.

A Review of "Security Assessment of the Spanish Contactless Identity Card"

Ricardo J. Rodríguez
Centro Universitario de la Defensa de Zaragoza, Spain
Università degli Studi della Campania "Luigi Vanvitelli", Italy
Email: rjrodriguez@unizar.es

Juan Carlos Garcia-Escartin
Dpto. Teoría de la Señal y Comunicaciones e Ing. Telemática
Universidad de Valladolid, Spain
Email: juagar@tel.uva.es

***Abstract*—The theft of personal information to fake the identity of a person is a common threat normally performed by individual criminals, terrorists, or crime rings to commit fraud or other felonies. Recently, the Spanish identity card, which provides enough information to hire on-line products such as mortgages or loans, was updated to incorporate a Near Field Communication (NFC) chip as electronic passports do. This contactless interface brings a new attack vector for criminals, who might take advantage of the RFID communication to virtually steal personal information. In this paper, we consider as case study the recently deployed contactless Spanish identity card assessing its security against identity theft. In particular, we evaluated the security of one of the contactless access protocol as implemented in the contactless Spanish identity card, and found that no defenses against on-line brute-force attacks were incorporated. We then suggest two countermeasures to protect against these attacks. Furthermore, we also analyzed the pseudo-random number generator within the card, which passed all the performed tests with good results.**

***Index Terms*—NFC, identity theft, security assessment**

Tipo de contribución: *Investigación ya publicada*

I. Extended Abstract

Identity theft is defined as the theft of personal information, such as name, date of birth, etc. – that is, any data that allows a party to fake the identity of other party [1]. Each country defines different laws that protect their citizens from this kind of theft. For instance, the Spanish law punishes the use of personal information to fake the identity of an individual and perform actions on its behalf with up to 3 years of prison [2].

In Spain, this personal information is collected in the Spanish identification (ID) card, abbreviated as DNI (*Documento Nacional de Identidad*, in Spanish), which is issued to any Spanish citizen. Data contained on this card are, among others, the first name, the family names, the unique identification number of the citizen, and the birth date.

An identity theft is normally performed by an individual criminal, a terrorist, or a crime ring, who will take advantage of the identity to commit fraud or other felonies [3]. In Spain, data written on the DNI are enough to hire different on-line products (such as telecommunication services, mortgages, or loans). Some reports quantified a total of 4.5 million of these cases in Spain, with an average fraud of 8000€ per case [4]. Examples of felonies performed by criminals after the theft of Spanish ID cards are reported in [5].

Recently, the DNI was updated to incorporate a Near Field Communication (NFC) chip, as electronic passports (e-passports, for short) do [6]. NFC is a bidirectional short-range (up to 10 cm) contactless communication technology operating in the 13.56 MHz band based on the ISO-14443 [7] and the Sony FeLiCa [8] Radio Frequency Identification (RFID) standards. NFC is vulnerable to multiple threats such as eavesdropping, data modification (i.e., alteration, insertion, or destruction), or relay attacks [9]–[11]. NFC is emerging in a wide range of applications, from ticketing, staff identification, or physical access control, to cashless payment, to name a few. Following this trend, to date, almost 300 different NFC-enabled phones are (or will be soon) available at the market [12]. Hence, the eruption of NFC-enabled phones (or devices) may bring criminals a new attack vector to these NFC-enabled ID cards, as DNI or e-passports.

In this paper, we performed an independent security assessment of the NFC-enabled DNI. In particular, we evaluated the possibility of stealing personal information from a Spanish citizen without his/her knowledge using NFC capabilities. Our experiments showed that, in general, the protocols used to communicate via contactless with a DNI are secure enough and well coded. However, we discovered that the DNI did not incorporate any mechanism to prevent (on-line) brute-force attacks. We also proposed a defense mechanism.

Let us remark that we are not providing here new vulnerabilities of or attacks to contactless protocols, as in [13]–[17]. This paper presents the results of an independent security assessment of the implementation of the NFC-enabled DNI as a case study. Our goal is to verify whether such implementation is performing as expected, in terms of security, since highly sensitive data depend on it. While some functionalities of the new card have been certified, the protocols we test have not, to our knowledge, undergone a formal certification for the concrete hardware and software combination in the Spanish DNI3.0 card. Some other functionalities related to digital signatures have passed an EAL4+ (AVA_VAN.5) level Common Criteria security evaluation [18] and a previous implementation of the PACE protocol which uses the same chip the NFC DNI card has also passed an EAL4+ evaluation, but for the software in German electronic travel documents [19]. These certifications suggest random number generation and the access protocols are correctly implemented, but the NFC-enabled DNI as a whole has not been yet certified. In this case study we cover some of the untested parts and give an independent evaluation of the session establishment protocols in the NFC implementation of the DNI.

II. Security Evaluation

The DNIe3.0 communicates through the NFC interface using the Basic Access Control (BAC) and the Password Authenticated Connection Establishment (PACE) protocols.

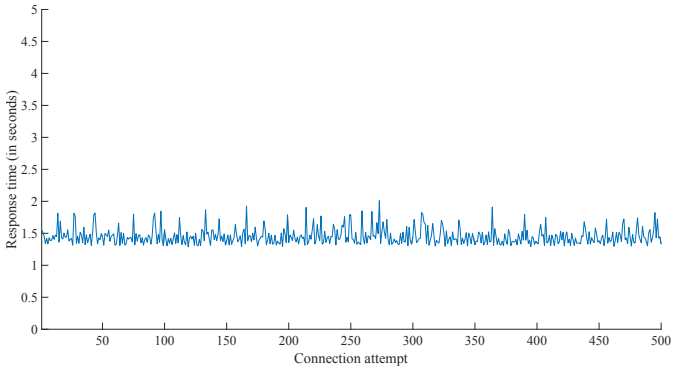


Fig. 1. Time spent in each PACE protocol connection attempt.

BAC was designed to protect less sensitive data and, in particular, to defend against skimming and eavesdropping threats [16]. The PACE protocol was proposed as an alternative to BAC, offering excellent protection against offline attacks [20]. It uses a weak password (with low entropy), verifies it, and generates cryptographically strong session keys after exchanging keys using the Diffie-Hellman protocol. As a password-derived key, it uses a 6-digit length number termed as Card Access Number (CAN). In the case of DNIe3.0, it uses a static CAN number printed on the front side of the card. CAN numbers are in $[0, 10^6)$ and hence, the entropy of this key is almost 20 bits, $\log_2 10^6 = 19.9316$. Since the key entropy used by PACE is much lower than the one used by BAC (roughly 61 vs. 20 bits, respectively), we focus on the evaluation of PACE.

We measured how long it took to perform 500 PACE protocol connection attempts. Figure 1 plots the time spent in each attempt. Our findings show that every PACE protocol connection took, on average, 1.4509 seconds, regardless of the key used. This time was (roughly) divided as follows: 200 ms to generate and operate with random numbers, 1200 ms to perform the Diffie-Hellman protocol, and 100 ms to generate, exchange, and check the authentication tokens.

These results evidence that there exists no defense implemented against on-line brute-force attacks. Regardless of the number of connection attempts, the execution time of the PACE protocol remains the same. Hence, supposing a compromised NFC-capable Android smartphone and assuming a DNIe3.0 continuously in NFC range, in the worst case personal data could be stolen in approximately 17 days.

We suggested two countermeasures to defend against on-line brute-force attacks in PACE protocol [21]. First, we propose to introduce a software method that detects consecutive connection attempts and when detected, delays the connection with the smartcard. Second, we propose to reduce the signal power from the card to the reader when an attack is detected. Both modifications make any brute force attack even less likely and almost infeasible. We communicated our findings to the National Coinage and Stamp Factory – Royal Mint, the Spanish National agency in charge of the development of the electronic Spanish ID card. They acknowledged our suggestions and told us to be taken into consideration for future DNIe3.0 implementations.

Regarding randomness of nonces generated by the card, they seem to be free from obvious correlations. The collected sequences were submitted to different randomness test, including an entropy assessment, the FIPS140-2 battery, and a delayed coordinates test.

The full version of this paper was published in [21].

Acknowledgments

This work was supported in part by MINECO TIN2014-58457-R, by CUD-2017-14, by TEC2015-69665-R, and by Junta de Castilla y León VA089U16.

References

[1] M. Jakobsson and S. Myers, *Phishing and Countermeasures: Understanding the Increasing Problem of Electronic Identity Theft*. Wiley, 2006.

[2] *Spanish Penal Code (Organic Law No. 10/1995 of Nov 23, 1995)*, Nov. 1995, available at http://www.wipo.int/wipolex/en/details.jsp?id=15759.

[3] W. Wang, Y. Yuan, and N. Archer, "A contextual framework for combating identity theft," *IEEE Security & Privacy*, vol. 4, no. 2, pp. 30–38, Mar. 2006.

[4] A. Freire, "El delito de robo de identidad (The crime of identity theft)," Online, Oct. 2015, in Spanish. Available at http://www.infoderechopenal.es/2015/10/delito-robo-identidad.html.

[5] M. Wieting, "Cuidado con perder el DNI," Online, April 2012, in Spanish. Available at http://www.abc.es/20120420/espana/abci-suplantacion-identidades-201204191917.html.

[6] G. Avoine, A. Beaujeant, J. Hernandez-Castro, L. Demay, and P. Teuwen, "A Survey of Security and Privacy Issues in ePassport Protocols," *ACM Comput. Surv.*, vol. 48, no. 3, pp. 1–37, Feb. 2016.

[7] International Organization for Standardization, "ISO/IEC 14443-3: Identification cards – Contactless integrated circuit(s) cards – Proximity cards – Part 3: Initialization and anticollision," Geneva, Switzerland, April 2011.

[8] Japanese Industrial Standard, "JIS X 6319-4:2010: Specification of implementation for integrated circuit(s) cards – Part 4: High speed proximity cards," October 2010.

[9] E. Haselsteiner and K. Breitfuß, "Security in Near Field Communication (NFC) – Strengths and Weaknesses," in *Proceedings of the Workshop on RFID Security and Privacy (RFIDSec)*, 2006.

[10] G. Madlmayr, J. Langer, C. Kantner, and J. Scharinger, "NFC Devices: Security and Privacy," in *Procs. 3rd International Conference on Availability, Reliability and Security*, March 2008, pp. 642–647.

[11] J. Vila and R. J. Rodríguez, "Practical Experiences on NFC Relay Attacks with Android: Virtual Pickpocketing Revisited," in *Proceedings of the 11th International Workshop on RFID Security*, ser. Lecture Notes in Computer Science, vol. 9440. Springer, 2015, pp. 87–103.

[12] NFC World, "NFC phones: The definitive list," January 2017.

[13] S. Vaudenay, "E-Passport Threats," *IEEE Security & Privacy*, vol. 5, no. 6, pp. 61–64, 2007.

[14] J.-H. Hoepman, E. Hubbers, B. Jacobs, M. Oostdijk, and R. W. Schreur, "Crossing Borders: Security and Privacy Issues of the European e-Passport," in *Procs. of the 1st International Workshop on Security (IWSEC)*. Springer Berlin Heidelberg, 2006, pp. 152–167.

[15] A. B. Jeng and L.-Y. Chen, "How to enhance the security of e-Passport," in *2009 International Conference on Machine Learning and Cybernetics*, vol. 5, Jul. 2009, pp. 2922–2926.

[16] J. Bender and D. Kügler, "Introducing the PACE solution," *Keesing Journal of Documents & Identity*, vol. 30, pp. 26–29, 2009.

[17] H. Richter, W. Mostowski, and E. Poll, "Fingerprinting Passports," in *NLUUG Spring Conference on Security*, 2008.

[18] Centro Criptológico Nacional (Spanish National Cryptologic Centre), "Documento 2014-39-INF-1766 v2. Informe de Certificación del producto DNIe-DSCF (dispositivo seguro de creación de firma) versión 3.0." 2017, in Spanish. Available at https://www.commoncriteriaportal.org/files/epfiles/2014-39_inf-1766_v2.pdf.

[19] Atos IT Solutions and Services GmbH, "Certification report BSI-DSZ-CC-0967-2016 for CardOS DI V5.3 EAC/PACE Version 1.0 of the BSI," 2016, https://www.commoncriteriaportal.org/files/epfiles/0967a_pdf.pdf.

[20] D. Carluccio, K. Lemke-Rust, C. Paar, and A.-R. Sadeghi, "E-Passport: The Global Traceability Or How to Feel Like a UPS Package," in *Procs. 7th International Workshop on Information Security Applications (WISA 2006). Revised Selected Papers*. Springer Berlin Heidelberg, 2007, pp. 391–404.

[21] R. J. Rodríguez and J. C. Garcia-Escartin, "Security Assessment of the Spanish Contactless Identity Card," *IET Information Security*, vol. 11, no. 6, pp. 386–393(7), Nov. 2017.

Medidas de Aplicación de la Directiva NIS a Proveedores de Servicios Digitales: Alcance y Limitaciones

M. Robles Carrillo, P. García Teodoro
Universidad de Granada - *Network Engineering & Security Group* (NESG)
mrobles@ugr.es, pgteodor@ugr.es

Abstract- **Desde su adopción el 6 de julio de 2016, la Directiva (UE) 2016/1148 relativa a las medidas destinadas a garantizar un elevado nivel común de seguridad de las redes y de los sistemas de información en la Unión ha suscitado un amplio debate, acrecentado cuando se ha aproximado la fecha límite para su transposición por parte de los Estados miembros prevista para el 9 de mayo de 2018. Paralelamente a ese proceso, la Comisión de la Unión Europea ha adoptado el Reglamento de Ejecución (UE) 2018/151 por el que se establecen las normas de aplicación de dicha Directiva respecto de la especificación de los elementos que han de tenerse en cuenta para la gestión de los riesgos de seguridad, así como los parámetros para la calificación de un incidente como significativo en el caso de los proveedores de servicios digitales. El análisis crítico de los elementos de seguridad y de los parámetros de impacto significativo sirve no solo para su comprensión, sino también para poner de manifiesto algunas carencias del régimen normativo establecido que justifican las propuestas de reforma que, a esos efectos, se incluyen en el apartado de conclusiones con vistas a su posible superación en próximas revisiones de la norma y, en particular, de los criterios aplicados en estos supuestos.**

***Index Terms*- Directiva NIS, Reglamento de Ejecución, seguridad de redes y sistemas, proveedores de servicios digitales**

Tipo de contribución: *Investigación en desarrollo*

I. Introducción

Mayo de 2018 constituye una fecha destacada en la agenda de la UE y de sus Estados miembros. Junto con la esperada aplicación de la normativa sobre protección de datos, cumple el plazo fijado a los Estados miembros para la transposición de la Directiva (UE) 2016/1148 del Parlamento Europeo y del Consejo, de 6 de julio de 2016, relativa a las medidas destinadas a garantizar un elevado nivel común de seguridad de redes y sistemas de información de la Unión (en adelante, Directiva NIS) [1].

La Directiva NIS es considerada la norma fundamental de ciberseguridad dentro de la UE, diseñada con el objetivo de lograr un elevado nivel de seguridad de redes y sistemas de información con la finalidad de mejorar el funcionamiento del mercado interior. Con esa intención, la Directiva NIS contiene medidas tanto de carácter material como de naturaleza orgánica. Entre las primeras, destacan tres: 1) la obligación de los Estados de adoptar una estrategia nacional de seguridad de redes y sistemas de información, que puede ser singularizada como tal o incluirse dentro de cada estrategia nacional general respetando, en cualquier caso, los procedimientos y contenidos determinados en el art. 7 de la Directiva NIS; 2) el establecimiento de requisitos en materia de seguridad y notificación de incidentes para operadores de servicios esenciales (OSE) y proveedores de servicios digitales (PSD); y 3) la obligación de los Estados de designar autoridades competentes, puntos de contacto único y CSIRT con atribuciones en materia de seguridad.

Junto con ello, en el plano orgánico, la Directiva NIS dispone la creación de un Grupo de Cooperación, para apoyar y facilitar la cooperación estratégica y el intercambio de información entre los Estados, y de una red de CSIRT, para contribuir al desarrollo de la confianza y la seguridad y promover una cooperación operativa rápida y eficaz.

Al tratarse de una directiva, los Estados miembros de la UE, que son sus destinatarios, tienen la obligación de proceder a su transposición en derecho interno a más tardar el 9 de mayo de 2018. Pero, desde su aprobación, la Comisión europea ha continuado adoptando las medidas normativas necesarias para dotarla de efectividad. Además de la Decisión de ejecución de 1 de febrero de 2016, por la que se crea el Grupo de Cooperación previsto en el artículo 11 de la Directiva [2], el 30 de enero de 2018 se aprueba el Reglamento de Ejecución (UE) 2018/151 por el que se establecen normas de aplicación de la Directiva NIS a nivel comunitario específicamente para los PSD [3].

En efecto, a lo largo de su articulado, la Directiva NIS establece un régimen jurídico diferenciado para las dos categorías de sujetos: los OSE y los PSD. Los primeros son definidos en el art. 4.4) como la entidad pública o privada incluida en el Anexo II que cumple los requisitos establecidos en el art. 5.2, donde se establecen los criterios y el procedimiento para su identificación por parte de los Estados miembros. El Capítulo IV de la Directiva se ocupa de los requisitos de seguridad en este caso, mientras que es el Capítulo V el que determina el régimen de los PSD, que se definen en sentido amplio, según el art. 4.6) de la Directiva NIS, como "toda persona jurídica que preste un servicio digital".

El fundamento de esta distinción radica en la convicción reconocida en la Directiva de que el grado de riesgo en el caso de los OSE es superior al existente respecto de los PSD, razón por la cual en este caso los requisitos de seguridad y notificación son menos rigurosos y su margen de actuación más amplio, aunque se requiera igualmente una armonización a escala de la UE por la naturaleza transfronteriza de su actividad. La idea es que deben ser tratados de manera uniforme y proporcionada en relación con su naturaleza y con

el grado de riesgo al que puedan enfrentarse. Una supervisión ligera, reactiva y *a posteriori* se justifica por la naturaleza de sus servicios y operaciones.

Dentro del Capítulo V de la Directiva, el artículo 16.8 dispone que la Comisión adoptará los actos de ejecución necesarios para especificar los requisitos relativos a los elementos de seguridad y a los parámetros para la determinación del alcance significativo de un incidente. El procedimiento seguido a esos efectos ha incluido la intervención del Comité de Seguridad de las Redes y Sistemas de Información en el marco de lo previsto en el Reglamento (UE) 182/2011[4]. No ha debido ser sencillo teniendo en cuenta que se ha adoptado varios meses después del plazo fijado para ello, por la propia Directiva, que era el 9 de agosto de 2017.

El objetivo de este trabajo es el análisis crítico del marco normativo formulado en el Reglamento 2018/151 adoptado en ejecución de la Directiva NIS para los PSD, identificando los elementos de seguridad, los parámetros de calificación y el complemento de impacto significativo en la calificación de un incidente de seguridad. El trabajo termina con unas conclusiones que plantean propuestas de revisión y mejora de esta normativa.

II. MARCO NORMATIVO

El Capítulo V de la Directiva NIS establece el marco normativo general de seguridad de las redes y sistemas de información de los PSD recogidos en el Anexo III que son: Mercados en línea, Motores de búsqueda en línea y Servicios de computación en la nube. Están expresamente excluidos del ámbito de aplicación subjetivo de estas disposiciones las microempresas y pequeñas empresas en los términos definidos en la Recomendación 2003/361/CE de la Comisión [5].

El Capítulo V comprende tres disposiciones dedicadas a los requisitos en materia de seguridad y notificación de incidentes (art. 16), la aplicación y observancia (art. 17) y la jurisdicción y territorialidad (art. 18). Solo los dos primeros aspectos son objeto de regulación en el Capítulo IV relativo a los OSE.

La formulación normativa de este conjunto de disposiciones y, en especial, el art. 16, no permite apreciar claramente el alcance de los compromisos establecidos respecto de cada uno de los sujetos, en mayor medida, si cabe, porque se encuentran definidos dentro de la norma dedicada a los requisitos en materia de seguridad y notificación de incidentes. En el caso de los Estados, se trata de "velar" para que los PSD adopten las medidas contempladas en esa disposición. No es precisamente un obligación de amplio alcance, ni cuyo cumplimiento se encuentre dotado de suficientes garantías. En cuanto a los PSD, se trata de cumplir determinados "requisitos". Es cierto que ha de ser la norma de transposición la que se ocupe de establecer las obligaciones a cargo de los sujetos de derecho interno, pero también es verdad que una formulación más precisa de la obligación de los Estados habría contribuido a un reforzamiento del compromiso en cuanto a los particulares.

Los tres requisitos de seguridad establecidos en el art. 16 son:

1) La determinación y adopción de medidas técnicas y organizativas de *gestión* adecuadas y proporcionadas en función de los riesgos;

2) La adopción de medidas de *prevención y de reducción del impacto* de los incidentes; y

3) La *notificación* sin dilación indebida de los incidentes a la autoridad competente o al CSIRT. Esta obligación se precisa en un doble sentido porque ha de incluir toda la información necesaria para valorar el impacto del incidente y porque se aplica solo en la medida en que disponga de acceso a esa información en función de los parámetros indicados en el apartado 4 del art. 16 [6].

Siguiendo lo previsto en el apartado 10, y sin perjuicio de lo dispuesto en el 6, en relación con la preservación de la seguridad y los intereses comerciales de los proveedores y de la confidencialidad de la información, los Estados no impondrán nuevos requisitos de seguridad o de notificación a los PSD.

Desde la perspectiva de su aplicación y observancia, sin entrar en los mecanismos generales que ofrece el Derecho de la UE, la garantía de la efectividad de estas disposiciones se concreta en el art. 17 de la Directiva con tres medidas de distinto alcance y naturaleza:

1) La adopción de medidas de supervisión *a posteriori* por parte de las autoridades competentes en caso de incumplimiento de los "requisitos" –y quizás habría que haber añadido, "obligaciones"- del art. 16 de la Directiva;

2) La dotación a las autoridades competentes de las atribuciones y medios necesarios para exigir a los PSD el suministro de la información requerida y la subsanación de cualquier incumplimiento; y

3) La cooperación y la asistencia mutua, incluido el intercambio de información, entre autoridades competentes cuando la situación tenga un carácter transfronterizo por la ubicación del proveedor o de sus redes y servicios.

Sobre la base de este marco normativo general, cuyo desarrollo reglamentario a estos efectos está previsto en el art. 16.9, el objetivo del Reglamento de Ejecución 2018/151 de la Comisión es doble: por una parte, determinar los elementos que se han de tener en cuenta para garantizar un nivel suficiente y adecuado de seguridad en las redes y sistemas en el ámbito de los servicios incluidos en ese Anexo III de la Directiva NIS; y, por otra parte, fijar los parámetros de calificación de un incidente con impacto significativo en la prestación de dichos servicios.

Subjetivamente, los destinatarios de la norma son los PSD. Su definición en el art. 4.6 de la Directiva NIS como "toda persona jurídica que preste un servicio digital", se completa mediante una remisión expresa al art. 1.1.b) de la Directiva (UE) 2015/1535 de conformidad con el cual incluye "todo servicio de la sociedad de la información, es decir, todo servicio prestado normalmente a cambio de una remuneración, a distancia, por vía electrónica y a petición individual de un destinatario de servicios" [7].

Materialmente, el contenido esencial de la norma concierne a la especificación de los elementos que se han de tener en cuenta para gestionar los riesgos existentes para la seguridad de las redes y los sistemas de información. La especificación se define en la Directiva NIS como una especificación técnica en el sentido del art. 2.4 del Reglamento (UE) 1025/2012, en virtud del cual se trata de "un documento en el que se prescriben los requisitos técnicos que debe reunir un producto, proceso, servicio o sistema" y

que establece uno o más de los aspectos siguientes: características del producto o del servicio, métodos y/ o procedimientos de producción, o métodos y criterios para evaluar el rendimiento [8].

Jurídicamente, el problema principal estriba en determinar el alcance de las obligaciones y requisitos. En su articulado, la Directiva NIS utiliza la fórmula "los Estados velarán" por que los PSD tomen las medidas o notifiquen los incidentes, según el caso. En su título, el Reglamento de Ejecución 2018/151 hace referencia a las normas "que han de tener en cuenta" los PSD. En su primer considerando, se refiere a que los PSD "pueden tomar las medidas". Esta apreciable falta de sintonía en la determinación del alcance y contenido de la norma puede conducir a una regulación dispar en función del alcance que cada Estado quiera otorgar a la función de "velar" en la adopción de su normativa interna y a un eventual desajuste entre esa normativa y el contenido del mismo del Reglamento de Ejecución 2018/151 si aquella va más lejos, en la definición de su alcance, de lo que se interpreta en este marco reglamentario. La primacía de la normativa europea podría frenar esa posibilidad, a pesar de ser un avance deseable en términos de garantía de la seguridad de redes y sistemas.

Con carácter general, en la totalidad de esta normativa, habría que plantearse si el tratamiento conjunto de los requisitos de seguridad –que son procedimentales en su condición de medios, pero finalistas desde el punto de vista de la seguridad material que se pretende alcanzar- y los de notificación –esencialmente, procedimentales- puede incidir negativamente en el significado y el valor intrínseco de los primeros que son el componente básico de esta normativa. Dicho en otros términos, quizás habría sido conveniente distinguir con mayor relevancia jurídica la necesidad de homogeneizar los requisitos materiales de seguridad de redes y sistemas respecto de los procedimientos de notificación de los incidentes de seguridad. Los elementos de seguridad son razonablemente el elemento principal de este cuerpo normativo.

III. ELEMENTOS DE SEGURIDAD

Dentro de los requisitos de seguridad, el art. 16 de la Directiva NIS contempla la seguridad de los sistemas e instalaciones; la gestión de incidentes; la gestión de la continuidad de las actividades; la supervisión, auditorías y pruebas; y el cumplimiento de las normas internacionales. El art. 2 del Reglamento de Ejecución 2018/151 desarrolla esos elementos de seguridad precisando su alcance y contenido. Los PSD garantizarán la disponibilidad de la documentación necesaria para acreditar el cumplimiento de dichos elementos de seguridad (art. 2.6).

La *seguridad de los sistemas e instalaciones* comprende la gestión sistemática de redes y sistemas de información, la seguridad física y del entorno, la seguridad de abastecimiento y el control de acceso a redes y sistemas de información.

Las medidas que han de adoptar los PSD en materia de *gestión de incidentes* cubren una trayectoria completa desde el establecimiento de procedimientos y procesos de detección de anomalías y vulnerabilidades, la notificación de incidentes, los mecanismos de respuesta y comunicación hasta, finalmente, la evaluación de la gravedad de los incidentes.

La *gestión de la continuidad* de las actividades se define en el art. 2.4 del Reglamento de Ejecución 2018/151 como "la capacidad de una organización de mantener o, en su caso, restablecer, después de un incidente perturbador, la prestación de servicios a niveles aceptables preestablecidos". Ello incluye el establecimiento y utilización de planes de contingencia basados en análisis de impacto y la disposición de capacidades de recuperación en caso de catástrofe.

El apartado de *supervisión, auditoría y pruebas* se concreta en las políticas de supervisión y registro, la planificación de contingencias durante los ejercicios, los ensayos con las redes y sistemas de información y las evaluaciones de seguridad en términos de funcionalidad, eficiencia y eficacia.

El *cumplimiento de las normas internacionales* es un requisito de seguridad previsto en la Directiva NIS que, siguiendo el art. 2.5 del Reglamento de Ejecución 2018/151, se refiere a dos categorías de normas: por una parte, las adoptadas por un organismo internacional de normalización en los términos definidos en el art. 2.1.a) del Reglamento (UE) 1025/2012; o las normas y especificaciones aprobadas a nivel europeo, internacional o, incluso, nacional, de acuerdo con lo previsto en el art. 19 de la Directiva (UE) 2016/1148. La variedad de opciones es apreciable [9], aunque la inclusión de la certificación nacional puede, no obstante, ser un factor de distorsión. Posiblemente, los avances en materia de certificación europea contemplados en la propuesta de Reglamento presentada por la Comisión para la reforma de ENISA y de la certificación de ciberseguridad puedan contribuir a una mejor regulación de este aspecto [10].

IV. PARÁMETROS DE CALIFICACIÓN

El artículo 16.4 de la Directiva NIS establece los siguientes parámetros de seguridad en relación con los PSD para determinar si un incidente es significativo: el número de usuarios afectados, la duración, la extensión geográfica, el grado de perturbación del funcionamiento del servicio y el alcance del impacto sobre las actividades económicas y sociales.

Solo los tres primeros supuestos se reconocen en el art. 14.4 de dicha Directiva como parámetros a esos efectos en el caso de OSE. Esta asimetría no parece tener una especial justificación. El grado de perturbación en el funcionamiento del servicio o el alcance de su impacto son susceptibles de baremación en el caso de los PSD, junto con los demás, como criterios no exclusivos, porque no se trata de una enumeración cerrada, pero no se identifican como tales respecto de los OSE, a pesar de que tampoco se concibe como una lista exhaustiva.

Contribuyendo a ese estatuto asimétrico, la obligación de notificación que tienen los operadores únicamente se aplica a los PSD cuando tengan acceso a la información necesaria para valorar el impacto del incidente atendiendo a los extremos indicados como elementos de seguridad en el art. 14.1. En cualquier caso, y redundando en el alcance de la obligación de notificación, en ninguno de los supuestos se exigirá a los PSD que recopilen información adicional a la que no tengan acceso (art. 3.6), circunstancia que limita considerablemente el alcance de dicha norma. Quizás habría sido aconsejable mitigar el valor de refugio, o de exclusión de responsabilidad a esos efectos, incluyendo alguna forma de exigencia de una diligencia debida o mínima en relación con la obtención de esa información [11].

El art. 3 del Reglamento 2018/151 concreta estos parámetros previstos en la Directiva NIS en los siguientes términos.

La determinación del *número de usuarios* afectados por el incidente, siempre que el PSD se encuentre en condiciones de proceder a su estimación, se apreciará atendiendo bien al número de personas físicas y jurídicas afectadas con las que se haya celebrado un contrato de prestación de servicios o bien el número de usuarios afectados que hayan utilizado el servicio basándose, en particular, en el tráfico de datos previo (art. 3.1). La primera opción ofrecería un resultado potencialmente más objetivo, pero no necesariamente coincidente en mayor extremo con la realidad, mientras que la segunda mostraría un balance posiblemente más real, pero también susceptible en mayor medida de interpretación según los criterios, además del tráfico de datos previo, que se aplicasen o del valor que se otorgase a ese concepto en particular.

El criterio de la *duración* se define en el art. 3.2 pero se cuantifica en el art. 4.1.a). Comprende el plazo transcurrido desde que se produce una perturbación que afecta a la disponibilidad, autenticidad, integridad o confidencialidad hasta el restablecimiento del servicio. Considerado en términos de horas de usuario, el impacto significativo se produce cuando la duración supera los 5 millones de horas. No queda claro, sin embargo, si esa cuantificación supone sumar el número de horas durante las cuales se ha producido la afectación de cada una de aquellas funcionalidades por separado o en su conjunto. No puede merecer la misma calificación un incidente que limita solo la disponibilidad del sistema a otro que afecta a la disponibilidad, integridad y confidencialidad, por ejemplo, durante el mismo margen de tiempo.

La *extensión geográfica* es un parámetro que se circunscribe en el art. 3.3 a la capacidad del PSD de determinar si el incidente afecta a Estados miembros concretos, sin mayor precisión en cuanto al número de ellos o al alcance territorial de la incidencia. También en este caso hubiese sido deseable una mayor precisión en términos de seguridad jurídica referidos a la apreciación de las dimensiones del incidente en cuestión en términos globales, locales o intermedios dentro de la UE.

El *grado de perturbación del funcionamiento* del servicio es un indicador que se medirá, según el art. 3.4, en relación "con una o varias" de las siguientes características afectadas por el incidente: disponibilidad, autenticidad, integridad o confidencialidad de los datos o servicios. Esta referencia a una o a varias, que habría sido útil en la apreciación del criterio de la duración, podría haberse precisado, incluso, simplemente ordenando de mayor a menor gravedad el tipo de afectación porque no es igual, sin ir más lejos, un atentado a la integridad que uno a la disponibilidad de los datos y servicios.

El *alcance del impacto sobre las actividades económicas y sociales* se cifrará considerando si el incidente ha causado pérdidas significativas, materiales o inmateriales, atendiendo a los usuarios que el PSD pueda determinar basándose en el número de usuarios potencialmente afectados o en indicaciones como el carácter de sus relaciones contractuales con el cliente, entre otras posibilidades.

El artículo 4, al regular el impacto significativo, precisa algunos de esos parámetros, pero no todos porque no hay una correspondencia exacta o, cuando menos, clara entre dichos parámetros y el desencadenante cuantitativo o finalista de su ascenso a la condición de impacto significativo.

OBJETO	BIEN JURÍDICO	CUANTIFICACIÓN
Servicio	Disponibilidad	5.000.000 horas
Incidente	Autenticidad Integridad Confidencialidad	100.000 usuarios
Incidente	Seguridad pública Pérdida de vidas humanas	Sin cuantificar
Incidente	Perjuicio material	1.000.000 euros

Tabla 1. Cuantificación de 'impacto' significativo en NIS para PSD.

V. EL COMPONENTE "IMPACTO SIGNIFICATIVO"

La determinación de los criterios de calificación de un incidente dentro de la categoría de "impacto significativo" constituye, posiblemente, una aportación destacada y original del Reglamento de Ejecución 2018/151. Frente a ello, el Esquema Nacional de Seguridad [12] define el nivel de seguridad y la categoría de un sistema como BAJO/A-MEDIO/A-ALTO/A exclusivamente en términos cualitativos, lo que introduce un notable grado de inseguridad y de incertidumbre que poco o nada contribuye al objetivo de la securización.

El art. 4 del Reglamento de Ejecución 2018/151 asigna valores cuantificables, atendiendo a los distintos parámetros y bienes jurídicos afectados en cada caso, de acuerdo con la Tabla 1.

El análisis del modelo diseñado en esta tabla permite identificar tres modalidades de valoración: 1) La cuantificación numérica del grado de afectación del servicio; 2) La combinación de una cuantificación numérica y un resultado; y 3) La existencia de un riesgo de resultado.

1) La cuantificación numérica como criterio de calificación está prevista en los apartados a) y d) del art. 4.1, esto es: cuando el servicio prestado por el PSD no está disponible en número de horas (5.000.000) contabilizadas en términos de número de usuarios afectados y duración de la contingencia; y cuando el incidente causa a un usuario "en la Unión" un daño material superior a un millón de euros. Esta referencia se analiza más adelante.

2) La combinación de una cuantificación numérica y un resultado se da en el supuesto b) del art. 4.1, al exigir que se haya afectado "a más de 100.000 usuarios en la Unión" junto con el requisito material de la pérdida de autenticidad, integridad o confidencialidad de los datos o de los servicios ofrecidos o accesibles mediante una red o sistema de información. Además de reproducir los problemas derivados de la referencia "en la Unión", a los que se hará referencia a continuación, no se menciona la "disponibilidad" de los datos o los servicios, a pesar de ser una categoría tan importante como las demás, tan necesitada de protección como ellas y tan vulnerable a fenómenos delictivos de crecimiento exponencial como el *ransomware*. Es cierto que el apartado a) del artículo 4.1 contempla la "indisponibilidad", pero solo

referida al servicio prestado por el PSD y no a los datos almacenados, transmitidos o tratados.

3) La existencia de un riesgo para la seguridad pública o de pérdida de vidas humanas, sin componente numérico al efecto, constituye una justificación completamente lógica para calificar un incidente de impacto significativo. Plantea, no obstante, el problema de determinar el alcance del concepto de riesgo o, dicho en otros términos, quién y cómo se determina la presencia del mismo más allá del caso evidente de que se haya producido una pérdida de vidas humanas o una situación perfectamente objetivable de afectación de la seguridad pública. La afectación de la seguridad pública es difícilmente evaluable en términos objetivos para cualquier supuesto desde el momento en que el concepto mismo de "seguridad" es contingente y contextual [13]. También es cuestionable la mención limitada al concepto de seguridad pública y la ausencia de referencia a la seguridad nacional [14].

Con carácter general, en estas disposiciones, el recurso a la fórmula un "usuario en la Unión" no está exento de problemas. Como primera providencia se utiliza en tres de los cuatro supuestos y no se emplea precisamente en el que resultaría más acertado, por ser más gravoso y por sus propios efectos, que es la creación de un riesgo para la seguridad pública o la pérdida de vidas humanas. En ambos casos parece más lógico que la contingencia haya de producirse en la Unión Europea pero no se establece esa precisión presente, sin embargo, en los demás.

Esta referencia al "usuario de la Unión" es criticable porque, generalmente, se recurre a criterios más precisos basados en la competencia territorial o personal que atienden bien a la localización del usuario dentro del territorio de la UE, como nacional de un Estado miembro o como residente, o bien a su condición de nacional de un Estado y, en consecuencia, ciudadano de la UE. De hecho, al determinar el régimen de jurisdicción, el art. 18 de la Directiva NIS aplica el criterio de la territorialidad, expresado como el lugar donde se encuentra el establecimiento principal del proveedor, y establece una distinción entre, por una parte, PSD que tienen su establecimiento principal en el Estado miembro donde tienen fijado su domicilio social y a cuya jurisdicción están sometidos y, por otra parte, los PSD que carecen de establecimiento y que habrán de designar un representante en alguno de los Estados que será determinante de su jurisdicción. Esta cláusula que fija claramente la competencia sobre una base de territorialidad no se ha reproducido, cuando se hubiese podido hacer, para fijar territorialmente el impacto del incidente que merece la calificación de significativo.

La expresión "usuario en la Unión" genera algunos interrogantes adicionales. En primer lugar, conviene recordar la problemática general que plantea la trazabilidad desde el punto de vista técnico y la atribución desde una perspectiva jurídica para determinar la localización del usuario "en la Unión", cuando podría haberse definido considerando la incidencia respecto de un usuario domiciliado en la Unión o, también, incluso, en su caso, de un nacional de alguno de sus Estados miembros.

En efecto, en segundo término, la fórmula utilizada puede suponer la no calificación como incidente de impacto significativo en el caso de ciudadanos de la UE que no se encuentren físicamente en la Unión y la calificación positiva en caso de nacionales de terceros países ajenos a la UE por el hecho de encontrarse en la UE en el momento del incidente. Esta situación es doblemente anómala porque marca sin justificación una diferencia de trato entre los ciudadanos de la UE que se encuentran dentro y fuera de ella y porque protege al extranjero que se encuentre en la Unión, mientras desatiende al propio ciudadano por no encontrarse en ella. Es cierto que no siempre resulta fácil conciliar las competencias a título territorial y a título personal y resolver los posibles conflictos positivos o negativos que se puedan plantear, pero también es verdad que no hay una justificación evidente para excluir la afectación producida en el exterior a un ciudadano de la Unión a efectos de calificación del incidente. Podría entenderse la voluntad de circunscribir los casos al ámbito territorial de actuación de la UE, pero habrían de asumirse las limitaciones que implica ese criterio en el contexto singular del ciberespacio a efectos de delimitación territorial [15].

En tercer lugar, con la expresión "en la Unión", la cuantificación del daño se complica innecesariamente porque podría ocurrir que se produjese un daño global superior pero menor al requerido "en la UE" para considerarlo un incidente de impacto significativo. Podría darse la circunstancia de que, considerando los usuarios vinculados a la UE, por ciudadanía o territorio, sí se alcanzase o superase ese límite sin que el incidente pudiese ser considerado de impacto significativo. Para terminar, podría entenderse que se produce esa situación cuando los servidores se encuentran en la Unión, aunque no lo esté el usuario o el proveedor o los propios datos. Si los datos afectados están en la nube, cabría entender que están o no en la UE cambiando con ello la posibilidad de cuantificarlos para determinar el impacto del incidente.

En su conjunto, los criterios de calificación del impacto significativo de un incidente no responden a un planteamiento global coherente cuando podría haberse utilizado una fórmula más precisa atendiendo a los criterios tradicionales de atribución de competencias –territorial y personal- y a la singularidad de la actividad ciberespacial a efectos de localización, en particular, la ausencia de delimitación física o los servicios en la nube. Junto a ello, quizás se habría requerido también la previsión de que un daño material significativo equivalente a la destrucción podría ser singularizado como una modalidad de incidente que, por sus efectos definitivos, no habría de llegar necesariamente al umbral del millón de euros para considerarse significativo.

De conformidad con lo previsto en el art. 4.2, estos criterios podrán ser revisados por la Comisión siguiendo dos indicadores: las mejores prácticas recopiladas por el Grupo de Cooperación de conformidad con el art. 11.3 de la Directiva NIS y los debates previstos en esa misma disposición. Con vistas a esa posible reforma se incluyen las propuestas realizadas en las siguientes conclusiones.

III. CONCLUSIONES

La seguridad de las redes y de los sistema de información constituye, desde hace tiempo, una prioridad dentro de la UE que ha justificado finalmente la adopción de la Directiva NIS que establece dos categorías de sujetos diferenciadas: OSE y PSD. Habiendo sido considerada mayor la importancia en términos globales de los primeros, no es significativamente mucho menor la de los segundos. Además de las relaciones de dependencia existentes entre ambos en no pocos supuestos, en el marco de la UE, atendiendo a sus competencias y funcionamiento, la figura del PSD adquiere una singular

Aspecto	Mejora	Aspecto	Mejora
Marco normativo general *Capítulo V (arts. 16 a 18) y Anexo III de la Directiva NIS*	- Mayor asimetría en el régimen jurídico de operadores y proveedores - Mayor y mejor definición del alcance jurídico de las obligaciones - Separación de los requisitos de seguridad y de notificación - Previsión de mecanismos de garantía más eficaces - Mejor interacción entre el concepto, los elementos y los parámetros de seguridad	Parámetros de Calificación *Art. 16.4 de la Directiva NIS Art. 3 Reglamento de Ejecución 2018/151*	- Mejora del modelo de determinación del número de usuarios - Reformulación del criterio de la duración en función de la gravedad y el número de bienes jurídicos afectados - Gradación del criterio territorial - Precisión de los criterios de grado de perturbación y alcance del impacto
Elementos de seguridad *Art. 16 de la Directiva NIS Art. 2 Reglamento de Ejecución 2018/151*	- Homogeneización del modelo de certificación	Impacto significativo *Art. 4 del Reglamento de Ejecución 2018/151*	- Mejor definición del concepto de usuario - Reformulación del supuesto de seguridad pública y pérdida de vidas humanas - Inclusión de la pérdida de disponibilidad de datos - Precisión del concepto de tiempo

Tabla 2. Propuestas de mejora.

relevancia, como ha puesto de manifiesto reiteradamente la propia ENISA [16].

El marco normativo general establecido en la Directiva NIS y desarrollado en el Reglamento de Ejecución 2018/151 -analizado en el apartado II- plantea varios interrogantes y problemas, destacando los siguientes:

1) La necesidad de una mayor simetría en la construcción del régimen normativo de OSE y PSD. No se trataría necesariamente de equiparar sus estatutos, sino de homogeneizarlos en aras de una mayor transparencia y una mejor comprensión del sistema.

2) Una definición más precisa del alcance y contenido de las obligaciones de los Estados, como destinatarios de la Directiva NIS, en relación con la actividad de los PSD, así como de las que deberían corresponder a estos últimos, si es que cabe otorgarles esa definición, de conformidad con el Reglamento de Ejecución 2018/151.

3) Una separación solo aparentemente formal, por sus connotaciones de fondo, entre responsabilidades y requisitos de seguridad, por una parte, y compromisos en materia de notificación, reforzando el valor de los primeros y ubicando estos últimos en el terreno claro de los procedimientos.

4) La previsión de mecanismos de garantía de la efectividad de estas disposiciones adecuados y suficientes, atendiendo a la naturaleza de las medidas, para coadyuvar a su cumplimiento y detectar lagunas y carencias. Sin entrar en más detalles, la supervisión *a posteriori* una vez detectados eventuales incumplimientos está lejos de responder a las políticas proactivas que requiere la garantía de seguridad de redes y sistemas.

5) Una mayor y mejor interacción entre el concepto y el contenido de los elementos de seguridad y de los parámetros de calificación, así como de estos últimos respecto de los indicadores de impacto positivo.

La articulación de los *elementos de seguridad* considerados en el apartado III de este trabajo, tanto en la Directiva NIS, como especialmente en el Reglamento de Ejecución 2018/151, responde a una construcción lógica, integral y global en la que se han incorporado secuencialmente los componentes requeridos a esos efectos. El modelo de certificación adolece de algunas carencias que presumiblemente se cubrirán con la normativa prevista con ocasión de la reforma de ENISA.

El estudio de los *parámetros* que han de tenerse en cuenta para determinar el impacto significativo de un incidente, desarrollado en el apartado IV, permite realizar algunas observaciones con vistas a su posible remodelación. Entre ellas destacan las siguientes:

1) La determinación del número de usuarios podría realizarse mediante una combinación de los dos elementos que ofrece alternativamente el art. 3.1.1 del Reglamento de Ejecución de manera que el componente de objetividad que aporta el primero de ellos se pudiese contrastar con el componente de realidad que ofrece el segundo sobre la base del tráfico de datos.

2) El criterio de la duración debería ser atemperado en función de los bienes jurídicos afectados y del mayor o menor número de los mismos.

3) El criterio de la extensión geográfica habría, asimismo, de modularse diferenciando un alcance global, cuando afecta

a la mayor parte de la UE, local cuando se circunscriba a un número muy reducido de Estados o de nivel intermedio.

4) El grado de perturbación del funcionamiento y el alcance del impacto son categorías formuladas con una generalidad que puede atenazar las garantías en materia de seguridad jurídica.

Junto con lo anterior, siguiendo el análisis realizado en el apartado V de este trabajo sobre los criterios para la determinación de la existencia de un *impacto significativo* respecto de esos parámetros, se plantean las siguientes propuestas de reforma (véase Tabla 2):

1) Una definición más precisa del concepto de usuario al que hacen referencia los apartados a), b) y d) del art. 4.1, atendiendo a su localización en el territorio de la UE, con residencia/domicilio social o no, pero incluyendo la posible afectación de ciudadanos de la UE que se encuentren fuera de ella.

2) Una reformulación del supuesto contemplado en el apartado c) para situar o no claramente el riesgo dentro de la UE, tanto en términos de seguridad como de pérdida de vidas humanas, y para incorporar un concepto de seguridad más preciso usando solo esa expresión de modo genérico o sumándole la seguridad nacional.

3) La inclusión de la pérdida de disponibilidad de los datos en el apartado b) del art. 4.1.

4) Una precisión del concepto tiempo en los términos indicados por ENISA [17].

En el momento de terminar estas páginas, aún se está discutiendo en sede parlamentaria la norma de transposición de la Directiva NIS al derecho español [18]. Si el Reglamento de Ejecución 2018/151 se hubiese adoptado en el plazo previsto, podría haber servido en mayor medida para la adaptación de esa normativa o podría haberse visto completado con las eventuales aportaciones de la norma española. En cualquier caso, más allá de las críticas que pueda merecer esta normativa en aspectos concretos, hay que valorar positivamente su contribución al objetivo común y compartido de garantizar la seguridad de redes y sistemas dentro del conjunto de la UE.

AGRADECIMIENTOS

Este trabajo ha sido financiado parcialmente por el Gobierno de España, con fondos FEDER, a través del proyecto TIN2017-83494-R.

REFERENCIAS

[1] *DOUE*, L 194, de 19 de Julio de 2016, p. 1.

[2] Decisión de Ejecución (UE) 2017/179 de la Comisión de 1 de febrero de 2017 por la que se establecen las disposiciones de procedimiento necesarias para el funcionamiento del Grupo de cooperación a que se refiere el artículo 11, apartado 5, de la Directiva (UE) 2016/1148 del Parlamento Europeo y del Consejo, relativa a las medidas destinadas a garantizar un elevado nivel común de seguridad de las redes y sistemas de información en la Unión (*DOUE*, L 28, de 2 de febrero de 2017, p. 73).

[3] Reglamento de Ejecución (UE) 2018/151 de la Comisión, de 30 de enero de 2018, por el que se establecen normas de aplicación de la Directiva (UE) 2016/1148 del Parlamento Europeo y del Consejo en lo que respecta a la especificación de los elementos que han de tener en cuenta los proveedores de servicios digitales para gestionar los riesgos existentes para la seguridad de las redes y sistemas de información, así como de los parámetros para determinar si un incidente tiene un impacto significativo (*DOUE*, L 26, de 31 de enero de 2018, p. 48).

[4] Reglamento (UE) 182/2011 del Parlamento Europeo y del Consejo, de 16 de febrero de 2011, por el que se establecen las normas y los principios generales relativos a las modalidades de control por parte de los Estados miembros del ejercicio de las competencias de ejecución por la Comisión (*DOUE*, L 55, de 28 de febrero de 2011, p. 13).

[5] Recomendación 2003/361/CE de la Comisión de 6 de mayo de 2003 sobre la definición de microempresas, pequeñas y medianas empresas (*DOCE*, L 124, de 20 de mayo de 2003, p. 36).

[6] Lina Jasmontaite, "Building a Cybersecurity Culture in the EU Through Mandatory Notification of Data Breaches and Incidents: Differences and Similarities of Data Vulnerability Reporting Tools", en *Managing Risk in the Digital Society*, Universitá Oberta de Catalunya, Barcelona, p. 131; Florian Menges y Günther Pernul, "A comparative analysis of incident reporting formats", *Computer & Security*, Vol. 73, 2008, p. 87.

[7] Directiva (UE) 2015/1535 del Parlamento Europeo y del Consejo, de 9 de septiembre de 2015, por la que se establece un procedimiento de información en materia de reglamentaciones técnicas y de reglas relativas a los servicios de la sociedad de la información (*DOUE*, L 241, de 17 de septiembre de 2015, p. 1).

[8] Reglamento (UE) 1025/2012 del Parlamento Europeo y del Consejo, de 25 de octubre de 2012, sobre la normalización europea, por el que se modifican las Directivas 89/686/CEE y 93/15/CEE del Consejo y las Directivas 94/9/CE, 94/25/CE, 95/16/CE, 97/23/CE, 98/34/CE, 2004/22/CE, 2007/23/CE, 2009/23/CE y 2009/105/CE del Parlamento Europeo y del Consejo y por el que se deroga la Decisión 87/95/CEE del Consejo y la Decisión no 1673/2006/CE del Parlamento Europeo y del Consejo (DOUE, L 316, de 14 de noviembre de 2012, p. 12). La Directiva NIS remite al concepto del apartado 2.4 y no del 2.5 del Reglamento que es el que recoge el concepto de especificación técnica de las TIC.

[9] Eric Lachaud, "The General Data Protection Regulation and the Rise of Certification as a Regulatory Instrument", en *Managing Risk in the Digital Society*, Universitá Oberta de Catalunya, Barcelona, p. 147.

[10] Propuesta de Reglamento del Parlamento Europeo y del Consejo relativo a ENISA, la Agencia de Ciberseguridad de la UE, y por el que se deroga el Reglamento (UE) 526/2013, y relativo a la certificación de ciberseguridad de las tecnologías de la información y la comunicación, [COM (2017) 477, final].

[11] Karine Bannelier y Théodore Christakis, "Cyber-Attacks. Prevention-Reactions: The Role of States and Private Actors", *Les Cahiers de la Revue Défense Nationale*, 2017, p. 13.

[12] https://www.boe.es/boe/dias/2010/01/29/pdfs/BOE-A-2010-1330.pdf

[13] M. Robles Carrillo, "El ciberespacio: presupuestos para su ordenación jurídico-internacional", *Revista Chilena de Derecho y Ciencia Política*, Vol. 7, Nº 1, 2016, pp. 36 y ss.

[14] Claudia Aradau y Tobias Blanke, "Governing others: Anomaly and the algorithmic subject of security", *European Journal of International Security*, Vol. 3, Nª 1, 2017, p. 4.

[15] Oren Gross, "Legal Obligations of States Directly Affected by Cyber-Incidents", *Legal Studies Research Paper Series*, Research Paper Nº 15-03, 2015, p. 14.

[16] ENISA, *Technical Guidelines for the implementation of*

minimum security measures for Digital Services Providers, Diciembre, 2016. Consulta: Febrero de 2018 (https://www.enisa.europa.eu/publications/minimum-security-measures-for-digital-service-providers).

[17] ENISA, *Incident notification for DSPs in the context of the NIS Directive*, Febrero, 2017. Consulta: Febvreo 2018 (https://www.enisa.europa.eu/publications/incident-notification-for-dsps-in-the-context-of-the-nis-directive/ Diciembre, 2016. Consulta: febrero de 2018 (https://www.enisa.europa.eu/publications/minimum-security-measures-for-digital-service-providers).

[18] http://www.minetad.gob.es/telecomunicaciones/es-ES/Participacion/Documents/anteproyecto-ley-seguridad-redes-sistemas-informacion/Anteproyecto-Ley-NIS-29nov-2017.pdf

Marisma-BiDa: Entorno Integrado de Análisis y Gestión de Riesgos en Big Data

Julio Moreno
Grupo de investigación GSyA
Universidad de Castilla-La Mancha, Ciudad Real, España
Julio.Moreno@uclm.es

Luis E. Sánchez
Dpto I+D+i
Marisma Shield S.L., Tomelloso, España
luisenrique@sanchezcrespo.org

Antonio Santos-Olmo
Dpto I+D+i
Sicaman Nuevas Tecnologías S.L., Tomelloso, España
asolmo@sicaman-nt.com

David G. Rosado
Instituto de Tecnologías y Sistemas de Información (ITSI)
Universidad de Castilla-La Mancha, Ciudad Real, España
David.GRosado@uclm.es

Manuel A. Serrano
Grupo de investigación Alarcos
Universidad de Castilla-La Mancha, Ciudad Real, España
Manuel.Serrano@uclm.es

Eduardo Fernández-Medina
Grupo de investigación GSyA
Universidad de Castilla-La Mancha, Ciudad Real, España
Eduardo.FdezMedina@uclm.es

Resumen- **Los datos son uno de los activos más importantes para todo tipo de compañías, habiendo crecido muchísimo su cantidad y las formas de explotarlos. En ese contexto, aparece Big Data, como un conjunto de tecnologías que gestionan los datos para obtener información que soporte la toma de decisiones. Estos sistemas no fueron concebidos para ser seguros, dando lugar a importantes riesgos que deben controlarse. En este artículo proponemos una técnica de análisis y gestión de riesgos para entornos Big Data, que se imbrica con una metodología de gestión de seguridad (MARISMA), soportada por un entorno tecnológico en la nube (eMARISMA), que está siendo utilizada por decenas de clientes. Esta metodología y su entorno están concebidos para poder ser fácilmente adaptados a contextos particulares como Big Data. Nuestra propuesta, denominada MARISMA-BiDa, se basa en los principales estándares, como la ISO/IEC 27.000, o la arquitectura de referencia del NIST para Big Data.**

***Index Terms*- Análisis y gestión de riesgos, Metodología, Big Data**

Tipo de contribución: *Investigación original*

I. Introducción

Los datos se han convertido en algo cada vez más importante en compañías de cualquier campo. No solo son fundamentales para organizaciones relacionadas con el campo de las tecnologías de la información, sino que también son cruciales para industrias tan variadas como la sanitaria, la educación, la ingeniería o los gobiernos. Para todos ellos, los datos son esenciales para llevar a cabo sus actividades diarias y ayudar a la alta gerencia a alcanzar sus objetivos de negocio y como consecuencia, tomar mejores decisiones basadas en la información extraída de esos datos [1, 2]. Además, cada vez generamos una mayor cantidad de datos. Se estima que el 90 por ciento de los datos generado por los humanos en toda su historia se han creado en los últimos años. Por ejemplo, en todo 2003 se generaron 5 Exabytes de datos, en 2013 esa cantidad de datos se generaba cada 2 días [3]. Esta tendencia de aumentar la cantidad de datos generadas no parece que vaya a cambiar en un futuro cercano, todo lo contrario, ya que en 2016 se generaron 16.1 Zettabytes y se espera que para 2025 se alcancen los 163 Zettabytes [4]. El incremento del uso de redes sociales, datos multimedia y el Internet de las Cosas (IoT), produce cada vez una mayor cantidad de datos [5]. Por otro lado, muchos de estos datos tienen un formato no estructurado, lo cual, junto con la rápida producción de los mismos (en muchos casos en tiempo real) complica su análisis mediante sistemas tradicionales. Este conjunto de características se conocen como las 3 Vs (Volumen, Variedad y Velocidad) de Big Data [6]. Big Data surge como respuesta a la necesidad de analizar y entender mejor estos datos, para de esta forma, obtener información valiosa para la organización.

Sin embargo, con cada nueva tecnología surgen nuevos problemas, y Big Data no es una excepción. Al usar Big Data no solo aumenta la escala de los problemas tradicionales de privacidad y seguridad sino que se añaden nuevos desafíos que deben ser afrontados [9]. Estos problemas se derivan del hecho de que Big Data no fue inicialmente concebido como un entorno seguro [10], pero en cambio, los riesgos de seguridad a los que un sistema de este tipo puede estar sometido son muy elevados. Tanto es así, que incluso los problemas de seguridad en Big Data son uno de los principales frenos para su adopción por parte de las compañías, por temores a tener pérdida de información confidencial, problemas de reputación, o incluso de incumplimiento de la legislación sobre protección de datos [11]. Por consiguiente, es muy importante contar con una serie de guías, metodologías y mecanismos para implementar de forma adecuada no solo el entorno Big Data, sino también su seguridad. Pero no solo eso, además, es ampliamente considerado que todo entorno global de gestión de seguridad de la información en la empresa, debe estar centrado en los riesgos [7, 8, 12] . Por lo que los riesgos de seguridad en Big Data, deben ser analizados y gestionados de manera adecuada, junto a los riesgos de otros tipos de activos de información.

Aunque existen numerosas propuestas para abordar el análisis y gestión de riesgos de manera sistemática, como MAGERIT [13], OCTAVE [14] o CRAMM [15], estas

propuestas frecuentemente ofrecen dificultades para su aplicación en la práctica, no cuentan con herramientas adecuadas para su procesamiento (o en caso de existir, éstas no son muy usables), están pensadas para ser aplicadas en grandes compañías, y no son sensibles al contexto, sin contar con capacidades de adaptación para entornos especiales, que requieran un especial tratamiento de sus riesgos. Con el objetivo de mejorar esas carencias, se desarrolló la metodología denominada MARISMA (Methodology for the Analysis of Risks on Information System, using Meta-Pattern and Adaptability), así como un entorno tecnológico llamado eMARISMA que la soporta (www.emarisma.com). MARISMA es una metodología basada en la reutilización de conocimiento para el proceso de análisis y gestión de riesgos, mediante el uso de unas estructuras que denominamos patrones, que permiten soportar diferentes tipos de casos, al tiempo que ayuda a reducir drásticamente el esfuerzo del proceso El entorno tecnológico eMARISMA es una plataforma en la nube que implementa los procesos de MARISMA, permitiendo automatizar el proceso de análisis y gestión de riesgos, dando soporte a la reutilización, y permitiendo actualización en tiempo real de los indicadores de riesgo.

Así, nuestra propuesta se basa en la creación un patrón específico, que permite gestionar y controlar los riesgos en entornos Big Data, que ha sido creado en base a las recomendaciones dadas por la normativa ISO/IEC 27000 y en la arquitectura de referencia para Big Data propuesta por la organización NIST [16] que contempla las necesidades inherentes de este tipo de sistemas. Es necesario destacar, que este patrón para la gestión y control de riesgos no tiene el objetivo de cubrir todos los riesgos de organización que podrían afectar de forma indirecta al entorno Big Data, ya que se asume que debería estar incluido dentro de un marco de gestión de riesgos más global de la compañía.

El resto de este trabajo se organiza del siguiente modo: En primer lugar se realiza una introducción a los antecedentes relacionados con el tema, seguido a continuación por una sección que introduce las principales características de la metodología , así como el entorno tecnológico que le da soporte. A continuación, se explica el patrón MARISMA-BiDa, incluyendo el proceso para crearlo y los diferentes estándares y marcos de referencia que tiene en consideración. Finalmente, se incluye una sección de conclusiones y trabajo futuro.

II. Antecedentes

En esta sección se definirán los diferentes marcos, metodologías y estándares relacionados con la gestión y análisis de riesgos. Por otro lado, también se explicarán las diferentes propuestas en este ámbito que se relacionan con sistemas Big Data.

Entre las principales propuestas para el análisis y gestión del riesgo podemos destacar MAGERIT [13], OCTAVE [14], CRAMM [15], los diferentes estándares de ISO/IEC, COBIT o NIST. MAGERIT implementa el proceso de gestión de riesgos dentro de un marco de trabajo para que los órganos de gobierno tomen decisiones teniendo en cuenta los riesgos derivados del uso de tecnologías de información. Por su parte, OCTAVE es una técnica de planificación y consultoría estratégica en seguridad que se basa en el riesgo tecnológico. CRAMM es una metodología de gestión de riesgos que comporta tres fases: identificar, analizar y gestionar los riesgos.

En cuanto a los principales estándares de gestión de la seguridad, muchos de ellos han intentado incorporar dentro de sus procesos el análisis y gestión del riesgo: la familia de normas ISO/IEC 27000 y específicamente la norma ISO/IEC 27005 [17] establece las directrices para la gestión del riesgo en la seguridad de la información. La norma ISO/IEC 21827/SSE–CMM [18, 19] establece un modelo de capacidad y madurez en la ingeniería de seguridad de sistemas, que describe las características esenciales de los procesos que deben existir en una organización para asegurar una buena seguridad en los sistemas, incluyendo en las fases previas un proceso orientado al riesgo. El estándar ISO/IEC 15443 [20] clasifica los métodos existentes dependiendo del nivel de seguridad y de la fase del aseguramiento. La evaluación del aseguramiento se divide en proceso, producto y ambiente, mientras que las fases del análisis del riesgo son diseño/implementación, integración/verificación, réplica, transición y operación. ITIL [21, 22] ofrece un elemento para una correcta gestión de riesgos: el conocimiento actualizado y detallado de todos los activos de la organización y de las relaciones, pesos y dependencias entre ellos. Dicho conocimiento es administrado por ITIL desde el proceso de gestión de la configuración de soporte al servicio, y mediante el uso de la herramienta básica sobre la que se construye una aproximación coherente a la gestión eficiente de las TI, la CMDB (Configuration Management Database). COBIT [23] es una metodología para el adecuado control de los proyectos de tecnología, los flujos de información y los riesgos que implica la falta de controles adecuados. Incluye un proceso orientado a evaluar los riesgos, el cual, se centra principalmente en los criterios de confidencialidad, integridad y disponibilidad, y de forma secundaría en criterios de efectividad, eficiencia, cumplimiento y confiabilidad. Por último, la organización NIST [24] ha propuesto un marco de gestión de riesgos genérico que es aplicable a cualquier sistema de información.

En cuanto a propuestas para el análisis y gestión de la seguridad de riesgos en sistemas Big Data, no se han encontrado propuestas concretas. Si bien, se han localizado algunos artículos que comentan la problemática, ninguno llega a establecer una metodología específica. En [25] se realiza una análisis de diferentes controles de seguridad en Big Data. Paryasto et al. [26] llegan a la conclusión de que para evaluar el riesgo de seguridad en Big Data se podría adaptar el proceso propuesto por NIST. Damiani [27] defiende la necesidad de realizar un análisis de riesgo específico para sistemas Big Data debido a sus características inherentes que pueden causar la aparición de nuevas amenazas como las fugas de datos masivos, y para manejar estas amenazas aboga por un proceso similar al de la propuesta del NIST. Por otro lado, la mayoría de los trabajos que relacionan el análisis de riesgos con Big Data hacen uso de las ventajas de esta tecnología para analizar y gestionar los riesgos de otros

sistemas de información [28], [29]. Es decir, utilizan Big Data como herramienta, pero no como sistema foco sobre el cual analizar y gestionar sus riesgos propios.

Finalmente, como forma de solucionar la necesidad de realizar el análisis y gestión de riesgos en Big Data hemos decidido utilizar la metodología MARISMA, que se desarrolló tratando de mejorar las propuestas anteriores, y que, dado que cuenta con patrones como mecanismo para ser extendida y adaptada a nuevos contextos, pensamos que es adecuada para nuestro objetivo.

III. MARISMA

A lo largo de los últimos años, y utilizando el método científico Investigación-Acción se identificó una serie de carencias asociadas a las principales propuestas de métodos y procesos para abordar el análisis y gestión de riesgos, que cuestionaban su efectividad, y que comprometía por lo tanto su valor para las compañías. Para tratar de solucionar dichos problemas, se desarrolló la metodología denominada MARISMA, así como una herramienta que diera soporte a dicha metodología, denominada eMARISMA (www.emarisma.com).

Entre la principal problemáticas que se identificó, se pueden destacar: i) Elevado coste y complejidad a la hora de hacer un análisis de riesgos; ii) Su falta de orientación a las PYMES; iii) El resultado no ayudaba a que el siguiente análisis de riesgos fuera más sencillo, al no contar con procesos de reutilización de conocimiento, y iv) Los análisis de riesgos eran estáticos.

Todos estos problemas hacían que las empresas no vieran un valor real en el análisis de riesgos. Para solucionar los problemas detectados en el análisis y gestión del riesgo, se desarrolló una metodología orientada a las PYMES y enfocado a reducir los costes de generación y mantenimiento del proceso de análisis y gestión del riesgo. Este proceso se ha obtenido mediante la aplicación del método de investigación en acción y se ha enmarcado dentro de una metodología (MARISMA) que acomete todos los aspectos relacionados con la gestión de la seguridad [30, 31], y bajo la premisa de que cualquier sistemas de Análisis de Riesgos valido para las PYMES también será extrapolable a grandes compañías.

Esta metodología asocia el análisis y la gestión del riesgo a los controles necesarios para la gestión de la seguridad y consta de tres procesos que interactúan entre sí para dar solución a la problemática detectada. En la Figura 1, podemos ver como estos tres procesos se relacionan. El proceso GPRA (Generador de Patrón para Análisis de Riesgos) se encarga de generar y contener los patrones (estructuras de conocimiento que relacionan los elementos necesarios para elaborar un análisis de riesgos), los cuales son instanciados por un cliente a la hora de realizar un análisis de riesgos (proceso GARM - Generador de Análisis y Gestión del Riesgo). Una vez realizado un análisis de riesgos, este sufrirá eventos de seguridad, que serán gestionados mediante el proceso DRM (Gestión Dinámica del Riesgo). Estos eventos de seguridad generaran conocimiento para adaptar los niveles asociados a los elementos del análisis de riesgos, haciendo que se recalcule el riesgo (dinamismo), y permitirá adaptar también los elementos asociados al patrón seleccionado (evolución). Finalmente, la propia evolución del patrón, permitirá sugerir mejoras a todos los clientes que usaron ese patrón para analizar sus riesgos. A continuación, se explicará de forma detallada el contenido de los patrones utilizados en MARISMA, así como se resumirá la principal funcionalidad de la herramienta que le da soporte automatizado.

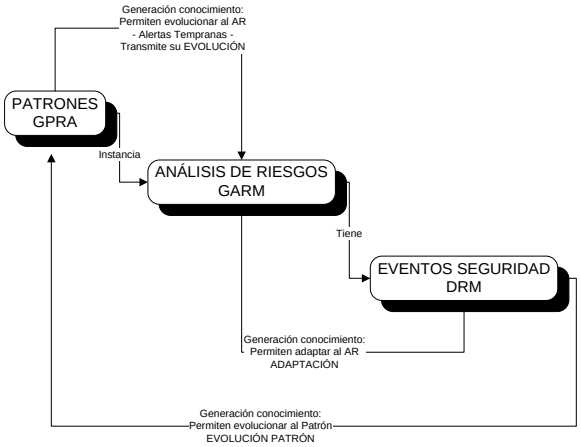


Fig. 1. Esquema general de procesos de MARISMA

A. *Patrón en MARISMA.*

Uno de los grandes problemas que surgió desarrollar con clientes los distintos análisis de riesgos, era ver la forma de poder reutilizar el conocimiento y experiencia que adquirían los consultores al ejecutar el proceso.

Para solucionar este problema, se requería ver qué elementos componían los diferentes análisis de riesgos, con independencia de las normativas que se utilizaran y qué relaciones debían establecerse entre ellos, con el objetivo de crear una meta-estructura que pudiera contenerlos. De este análisis se identificó que todos los análisis de riesgos, tenían controles, activos de información y amenazas, y que estos elementos estaban relacionados entre sí.

Un patrón contiene, por lo tanto, los elementos necesarios para llevar a cabo un proceso de análisis y gestión de riesgos en un contexto determinado. Por ejemplo, el primer patrón desarrollado, fue un patrón de gestión de la seguridad genérico, y para su desarrollo se utilizaron recursos de la ISO27001 y de Magerit. La estructura que contiene todos estos elementos, se ha denominado CAT (Control – Assets – Theats), al ser éstos los tres elementos base de un análisis de riesgos. Dicho meta-patrón será la estructura común a todos los patrones. En la Figura 2, podemos ver los elementos básicos que contienen el meta-patrón CAT de MARISMA, que explicamos a continuación:

- Dominios, Objetivos de Control y Controles: Los Dominios son las agrupaciones de los objetivos de control para un mismo aspecto (legal, tecnológico, organizativo, etc). Por su parte, los objetivos de control, son una agrupación de controles que comparten el mismo objetivo frente a amenazas. Los controles son aquellos elementos que nos van a permitir frenar las amenazas y que normalmente se pueden extraer de estándares internacionales (Ej: COBIT, ISO27001, NIST, etc).

- Tipos de Activos y Criterios de Riesgos: Definen los tipos de activos de información que queremos proteger (Ej: servidores, software, personas, etc.) y los criterios de riesgos que se pueden ver afectados (Ej: confidencialidad, disponibilidad, etc.).
- Tipos de Amenazas y Amenazas: Las amenazas son aquellos elementos que intentaran encontrar vulnerabilidades en los controles para causar un impacto en los criterios de riesgo de los activos. Estás amenazas dependerán del tipo de análisis de riesgos que se quiere realizar, y podrán agruparse en tipos.
- La Matriz Objetivos-Amenaza: Establece la relación de dependencia entre los controles (a nivel de objetivo de control) y las amenazas. Es decir, cada control se define con el objetivo de control para uno o varios tipos de amenaza, estableciéndose una dependencia entre esos dos elementos.
- La Matriz Tipo Activo – Amenaza: Nos permite establecer la relación entre las amenazas y los criterios de riesgos que pretenden atacar para cada tipo de activo.

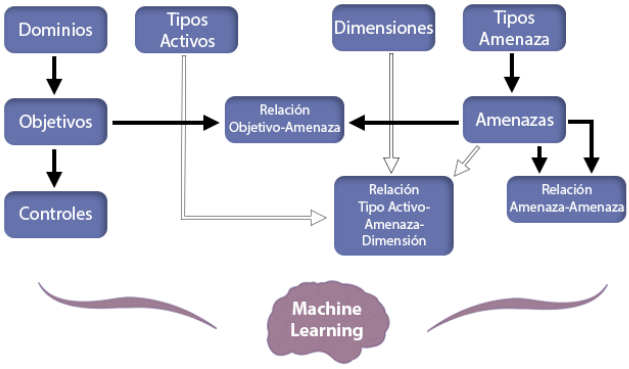


Fig. 2. Esquema general de un patrón de MARISMA

Una vez definidos estos elementos se crea un patrón válido para realizar un análisis de riesgos dentro de una compañía.

B. Herramienta eMARISMA.

Para dar soporte a la metodología MARISMA, se desarrolló en modelo SaaS y con tecnología Java una herramienta denominada eMARISMA (www.emarisma.com), que permite soportar todos los procesos que componen la metodología. Esta herramienta ha permitido ya la integración de varios patrones, en concreto para la gestión de la seguridad (genérico) y para infraestructuras críticas, y han sido aplicados en varias decenas de compañías de España y Colombia. A continuación, se muestran algunas capturas de pantalla de la herramienta y se explica de forma breve los procesos de cada una de las zonas.

- GPRA: eMARISMA tiene una zona que permite visualizar los diferentes patrones existentes, y utilizarlos como base para crear otros patrones (Ej: patrones sectoriales). El proceso GPRA también almacenara el conocimiento generado en el GSS (Global Security Shield – Escudo de Seguridad Global), que se alimentara de los eventos de seguridad de proceso DRM, utilizando dicha información para añadir métricas de riesgo externo y temporal en las compañías.
- GARM: en eMARISMA, mediante la selección del patrón más adecuado y la identificación de un pequeño conjunto de los principales activos se obtiene un detallado mapa de la situación actual (análisis del riesgo) y un plan de recomendaciones de cómo mejorarlo (gestión del riesgo). Este sistema irá evolucionado de forma dinámica según se vayan produciendo eventos en el sistema (ver Figura 3). Este proceso está formado por cuatro etapas: i) Se crea el árbol de Análisis de Riesgos de la compañía. La herramienta eMARISMA no solo permite realizar un análisis de riesgos sobre una compañía, sino todo un árbol de jerarquías de riesgos; ii) Se seleccionan los activos que estarán dentro del alcance de la auditoria, pudiendo solicitar también activos compartidos a otras compañías. Estos activos pueden agruparse en grupos de activos diferentes. Cada grupo de activo tendrá asociado un checklist de controles diferentes; iii) Se seleccionan la probabilidad de ocurrencia de las amenazas, y el porcentaje en que degradan los criterios de riesgo de los activos; y iv) Finalmente, el sistema automáticamente realiza un análisis de riesgos, y calcula el plan de tratamiento de riesgos más adecuado para que la compañía alcance un nivel de riesgo dentro de los limites definidos de la forma más óptima.

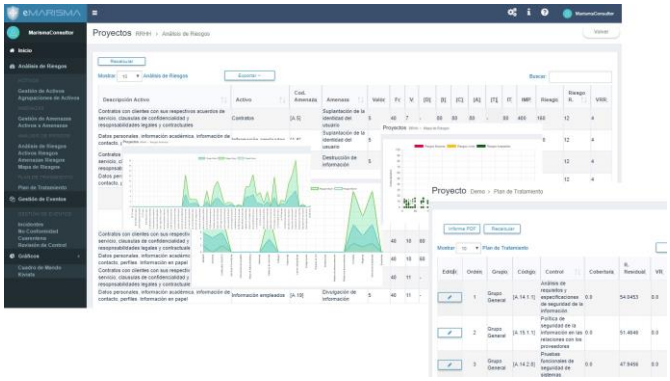


Fig. 3. Herramienta eMARISMA. Generador de Análisis de Riesgos. Creación del Análisis y el Plan de Tratamiento del Riesgo.

- DRM: Mediante la utilización de los elementos que conforman el patrón, las cuáles interconectan los diferentes artefactos del mismo, el sistema irá recalculando el análisis de riesgos según se produzcan eventos de seguridad, fallen las métricas definidas o los auditores detecten “no conformidades” en los controles. Esto dotara al sistema de capacidad de autocorregirse, permitiendo evolucionar a los patrones y adaptarse a los cambios, también dotara de dinamismo al análisis de riesgos, permitiéndole evolucionar con cada evento de seguridad. Este proceso está formado por dos elementos (ver Figura 4): i) Eventos de Seguridad: La herramienta eMARISMA permite diferentes posibilidades para facilitar el dinamismo del análisis de riesgos, bien mediante la evolución natural del nivel de cumplimiento de los controles, o bien mediante la categorización de los eventos de los elementos involucrados en los eventos de seguridad. La introducción de estos elementos, permite aprender, adaptarse y evolucionar al análisis de riesgos y al patrón que lo instancia, dotando al sistema de inteligencia y dinamismo y ii) Cuadros de Mando: Finalmente en la

Figura 10, podemos ver como la herramienta representa mediante cuadros de mandos y kiviat los niveles de seguridad que en cada instante tiene la compañía, de forma que se puede realizar un seguimiento en tiempo real del riesgo de la compañía.

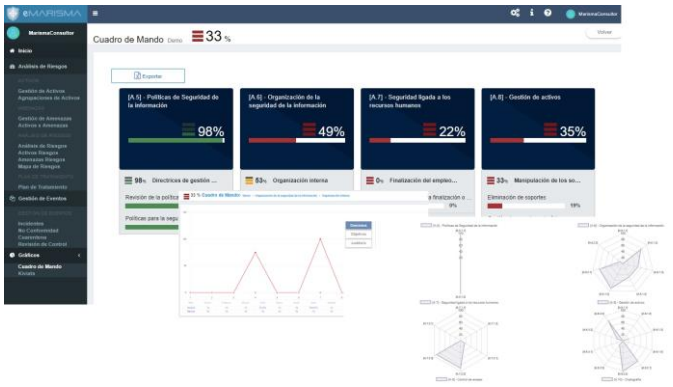


Fig. 4. Herramienta eMARISMA. Gestor de Eventos. Cuadros de Mando.

De esta forma, se ha desarrollado un proceso de muy bajo coste de mantenimiento para la compañía y que le permite tener un sistema de análisis de riesgos completamente dinámico y con la capacidad de informar en todo momento de la situación real de la compañía, reducir el nivel de subjetividad mediante la adaptación de los parámetros del riesgo e incluir la capacidad de no solo tener en cuenta el riesgo interno, sino también el riesgo externo y temporal al que están sometidas las compañías.

La metodología MARISMA permite desarrollar un marco de gestión de la seguridad propuesto por el NIST y especializado en el campo del Big Data. Para ello se definiría un patrón especializado, utilizando el proceso GPRA y él meta-patrón CAT. Se trata de un marco genérico aplicable a cualquier sistema de información.

IV. MARISMA-BIDA

En esta sección se define en cada una de las subsecciones los diferentes componentes que forman parte del patrón específico para la gestión y control de riesgos en Big Data, MARISMA-BiDa.

A. *Dominios, Objetivos y Controles*

Para definir los diferentes dominios y objetivos de control de nuestro patrón, se ha tomado como base la normativa ISO/IEC 27000. Se han estudiado los dominios y controles que expresan y se han adaptado a las necesidades específicas de Big Data. Tal y como se expresó en la introducción del artículo, se parte del supuesto de que la gestión y control de riesgos del sistema Big Data, se encuentra encuadrada dentro de una gestión de riesgos más global y genérica de la compañía. Por ello, algunos dominios y controles no se consideran necesarios para el patrón que nos ocupa. Los dominios, objetivos y controles resultantes se encuentran en la Tabla I. En cada uno de los controles se muestra entre paréntesis el tipo de relación con los controles propuestos por la norma ISO/IEC 27000: N (control nuevo), A (adaptado respecto a un control del estándar) y NM (no modificado, control existente en la norma). Para mejorar la legibilidad de las tablas solo se encuentran aquellos que son aplicables a un entorno Big Data.

B. *Tipos de activo*

Para especificar los diferentes tipos de activo se han considerado principalmente la arquitectura de referencia para Big Data realizada por la organización NIST específicamente el volumen dedicado a definir una taxonomía de Big Data [32]. Este documento indica un conjunto de tipos de activo que son típicos en sistemas Big Data. La Figura 5, muestra los tipos de activo en el patrón:

- Infraestructura hardware: son los recursos hardware necesario para que el entorno realice cualquier tipo de computación. Esto incluye cualquier sistema de almacenamiento.
- Servicios y Aplicaciones: catálogo de acciones que el entorno Big Data ofrece.
- Datos: este es el activo principal de un entorno Big Data. Incluye no solo el dato en sí, sino también sus metadatos.
- Recursos analíticos: esta categoría engloba los diferentes protocolos y algoritmos para realizar análisis de datos en Big Data.
- Técnicas de seguridad y privacidad: este tipo de activo incluye cualquier recurso relacionado con la seguridad, como por ejemplo, cualquier documento con las mejores prácticas de seguridad, métodos de criptografía o información sobre cómo realizar el control de acceso.
- Individuos y roles: cualquier tipo de actor que interaccione con el entorno Big Data a lo largo de su ciclo de vida.

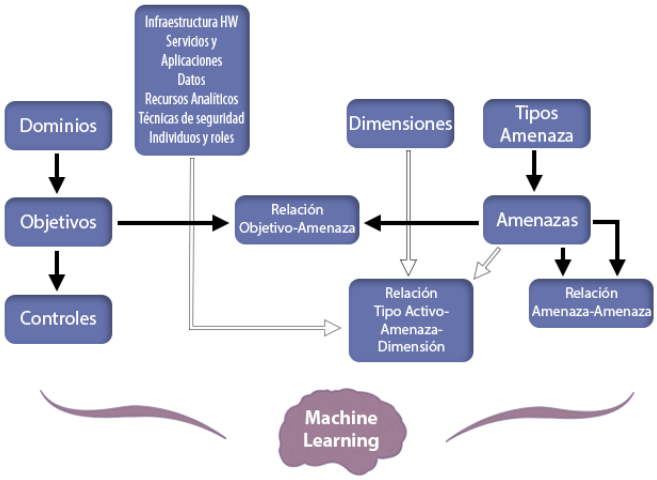


Fig. 5. Tipos de activo en MARISMA-BiDa

C. *Dimensiones*

Para las dimensiones del patrón MARISMA-BiDa, se han considerado las diferentes Vs típicas de los sistemas Big Data. En un principio se definieron 3 Vs básicas [6]:

- Volumen: se refiere a la cantidad de datos creados desde las diferentes fuentes de datos.

- Velocidad: es la tasa de generación de los datos. La rapidez a la que estos se crean y deben ser analizados. Puede llegar a ser en tiempo real.
- Variedad: actualmente los datos no solamente se encuentran en formato estructurado, como en una base de datos, sino que también se hayan en muchas formas como audio, imágenes o documentos.

Esta serie de características complica el análisis de los datos por medio de los sistemas tradicionales. Además de estas características, diversos autores han añadido nuevas Vs que añaden complejidad a la vez que clarifican la definición de un sistema Big Data. Sobre estas nuevas Vs no hay un consenso general, como lo hay con las 3 Vs básicas, de entre todas las propuestas hemos querido destacar estas tres nuevas Vs [33]:

- Veracidad: los datos y fuentes de datos de donde se obtienen deben ser confiables con el fin de que la información resultante sea veraz.
- Variabilidad: los datos no solo se encuentran en diferentes formatos, sino que también cambian constantemente.
- Valor: probablemente una de las Vs más importantes. Nunca hay que perder la perspectiva de que el objetivo de Big Data es obtener información útil de los datos que se disponen. Un error común en las organizaciones es establecer un sistema Big Data por moda, sin tener claro cuál es su objetivo principal.

D. Tipo amenaza y amenazas

Para definir los diferentes tipos de amenaza que pueden afectar a un entorno Big Data se han seguido las recomendaciones dadas por la Agencia de Seguridad de las Redes y de la Información de la Unión Europea (ENISA), la cual, en [34] crea una visión general junto con una guía de buenas prácticas sobre cómo gestionar las amenazas en Big Data. ENISA identifica los siguientes tipos de amenaza:

- Daño no intencional: este tipo de amenaza incluye compartir o filtrar información debido a errores humanos, intervención no intencional o un uso erróneo de la administración de los sistemas.
- Escucha ilegal, intercepción o *hijacking*: este grupo incluye aquellas amenazas debidas a la alteración o manipulación de las comunicaciones entre dos componentes. Estos ataques no requieren de la instalación de nuevas herramientas o software en la infraestructura de la víctima.
- Actividades nefarias o abuso: este conjunto de amenazas engloba todas aquellas derivadas de una actividad delictiva. En contraposición con el tipo anterior, estas amenazas normalmente requieren que el atacante lleve a cabo algunas acciones previas para modificar la infraestructura de la víctima mediante el uso de herramientas y software específicos. Un ejemplo de este tipo de amenaza es un ataque de denegación de servicio (DoS).
- Legales: en este tipo de amenaza se recogen todas las amenazas que se producen debido a las implicaciones legales de un entorno Big Data, como el incumplimiento de leyes o regulaciones, el incumplimiento de requisitos de un contrato o el uno no autorizado de recursos con propiedad intelectual.
- Organizacionales: en este tipo de amenazas se incluyen todas aquellas que provengan de la esfera organizacional, como por ejemplo, la falta de formación del personal que gestiona el entorno Big Data.

Para la definición de las distintas amenazas que se pueden identificar en este tipo de sistemas hemos definido una matriz, en la cual, se integran por un lado los diferentes tipos de amenazas y por otro las diferentes dimensiones de Big Data. Si una amenaza no encaja en ninguna de las celdas de la matriz, se puede concluir que no se trata de una amenaza típica de un entorno Big Data. La Tabla II muestra un ejemplo de cómo se podrían identificar diferentes amenazas.

Tabla I. DOMINIOS, OBJETIVOS Y CONTROLES DEL PATRÓN MARISMA-BIDA

Dominio	**Objetivo**	**Controles**
Gestión de activos	Responsabilidad de los activos de Big Data	Inventario de activos de Big Data (A)
		Propiedad de los activos de Big Data (A)
		Uso aceptable de los activos de Big Data (A)
		Devolución de los activos de Big Data (A)
	Clasificación de la información	Clasificación de la información (NM)
		Etiquetado de la información (NM)
		Manipulado de la información (NM)
Control de acceso	Requisitos de negocio para el control de acceso	Política de control de acceso (NM)
		Acceso a las redes y a los servicios de red (NM)
	Gestión de acceso de usuario	Registro y baja de usuario (NM)
		Provisión de acceso de los roles típicos de Big Data (A)
		Gestión de privilegios acceso de roles típicos de Big Data (A)
		Gestión de la información secreta de autenticación de usuarios (NM)
		Revisión de los derechos de acceso de usuario (NM)
		Retirada o reasignación de los derechos de acceso (NM)
	Responsabilidad de usuario	Uso de la información secreta de la autenticación (NM)
	Control de acceso a sistemas y aplicaciones	Restricción de acceso a la información (NM)
		Procedimientos seguros de inicio de sesión (NM)
		Sistema de gestión de contraseñas (NM)
		Uso de utilidades con privilegios (NM)
		Acceso a la creación/modificación de algoritmos de recolección de datos (N)

		Acceso a la creación/modificación de algoritmos de preparación de datos (N)
		Acceso a la creación/modificación de algoritmos de análisis de datos (N)
		Acceso a la creación/modificación de algoritmos de visualización de datos
		Acceso a la creación/modificación de control de acceso a la información (N)
Criptografía	Controles criptográficos	Política de uso de los controles criptográficos (NM)
		Gestión de claves (NM)
Seguridad física y del entorno	Seguridad de los equipos	Emplazamiento y protección de equipos (NM)
		Seguridad del cableado (NM)
		Mantenimiento de los equipos (NM)
		Reutilización o eliminación segura de equipos (NM)
		Política de aseguramiento de la disponibilidad de los servidores (N)
Seguridad de las operaciones	Protección contra algoritmos maliciosos	Controles contra algoritmos maliciosos (N)
	Copias de seguridad	Disponer de copias de seguridad de la información (NM)
	Control de los algoritmos en explotación	Instalación de los algoritmos en explotación (A)
		Supervisión de los algoritmos en explotación (N)
	Consideraciones sobre la auditoría de Big Data	Controles de auditoría de entornos Big Data (A)
Seguridad de las comunicaciones	Gestión de la seguridad de redes	Controles de red (NM)
		Seguridad de los servicios de red en Big Data (A)
		Segregación de las redes (NM)
Adquisición, desarrollo y mantenimiento de los sistemas de información	Requerimientos de seguridad en sistemas de información	Análisis de requisitos y especificaciones de seguridad de la información
		Protección de las transacciones de servicios de aplicaciones (NM)
	Seguridad en el desarrollo y en los procesos de soporte	Política de desarrollo seguro (NM)
		Procedimiento de control de cambios en el sistemas (NM)
		Revisión técnica de las aplicaciones tras efectuar cambios en el sistema
		Restricciones a los cambios en los paquetes de software (NM)
		Principios de ingeniería de sistemas seguros (NM)
		Entorno de desarrollo seguro (NM)
		Externalización del desarrollo de software (NM)
		Pruebas funcionales de seguridad de sistemas (NM)
		Pruebas de aceptación de sistemas (NM)
	Datos de prueba	Protección de los datos de prueba (NM)
Relación con proveedores	Seguridad en las relaciones con proveedores	Política de seguridad de la información en las relaciones con proveedores
		Requisitos de seguridad en contratos con terceros (NM)
		Cadena de suministro de tecnología de la información y de las
		Aseguramiento de cumplimiento legal de los proveedores (N)
	Gestión de la provisión de servicios del proveedor	Control y revisión de la provisión de servicios del proveedor (NM)
		Gestión de cambios en la provisión del servicio del proveedor (NM)
Aspectos de seguridad de la información para	Redundancias	Disponibilidad de los recursos de tratamiento de la información (NM)
Cumplimiento	Cumplimiento de los requisitos legales y contractuales	Identificación de la legislación aplicable y de los requisitos contractuales
		Protección y privacidad de la información de carácter personal (NM)
		Regulación de los controles criptográficos (NM)

Tabla II. MATRIZ DE IDENTIFICACIÓN DE AMENAZAS EN ENTORNOS BIG DATA

	Daño no intencional	**Escucha ilegal**	**Nefario**	**Legales**	**Organizacional**
Volumen	Error configuración		Abuso de técnicas de	Violación de legislación	
Variedad		Intercepción de información		Abuso de datos personales	
Velocidad	Mal diseño		DoS/DDoS		Falta de formación
Variabilidad		Intercepción de información		Pérdida de control de los	
Veracidad			Inyección de código		Falta de formación
Valor	Filtrado información	Pérdida de exclusividad	Código malicioso		

E. Otras relaciones

Finalmente, el otro componente que forma parte de los patrones en MARISMA son las diferentes relaciones entre los componentes. No es necesario predefinir estas matrices, dado que la propia herramienta eMARISMA permite completarlas de manera automática a partir de un valor neutro. Dicho valor neutro evolucionará en función de los diferentes incidentes de seguridad que sucedan durante la ejecución.

V. CONCLUSIONES Y TRABAJO FUTURO

En este trabajo muestra la forma en que se utiliza la metodología MARISMA y la herramienta eMARISMA que la soporta, para generar un patrón de gestión y análisis de seguridad centrado en aspectos de Big Data, el cual permitirá gestionar de forma dinámica el riesgo asociado a los elementos de un entorno Big Data en una compañía.

Esta propuesta todavía no ha sido aplicada en casos de clientes reales, por lo que contamos en un trabajo futuro inmediato con poder refinarla y validarla con la experiencia de casos reales. Estos refinamientos, estarán principalmente enfocados en ajustar los principales conceptos del patrón MARISMA-BiDa, reafirmando los conceptos más relevantes de los ya identificados, y encontrando algunos otros en base a la experiencia. En un segundo plano, como trabajo futuro a medio plazo, se contempla la evolución de e-MARISMA a un sistema de aprendizaje en la nube que permita incorporar las incidencias de seguridad que afecten a uno de los sistemas, en todos aquellos que se encuentren relacionados o que se puedan ver afectados.

Agradecimientos

Este trabajo ha sido financiado por el proyecto SEQUOIA (Ministerio de Economía y Competitividad y el Fondo

Europeo de Desarrollo Regional FEDER, TIN2015-63502-C3-1-R) y el proyecto GENESIS (Consejería de Educación, Cultura y Deportes de la Dirección General de Universidades, Investigación e Innovación de la JCCM). Se agradece el apoyo de las compañías Sicaman Nuevas Tecnologías S.L. (www.sicaman-nt.com) y Marisma Shield S.L. (www.emarisma.com) que han facilitado la validación de casos de estudio y el uso de la herramienta eMARISMA.

REFERENCIAS

[1] J. Akoka, I. Comyn-Wattiau, and N. Laoufi, 'Research on Big Data – A systematic mapping study', *Computer Standards & Interfaces*, vol. 54, no. Part 2, pp. 105–115, Nov. 2017.

[2] V. Mayer-Schönberger and K. Cukier, *Big Data: A Revolution that Will Transform how We Live, Work, and Think*. Houghton Mifflin Harcourt, 2013.

[3] S. Sagiroglu and D. Sinanc, 'Big data: A review', *Collaboration Technologies and Systems (CTS), 2013 International Conference on*, pp. 42–47, May 2013.

[4] David Reinsel, John Gantz, John Rydning, 'Data Age 2025: The Evolution of Data to Life-Critical. Don't Focus on Big Data; Focus on the Data That's Big', Apr. 2017.

[5] I. A. T. Hashem, I. Yaqoob, N. B. Anuar, S. Mokhtar, A. Gani, and S. Ullah Khan, 'The rise of "big data" on cloud computing: Review and open research issues', *Information Systems*, vol. 47, pp. 98–115, 2015.

[6] M. Chen, S. Mao, and Y. Liu, 'Big data: A survey', *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[7] A. M. Barrientos and K. A. Areiza, 'Integration of a safety management system withan information quality management system.', Universidad EAFIT, 2005.

[8] R. Fredriksen, M. Kristiansen, B. A. Gran, K. Stølen, T. A. Opperud, and T. Dimitrakos, 'The CORAS framework for a model-based risk management process', presented at the 21st International Conference on Computer Safety, Reliability and Security (Safecomp 2002), 2002, pp. 94–105.

[9] H. Wang, X. Jiang, and G. Kambourakis, 'Special issue on Security, Privacy and Trust in network-based Big Data', *Information Sciences: an International Journal*, vol. 318, no. C, pp. 48–50, 2015.

[10] P. P. Sharma and C. P. Navdeti, 'Securing big data hadoop: a review of security issues, threats and solution', *Int. J. Comput. Sci. Inf. Technol*, vol. 5, 2014.

[11] B. Thuraisingham, 'Big data security and privacy', in *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*, 2015, pp. 279–280.

[12] G. Disterer, 'ISO/IEC 27000, 27001 and 27002 for information security management', 2013.

[13] Magerit V3, 'Methodology for Information Systems Risk Analysis and Management (MAGERIT version 3)', Ministerio de Administraciones Públicas (Spain), 2012.

[14] C. J. Alberts and A. J. Dorofee, *Managing Information Security Risks: The OCTAVE Approach*. 2002.

[15] CRAMMv5. 0, 'CRAMM v5.0, CCTA Risk Analysis and Management Method.', 2003.

[16] NBD-WG, NIST, 'NIST Big Data Reference Architecture', 2015. [Online]. Available: https://bigdatawg.nist.gov/_uploadfiles/M0639_v1_9796711131.docx. [Accessed: 18-Oct-2017].

[17] ISO/IEC27005, 'ISO/IEC 27005:2011, Information Technology - Security Techniques - Information Security Risk Management Standard (under development).', 2011.

[18] ISO/IEC21827, 'ISO/IEC 21827:2008, Information technology - Systems Security Engineering - Capability Maturity Model (SSE-CMM)', ISO/IEC, 2008.

[19] SSE-CMM, 'Systems Security Engineering Capability Maturity Model (SSE-CMM), Version 3.0. Department of Defense. Arlington VA. 326.', 2003.

[20] ISO/IEC15443-1, 'ISO/IEC TR 15443-1:2012, Information technology -- Security techniques -- A framework for IT security assurance -- Part 1: Overview and framework.', 2012.

[21] ISO/IEC20000-1, 'ISO/IEC 20000-1:2011, Information technology - Service management - Part 1: Specification.', 2011.

[22] ISO/IEC20000-2, 'ISO/IEC 20000-2:2012, Information technology - Service management - Part 2: Code of practice.', 2012.

[23] I. S. A. and C. A. (ISACA), *Cobit 5 A Business Framework for the Governance and Management of Enterprise*. ISACA, 2012.

[24] R. S. Ross and L. A. Johnson, 'Guide for Assessing the Security Controls in Federal Information Systems and Organizations: Building Effective Security Assessment Plans', 2010.

[25] C. Tankard, 'Big data security', *Network security*, vol. 2012, no. 7, pp. 5–8, 2012.

[26] M. Paryasto, A. Alamsyah, and B. Rahardjo, 'Big-data security management issues', in *Information and Communication Technology (ICoICT), 2014 2nd International Conference on*, 2014, pp. 59–63.

[27] E. Damiani, 'Toward big data risk analysis', in *Big Data (Big Data), 2015 IEEE International Conference on*, 2015, pp. 1905–1909.

[28] B. Ale, 'Risk analysis and big data', in *Safety and Reliability*, 2016, vol. 36, pp. 153–165.

[29] T.-M. Choi and J. H. Lambert, 'Advances in risk analysis with big data', *Risk analysis*, vol. 37, no. 8, pp. 1435–1442, 2017.

[30] L. E. Sánchez, A. Santos-Olmo, D. Garcia, and M. Piattini, 'Managing Security and its Maturity in Small and Medium-sized Enterprises', *J. UCS*, vol. 15, pp. 3038–3058, 2009.

[31] A. Santos-Olmo, L. E. Sánchez, E. Fernández-Medina, and M. Piattini, 'A Systematic Review of Methodologies and Models for the Analysis and Management of Associative and Hierarchical Risk in SMEs', presented at the 9th International Workshop on Security in Information Systems (WOSIS12) In conjunction with 11th International Conference on Enterprise Information Systems (ICEIS12)., Wroclaw, Poland, Jun-2012.

[32] NBD-WG, NIST, 'NIST Big Data Taxonomies', 2015. [Online]. Available: https://bigdatawg.nist.gov/_uploadfiles/M0639_v1_9796711131.docx. [Accessed: 18-Oct-2017].

[33] M. Ali-ud-din Khan, M. F. Uddin, and N. Gupta, 'Seven V's of Big Data understanding Big Data to extract value', in *American Society for Engineering Education (ASEE Zone 1), 2014 Zone 1 Conference of the*, 2014, pp. 1–5.

[34] ENISA, 'Big Data Threat Landscape and Good Practice Guide', Jan-2016. [Online]. Available: https://www.enisa.europa.eu/publications/bigdata-threat-landscape/at_download/fullReport. [Accessed: 18-Oct-2017].

Índice de autores

MONDRAGON

HUMANITY AT WORK

Finanzas
Industria
Distribución
Conocimiento

261
ENTIDADES

74.335
PERSONAS

15
CENTROS de I+D

Misión

MONDRAGON es una realidad socioeconómica de carácter empresarial, integrada por cooperativas autónomas e independientes, con hondas raíces culturales en el País Vasco, creada por y para las personas, inspirada en los Principios Básicos de nuestra Experiencia Cooperativa, comprometida con el entorno, la mejora competitiva y la satisfacción del cliente, para generar riqueza en la sociedad mediante el desarrollo empresarial y la creación de empleo preferentemente cooperativo, que:

- Se **sustenta** en compromisos de solidaridad y utiliza métodos democráticos para su organización y dirección.
- **Impulsa** la participación y la integración de las personas en la gestión, resultados y propiedad de sus empresas, que desarrollan un proyecto común armonizador del progreso social, empresarial y personal.
- **Promueve** la formación e innovación desde el desarrollo de las capacidades humanas y tecnológicas, y
- **Aplica** un Modelo de Gestión propio para alcanzar posiciones de liderazgo y fomentar la Cooperación.

Visión

Queremos llegar a ser personas comprometidas y con identidad cooperativa que configuran un Grupo empresarial rentable, competitivo y emprendedor en un contexto global; que aplican un modelo socioempresarial de éxito, ofreciendo soluciones integrales al mercado en base a la experiencia, conocimiento, innovación, intercooperación, alianzas estratégicas, atracción, impulso y generación del talento; y que genera recursos suficientes para aportar empleo de valor añadido y el desarrollo sostenible del entorno.

BASQUE CYBERSECURITY CENTRE

ZIBERSEGURTASUN EUSKAL ZENTROA
CENTRO VASCO DE CIBERSEGURIDAD

Qué es

El Basque CyberSecurity Centre (BCSC) es un centro que organizativamente se ubica dentro de la SPRI y cuyo objetivo es el de generar cultura de ciberseguridad en Euskadi.

Misión

Promover y desarrollar una cultura de la ciberseguridad en la sociedad vasca, dinamizar la actividad económica relacionada con la aplicación de la ciberseguridad y fortalecer el sector profesional.

Visión

Posicionar a Euskadi como un referente internacional en la aplicación de tecnologías de ciberseguridad a la industria, ser reconocido como punto de encuentro entre oferentes y demandantes locales de ciberseguridad, y liderar iniciativas de colaboración público-privadas tanto a nivel local como inter-regional.

Para hacer frente a estos retos el BCSC se alinea en sus iniciativas con agentes de la red vasca de ciencia, tecnología e innovación, y con actores públicos clave de Gobierno Vasco como son EJIE, la Ertzaintza y el Departamento de Educación.

Así mismo, se constituye como un equipo de respuesta a incidentes de seguridad informática para poder dar respuesta a los problemas derivados de los incidentes de seguridad en Euskadi.

Valores

- Objetividad.
- Eficacia, eficiencia y rentabilidad social.
- Dedicación plena, buena fe y ejemplaridad.
- Austeridad y honradez.
- Transparencia, accesibilidad y confidencialidad.
- Ética, responsabilidad y colaboración.

Para construir una Gipuzkoa más competitiva, con capacidad de atracción, posicionada tecnológicamente y que apuesta por la fabricación avanzada y responde de forma eficaz a los retos tecnológicos.

Para proteger y prevenir a nuestras industrias de los ataques cibernéticos y crear un nuevo sector económico especializado en ciberseguridad industrial.