ENSTA Bretagne
2, rue François Verny
29806 BREST cedex
FRANCE
Tel +33 (0)2 98 34 88 00
www.ensta-bretagne.fr

Status Report

promo 2017

March 10, 2017

Sniff Hynesim

RIGAUD MICHAËL

Contents

Contents 1

Introduction 3

I Bibliographic study 4

1 Hynesim 5
1.1 Presentation 5
1.2 Architecture 5

2 IDS 7
2.1 Presentation 7
2.2 Detection methods 8
2.3 Main IDS 9

3 SIEM 10
3.1 Description 10
3.2 Capabilities 10
3.3 Main SIEM 10

II Presentation of the topic 11

4 Presentation of the subject 12
4.1 Model checking 12
4.2 Position of my project 13

5 Technical choice 14
5.1 Application created for test 14
5.2 IDS 15
5.3 SIEM 15

6 Forecasting organization 16
6.1 Kanban 16
6.2 Github 17

III Technical study 19

7 Installation 20

7.1 Hynesim 20
7.2 Selks 21

8 Implementation **22**
8.1 Application 22
8.2 Probe 22
8.3 SIEM 23
8.4 Summary 24

9 Analysis of results **25**
9.1 Results 25
9.2 Way of improve 25

Conclusion **26**

A SELKS **28**

List of Figures **30**

Bibliography **31**

Introduction

Cyber security is one of the major threats of the 21st century, and attackers use increasingly sophisticated techniques. So one of the most important aims for cyber security engineers is to find a way to detect and stop attacks. To do that effectively cyber engineers need to analyze cyber attacks to find a way to detect them. One solution is to use simulators of network and information systems to reproduce scenario of attacks. Simulators permit to achieve attacks in a closed space with a huge agility. To do so, ENSTA Bretagne has decided as Thales and the DGA (French Procurement Agency) to use Hynesim[1].

The aim of this project is to elaborate a system of alert on Hynesim. This system need to analyze network events and applications events. To do that, I have to install an IDS[2] to analyze network traffic and implement an SIEM[3] to raise alert. A better description of the project is available part II.

To begin, I will achieve a bibliographic study which will present Hynesim, IDS and SIEM. Then, I am going to present the subject, the choices made to answer the topic, and the organization set up in this project. And to finish, I will explain how I installed the tools available to me, how I achieved the other tools that I needed, and the results of the project.

[1]This software is presented in the chapter 1
[2]Intrusion detection system
[3]Security information and event management

Part I

Bibliographic study

CHAPTER 1

Hynesim

Firstly, I will present Hynesim. In fact, I will use this tool to create our network and test our solution so it is important to introduce it.

1.1 Presentation

Figure 1.1: Hynesim logo

Définition 1.1 : *Hynesim*
Means HYbrid NEtwork SIMulation, is a distribution platform of simulation of information systems developed by Diateam. [3]

The platform was initially developed by Diateam for DGA[1] MI (Maitrise de l'information) to create virtual networks. But now is a major project to develop information systems and automatize cyber security attacks. This project has two version, an open source version and a professional version. The open source version has less options, but I will use this version for this project.

1.2 Architecture

In order to work, Hynesim needs a server with on it the main software. This software is the virtualization part. It manages virtual machines and networks.

Moreover, to see virtual machine and interact with them, users needed to have a client interface. This interface can be installed on a simple computer.

To add a virtual machine to Hynesim, I need to create it on Virtual Box[2] or VMWare[3] and then import them on Hynesim.

[1] French Procurement Agency
[2] More information here: `https://www.virtualbox.org/`
[3] More information here: `https://www.vmware.com/`

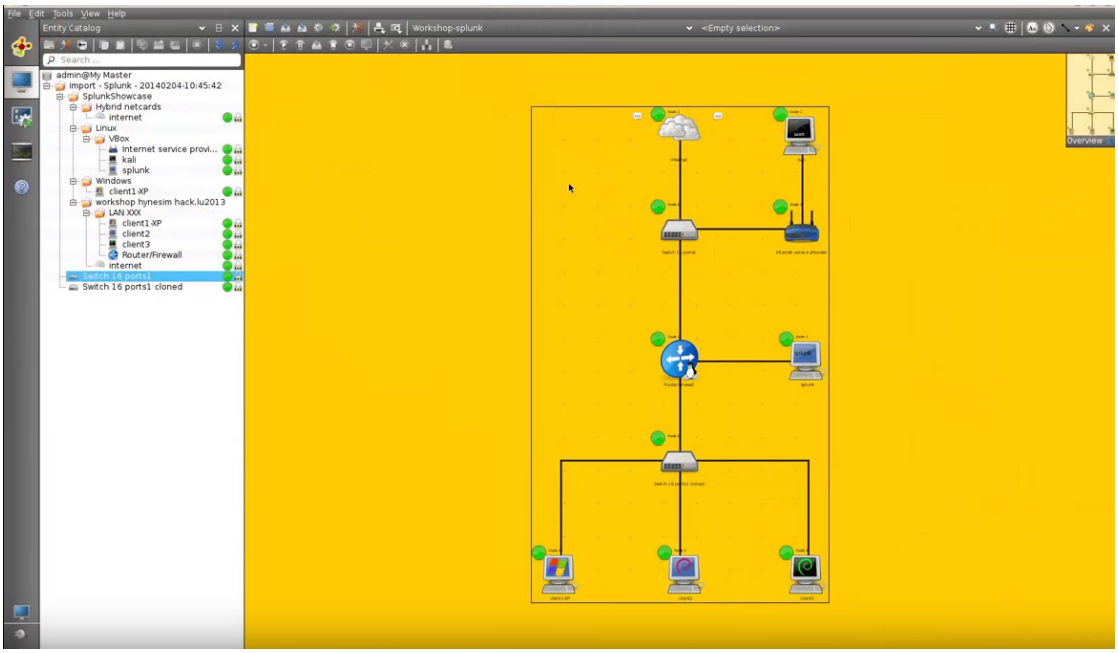

Figure 1.2: An example of a client Hynesim interface

CHAPTER 2

IDS

In this subject, a probe will be created to detect attacks. It is the aim of IDS that I are going to introduce.

2.1 Presentation

Définition 2.1 : *IDS*
An intrusion detection system (IDS) inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. [2]

There are many type of IDS:

- NIDS, network IDS. They listen and analyze the network and detect attacks from network packets. They are the most interesting for our subject, so this document will focus primarily on this type of IDS.
- HIDS, host IDS. These IDS are on a system and they detect intrusion within it.
- Hybrid IDS. They are composed of NIDS and HIDS.
- IPS. They are NIDS with active functions which help to stop attackers.
- KIDS/KIPS, kernel IDS. They are types of HIDS. They are on the system kernel. They are more effective and slower than HIDS.

In the following document I will talk about NIDS. But to detect attacks there is also many methods.

To be efficient IDS should have a good balance between some features.

- **Speed:** In fact an IDS should analyze packets as fast as possible, otherwise, it will behind the network traffic.
- **False alarm:** An IDS raises an alert when it detect attacks. But it could raise alarm during a normal utilization, a false alarm. It is one of the most important features, because at every alarm a system administrator needs to analyze the alert. So in company, every alarm cost time and money.
- **Probability of detection:** For an IDS, the capacity of detecting attacks. The higher the probability the more the IDS will raise false alarms. In some sensitive systems, I prefer to detect every attack and raise many false alarms. That depends on the system.

2.2 Detection methods

2.2.1 Misuse detection

This technique is the simplest. It uses attack signature to raise alerts. In fact, all attacks have a particularity, if I detect this particularity I can detect the attackers. There are three sub methods.

Pattern matching

In this technique I have a base of signature and the IDS looks for the pattern. If the pattern matches perfectly, this IDS raises an alert.

The problem is, only attacks which are in our base can be detected. So if there is a new attack (zero day), or if the attack is not perfectly the same, it cannot be detected.

However, this method is much used because it is high-performance, and with this method the IDS don't raise a lot of false alerts

Dynamic pattern matching

In this techniques the IDS is also based on signature but this data base is dynamic. In other words, the IDS has the faculty of adaptation and learning. The IDS improve its data base of signature automatically.

Protocol analysis

The last sub method I will present is the protocol analysis. This technique is based on the verification of protocol. The IDS will check if flows are compliant with RFC[1] standards. It will verify parameters of packets and fields. An IDS can check many protocols such as FTP, HTTP, ICMP, ...

The advantages of this method is that I can detect unknown attacks contrary to pattern matching. However, software publishers don't often respect RFC so this technique is not always very efficient.

2.2.2 Anomaly detection

This technique consists in detecting an intrusion through the analysis of the user's past behavior. So the IDS should create a profile of users from his use and raises alerts when there is an event outside this profile. To create a profile the IDS could use machine learning.

Advantages	Drawbacks
The IDS should be able to detect every type of attack even unknown (zero days) attacks.	This method is not reliable. Every alteration of the use creates an alert
The IDS is autonomous	This method needs a learning period. In fact, this method needs to learn the habit of users, so I need a period without attacks
	Hackers can need only time. In fact, if the attacker creates a new profile after many month with his attacks, he could attack silently

[1]Requests for Comments, is a type of publication from the Internet Engineering Task Force (IETF) and the Internet Society (ISOC), the principal technical development and standards-setting bodies for the Internet.

Probabilistic method

Bayésien network is a learning machine based on probability. The IDS will create a probabilistic model and will raise an alert if the user don't respect this model.

For example, I know that in 90% of cases in an HTTP request the first parameter is GET after a connection to the port 80.

2.3 Main IDS

There are many IDS on the market. The most popular open source solutions are Snort, Suricata, Bro, Fail2Ban, ACARM ... There are also some tools «all in one». It is usually OS[2] with an IDS, a tool to analyze alert, a tool to create rules,... The most popular are SELKS and ELKS. A description of SELKS is available page 28.

[2]Operating system

CHAPTER 3

SIEM

After some new discussion with my tutor, we decide to had a new component in our architecture. So we will add a SIEM to analyze all alert and take care of server's log. So we will add in this status report a part about SIEM which was not initially.

3.1 Description

Définition 3.1 : *SIEM*
In the field of computer security, security information and event management (SIEM) software products and services combine security information management (SIM) and security event management (SEM). They provide real-time analysis of security alerts generated by network hardware and applications.[11]

So SIEM are software which aggregate all security information, analyze them, and display them for user.

3.2 Capabilities

- **Data aggregation** aggregate data from many sources: network, IDS, log, applications, etc...
- **Correlation** Analyze alerts and events to correlate them to spot an attack.
- **Reporting and alerting** Create alert and reporting more clear.
- **Visualization** SIEM permit visualization of all event with for example dashboard.
- **Log management** Store all log in a central location. This permit to do forensic many month after events.

3.3 Main SIEM

There is many SIEM on the market. The most popular open source solutions are Prelude, OSSIM, and Cyberoram.

Part II

Presentation of the topic

CHAPTER 4

Presentation of the subject

After this bibliographic study, I will present the subject of my work and the aim of it. My work supposed that a model checking was done, so I will firstly explain that.

4.1 Model checking

Since few years, we are able to realize model to represent systems. These models permit to simulate systems. To do these models, there is many languages UML, SysML, Fiacre,... The figure 4.1 represents an example of a complex system.

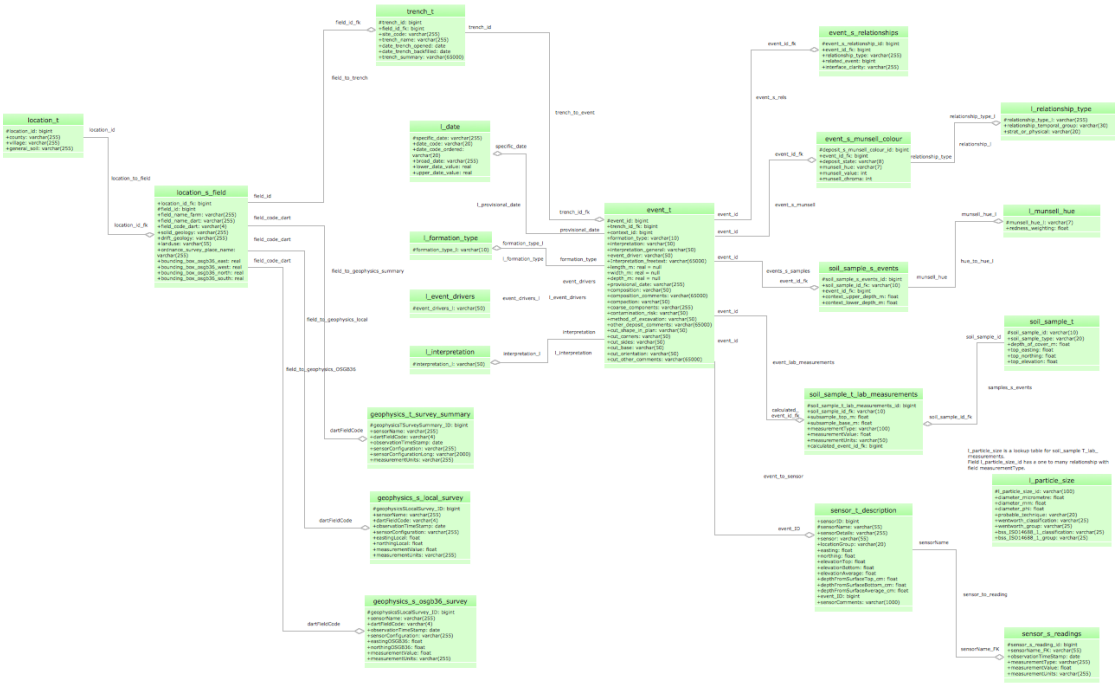

Figure 4.1: Complex model in UML

Then with the simulation of these systems, we can imagine that we are able to find all possible states of our system. So we know that if we are not in a valid state of the system, the system was potentially hacked.

4.2 Position of my project

So for my project, I guess I have all possible states for my system and I will check these possible states with the current status of the system. To do so, I have to find the state of applications and the state of the network, and analyze them.

I choose to use an IDS to have the status of the network. In fact, an IDS can raise alert when there is particular message, so I will raise event when I sniff particular message. To have a status of an application, I choose to implement probe in applications, or if it is possible use the log of these applications. Then, the SIEM will correlate all these messages and alerts to find the status of our system and it will compare it with all previous possible status.

The figure 4.2 represent the position of our project.

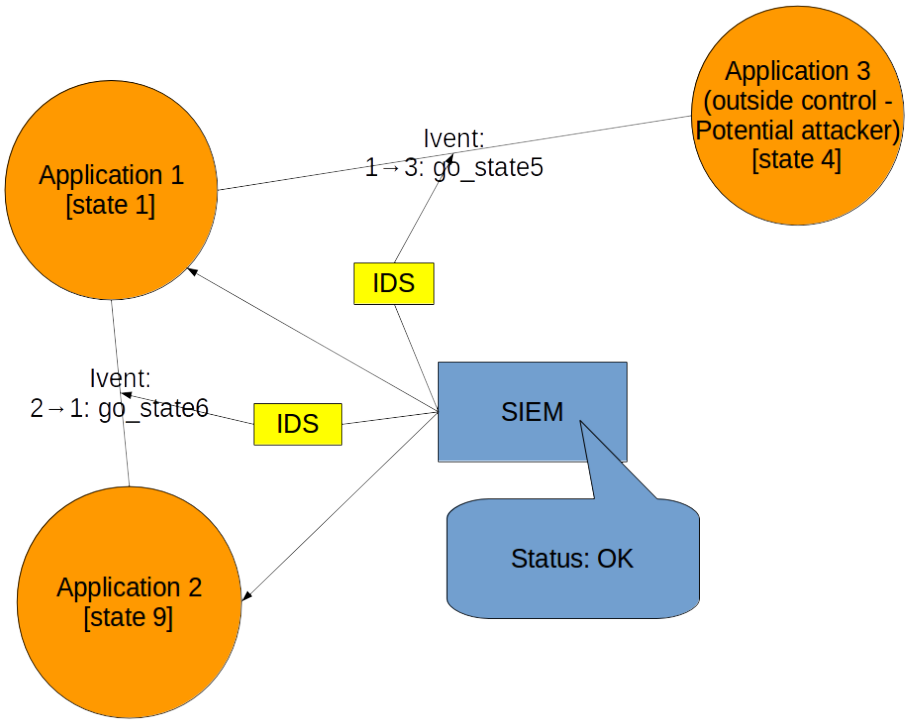

Figure 4.2: Position of my project

CHAPTER 5

Technical choice

5.1 Application created for test

As it was explain before, I have to protect specific applications. To do my tests, and prove the concept of my work, I choose to create a simple application. I plan to add some security issue to be more realistic.

This application has three states, and two parts. Firstly, there is an admin interface which contains the state admin. This interface is connected on the port 9000. Only one specific known computer can access to this interface. Of course, the aim of attackers is to arrive in this state.

Then there is the main application. It is connected to the port 9124, and everybody can connect to it. There is two states ping and pong. If the application is in the state ping and the user send «topong» the application go to state pong, and if the application is in the state pong and the user send «toping» the application go to the state ping. There is also an unknown backdoor, if the user is in the state ping for the third time and he send «admin», the application go to the admin interface.

The figure 5.1 represent a model of our application.

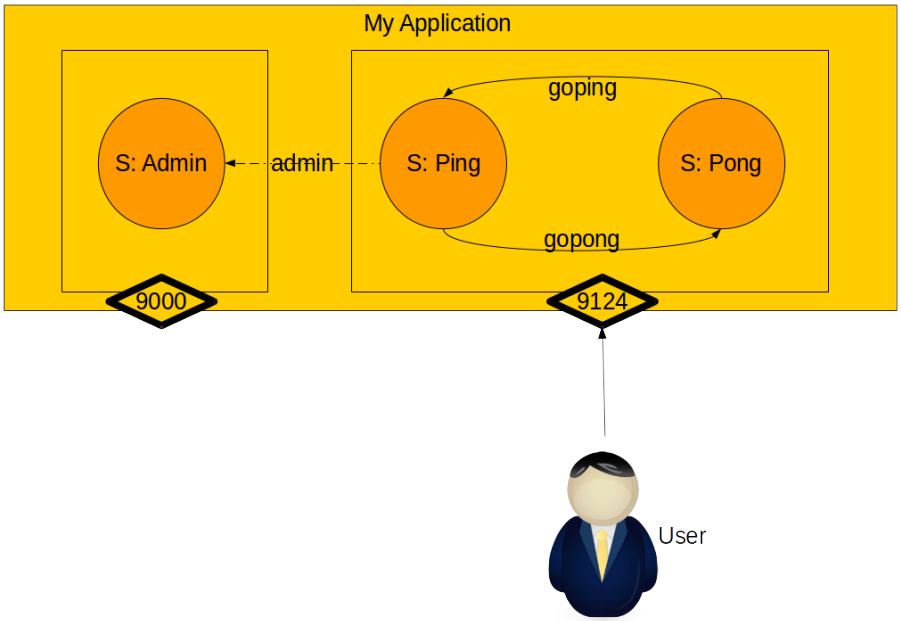

Figure 5.1: Model of the application

5.2 IDS

To monitor the network between my applications, I have to install an IDS. In this project, my IDS only need to check network packets and send alert when it detects some pattern. So I choose to use an IDS which use a misuse detection.

Moreover, my IDS only need to raise alert and save them. It is possible to find on the market of open source software many IDS which do that. So I choose one of the most popular open source IDS, SELKS.

5.3 SIEM

Then, for the SIEM I had to made a choice: use an open source solution of SIEM, or implement my own solution. The table 5.1 lists advantages and drawback of the two solutions.

Prelude	
Advantages	Drawbacks
Already implemented	Need to connect it to SELKS
More modular	Not adapted to my problem
	Need to configure it
	Heavy solution
My SIEM	
Advantages	Drawbacks
Well adapted for my problem	Need time to implement it
Less heavy	Only adapted for my problem

Table 5.1: Prelude vs my SIEM

For all the reasons cited on the previous table, I choose to implement my own SIEM.

CHAPTER 6

Forecasting organization

6.1 Kanban

To achieve this project, we decided to use some tools to arrange our work. First of all, we decided to use an agile technique of management which name Kanban.

Définition 6.1 : *Kanban*
Kanban is a new technique for managing a software development process in a highly efficient way. Kanban underpins Toyota's "just-in-time" (JIT) production system. The kanban system consists of a big board on the wall with cards or sticky notes placed in columns with numbers at the top [9]

Kanban is an inventory-control system to control the supply chain. It uses a board with columns. Each column represents a status, for example: to do, doing, done. In each column we put «notes» which represent a task. Moreover, each column has a maximum number of notes authorized.

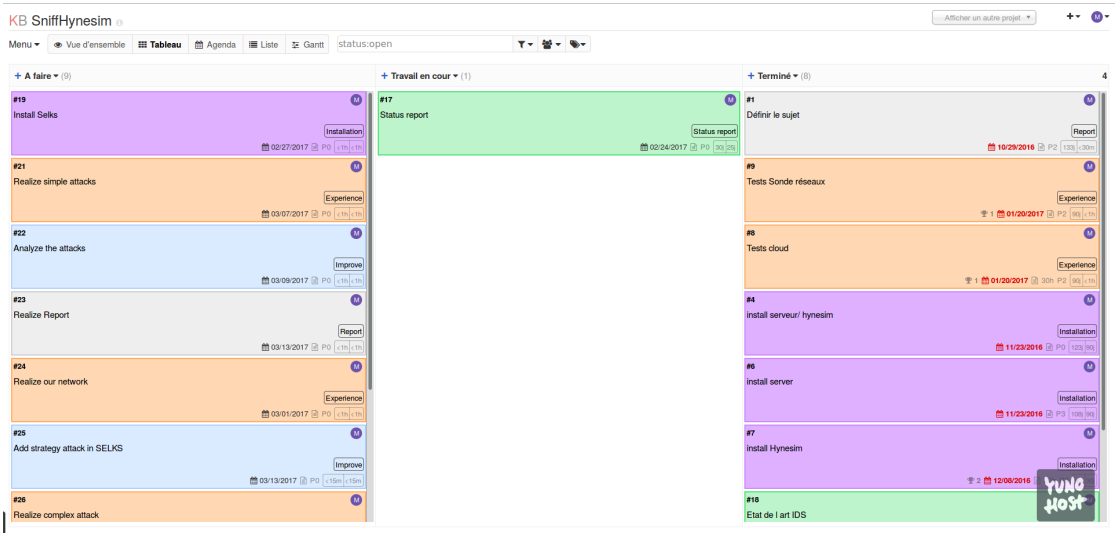

Figure 6.1: Kanboard of this project

Limiting the amount of tasks, at each step in the process, prevents overproduction and reveals bottlenecks dynamically. In fact, with this technique it is possible to have a better overview of the project and control it dynamically.

For this project, we use Kanboard[5] self-hosted on our own Yunohost server.[1]. You can see in the figure 6.1 the kanboard of this project. Each color represent a category of tasks: blue: improvement , purple: installation, red : experience, green: status report, and grey: report.

With this tool, it is also possible to see tasks as a Gantt diagram. The figure 6.2 presents our Gantt diagram.

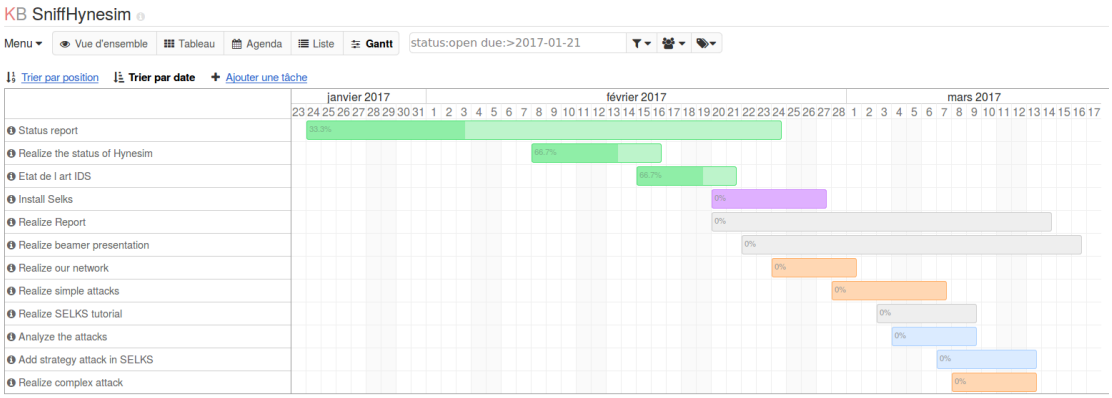

Figure 6.2: Gantt diagram of the project

6.2 Github

To achieve this project, we also decided to use Git and Github as a Git server.

Définition 6.2 : *Git*
Git is a version control system for tracking changes in computer files and coordinating work on those files among multiple people.

Git enables me to control version of our work and obtain a real showcase of it for our tutor. The figure 6.3 is a screenshot of our Github server.

[1] It is possible to see our kanboard at this link: `https://mic-rigaud.fr/kanboard/?controller=BoardViewController&action=readonly&token=10ea65eca908023dbcd8bc8dce75791c7a14d67912627dafaa5b71033222`

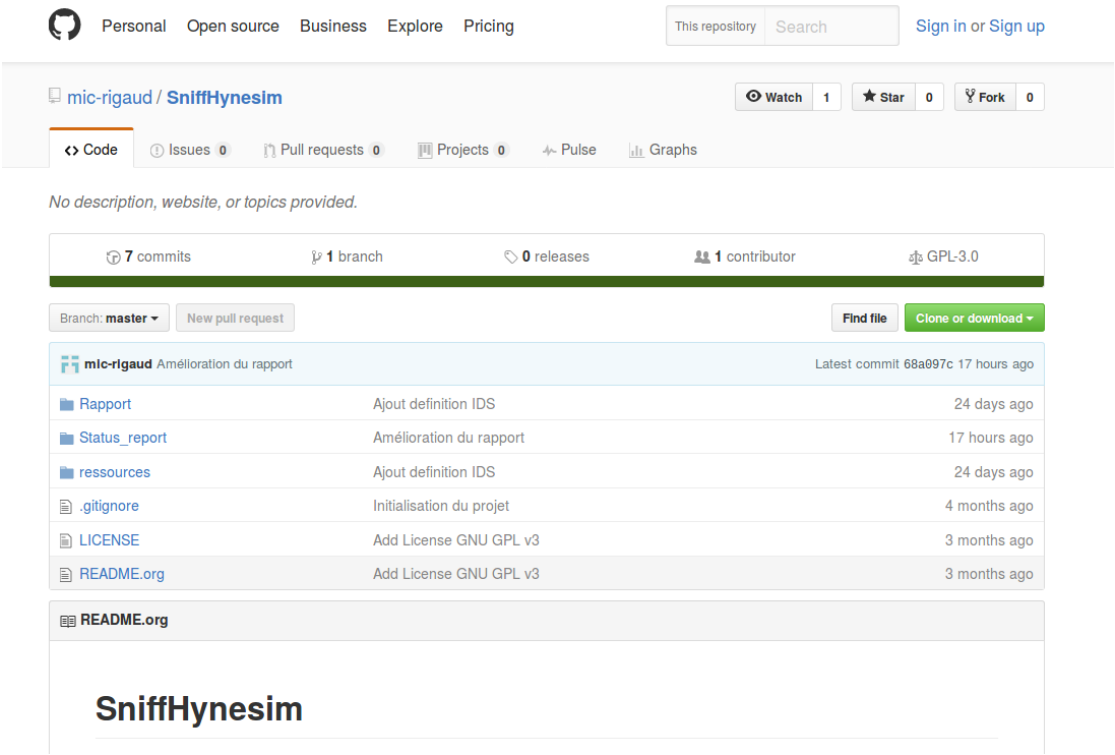

Figure 6.3: Github server

Part III

Technical study

CHAPTER 7

Installation

7.1 Hynesim

7.1.1 Requirement

To use Hynesim, Justin and I had to install the open source version of Hynesim server. To do so, M Champeau give us a computer with 16G of RAM. I installed Debian 8 on this machine. Them I installed the hynesim server with the basics network equipment. I follow the instructions write on the Hynesim web-page.

Then, Justin configure Hynesim to be accessible with the client. I will not detail this part, if you want more information you can read his report.

7.1.2 Import of virtual machines

Hynesim works with many virtual software as qemu, virtualbox, and vmware. For this project I use qemu because it is the most suitable for hynesim. To create virtual computer on hynesim, I firstly created the machine on virtualbox, then I converted it to a qemu virtual machine and to finish I import this machine to hynesim.

Hynesim is also provided with some virtual equipment: a switch, and an inter topology link.

An important detail, on the hynesim network machines haven't access to the internet. So I had to do update, and install before they are imported.

I created the left part of the network represented on the figure 7.1.

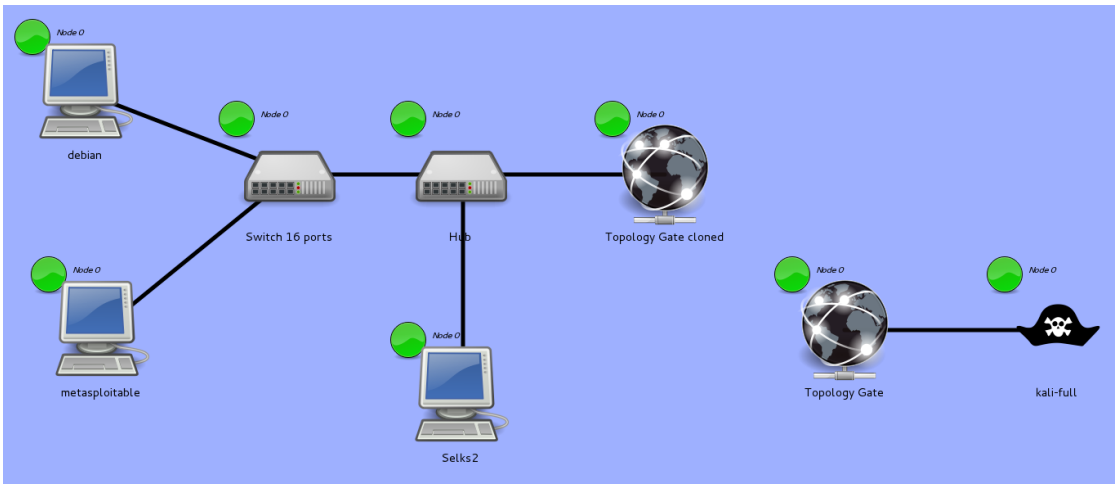

Figure 7.1: Network infrastructure

7.2 Selks

As it was explain on the previous section, before importing the machine I had to create an virtual machine. To do so, I use an ISO available on the Stamus web page [10] and I create a virtual machine with it. Then, before it was imported, I have set up important libraries for the realization of the SIEM[1].

The ISO is a key in hand version. So Suricata, Elasticsearch, Logstash, Kibana, Scirius, and Evebox are already installed and connected. However, I had to configure some particularity and take in charge the system.

The only particularity that I have to configure during the installation was the network. In fact, on the hynesim network there is no DHCP, so I had to configure by hand the IP of the machines. Moreover, I had to configure on Suricata the private address domain. In fact, Suricata doesn't pay so much attention to networks packets from the private network.

Then, I tested my IDS with basic attacks[2]. The can see on the figure 7.2 the evebox interface which permit to visualize alerts raised by Suricata.

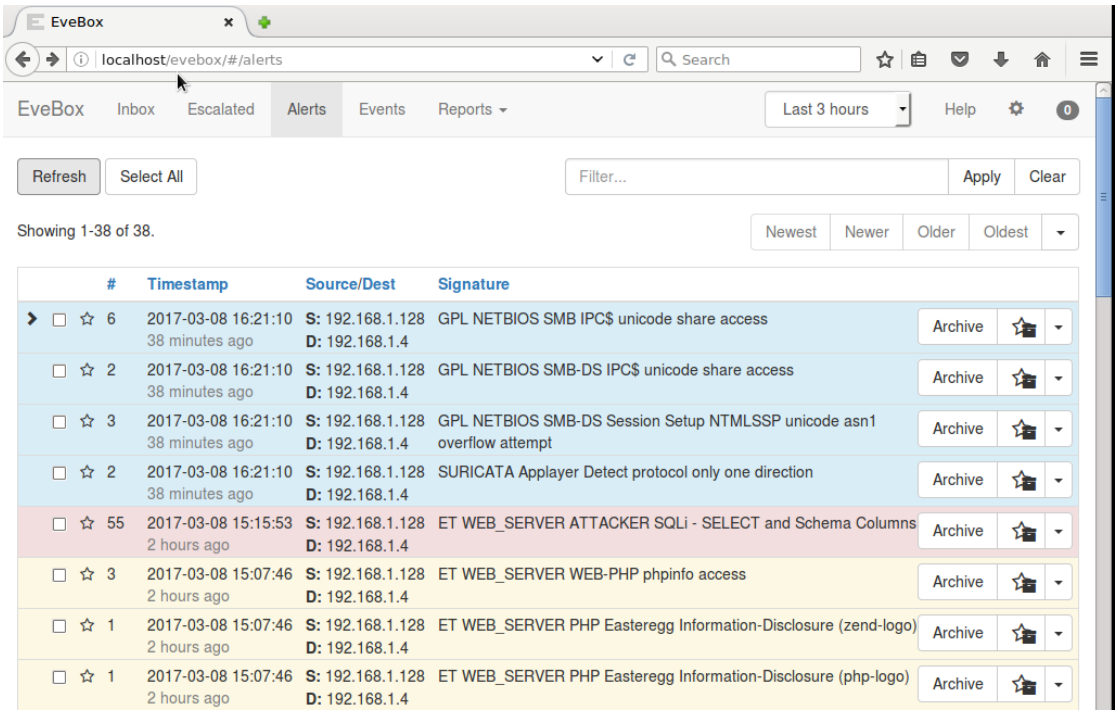

Figure 7.2: Example of alerts

[1]More information on the chapter 8.3
[2]These attacks were achieve by Justin Bouroux

CHAPTER 8

Implementation

In this chapter, I will explain how I implemented my solutions.

8.1 Application

First of all, I implemented the application for the tests. I choose to implement this application in python and it is hosted on the computer named Dedian (cf figure 7.1). As it was explain in the chapter 5, this application has three state: Ping, Pong and Admin.

Considering I have to implement the defense of this application, I have to delimit the sensitive space. It is obvious for this application, the sensitive state is the admin state. So I have to send many event when the user try to access to the admin interface. In this way I could know if somebody try without success to access to the admin interface, this person is doubtless an attacker.

To raise event, I send message to a particular Logstask instance which is on SELKS. This message contain the time and the state of the application. I will explain after, how the Logstash instance work.

Hosted	Debian
Language	Python
States	Ping, Pong, Admin
Sensitive state	Admin
Port open	9124, 9000
Outdoor communication	Selks to port 5000
Port 9124 accept communication from	Everybody
Port 9000 accept communication from	Metasploitable only

Table 8.1: Features of the application

8.2 Probe

8.2.1 Network probe

I already explain that I use SELKS as IDS, but I had to configure it to adapt it to my application. So I had rules on the Suricata ruleset. To add rules, I use the scirius interface which permit to administrate suricata's ruleset. I added the next rules:

```
alert tcp any any -> any 9124 (msg:"Action goping"; \
content:"goping"; sid:501; rev:5000;)

alert tcp any any -> any 9124 (msg:"Action gopong"; \
```

```
content:"gopong"; sid:502; rev:5001;)

alert tcp any any -> any 9000 (msg:"Connexion vers l interface admin"; \
flow:established,to_server sid:504; rev:5002;)
```

8.2.2 Application probe

As it was explain before, to have the status of the application, it sends to Logstash its status. Then Logstash filter these messages and write them on the Elasticsearch data base.

On the SELKS computer, there is a Logstash implemented with a filter adapted for Suricata but not for our application. So I write a filter to Logstash adapted for this application, and I run a new instance of Logstash with this configuration on SELKS. By this way, both instance of Logstash do their job without interaction.

The filter implemented, convert plain text message send by my application as json event readable by Elasticsearch. Moreover, the filter add some tag on this event facilitate the research on the data base. These tags are added on the label «alert_signature_id».

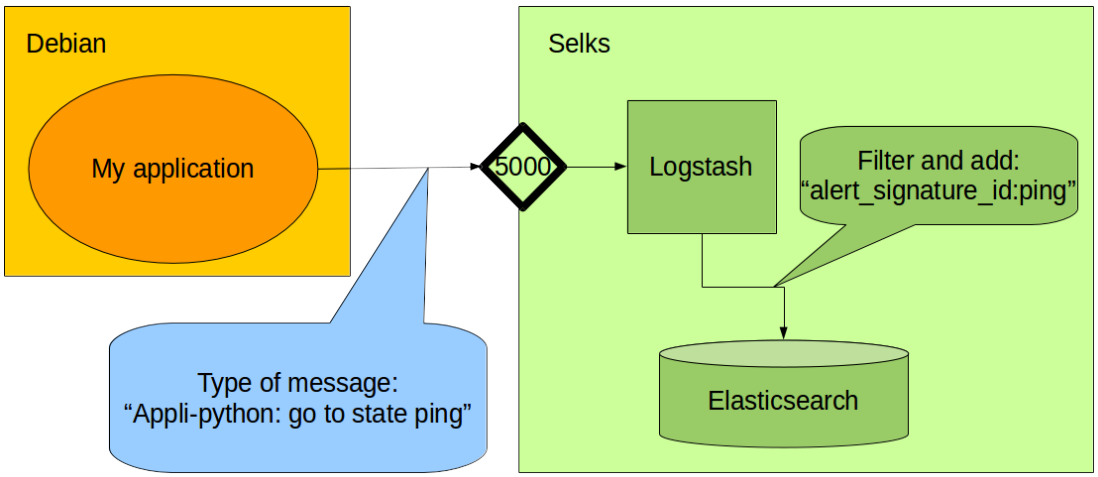

Figure 8.1: Data processing

8.3 SIEM

I also achieve to implement a SIEM. To do so, I used the «elasticsearch» library for python. With it I can make request to the data base easily. I collect the last events, I refresh the interface with these information, and analyze its. After analysis, I am able to say if the application is under attack.

The figure 8.2 represent the interface of the SIEM implemented.

```
root@SELKS:~/Documents/SIEM# ./main.py
##############################################################
                SIEM DEVELOPPE PAR MICHAEL
             Application sous license GPLV2
##############################################################

App-status: Ping   |   Net-status: Admin   |   Attaque: True

##############################################################
root@SELKS:~/Documents/SIEM#
```

Figure 8.2: My siem interface

8.4 Summary

The next figure represents a summary of the infrastructure of our system.

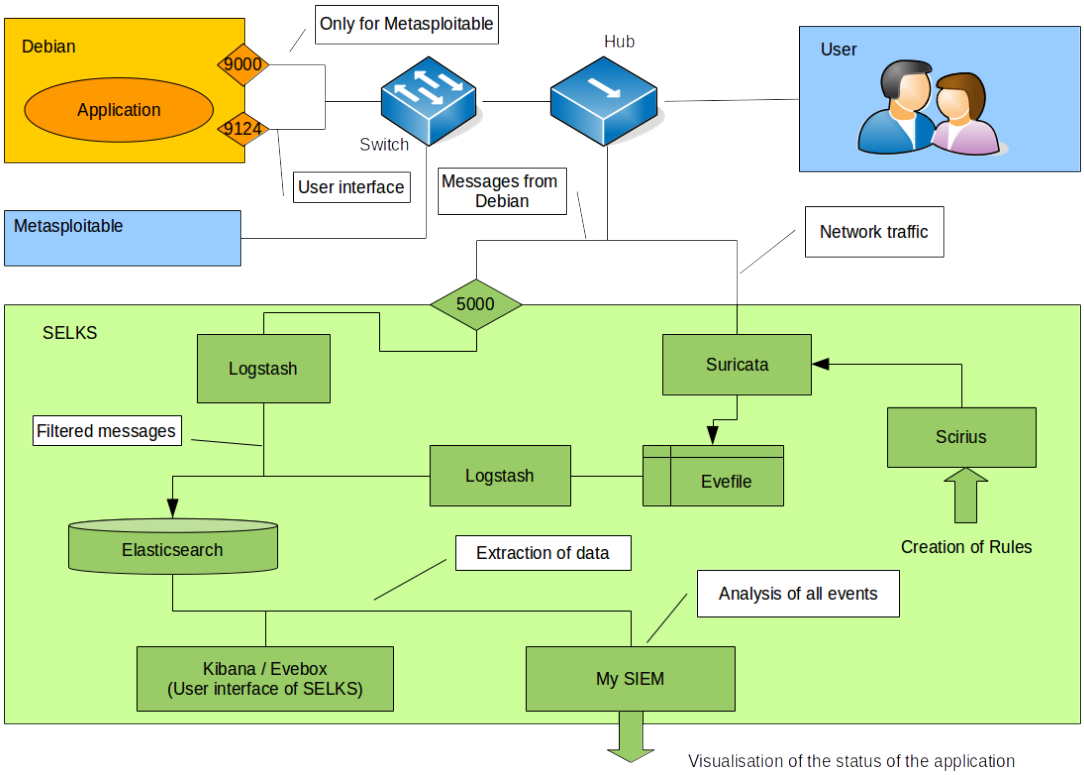

Figure 8.3: Summary of the situation

CHAPTER 9

Analysis of results

9.1 Results

In the system represented before, I represented the cyber defense. On the same system, Justin Bouroux achieved attacks and tried to penetrate the system. Of course, it was designed to have some security issue, but the most important for my system is the detection of its attacks. In fact, all system have security issue even the most secure[1], but if we can detect an intrusion we are able to ensure the service. If you want information about the attacks carried out, you can read the Justin's report.

9.2 Way of improve

[1] A minimum of «zero day»

Conclusion

After this study, I got to implement a system of detection. But because of the lack of time, I achieved a simple system, which is based on SELKS and on a SIEM implemented by me. This system is adapted to the application used for the tests but it needs some improvements to be adapted for other applications.

We can also notice that my system of detection can also be attacked which has not been envisaged in this study.

To conclude, I achieve a simple system of detection which need some improvements. However, this system prove that it is possible to retrieve all the states of an observed system. And if we could model this observed system, we would be able to extract all the possible states of the system via model checking. Then, we could compare this possible states with the current state of the system to detect attacks.

So it is a prove of contest of the possibility to use model checking to implement the security rules of an application.

Annex

APPENDIX **A**

SELKS

Définition A.1 : *SELKS*

SELKS is a free and open source Debian (with LXDE X-window manager) based IDS/IPS platform released under GPLv3 from Stamus Networks (https://www.stamus-networks.com/).[10]

The SELKS ISO is both Live and Installable ISO in one. Once installed it is ready to use out of the box solution.

SELKS is comprised of the following major components:

S Suricata IDPS - http://suricata-ids.org/

E Elasticsearch - http://www.elasticsearch.org/overview/

L Logstash - http://www.elasticsearch.org/overview/

K Kibana - http://www.elasticsearch.org/overview/

S Scirius - https://github.com/StamusNetworks/scirius

So SELKS is an OS which contains many softwares. Firstly there is Suricata which is the network analyzer which raise alerts. Alerts produce by Suricata are written in a EVE file. Then Logstash traduce this file into a JSON file readable by Elasticsearch. Elasticsearch is a data base. It keep all event and permit a navigation into many millions of events almost instantaneous. And to finish Kibana propose a User interface for this data base. The figure A.2 represents this architecture.[1]

You can see on figure A.3, an example of the interface of kibana which give an overview of alerts. This image was extract from the Twitter of Stamus Networks.

Figure A.1: screenshot of SELKS desktop

[1]SIEM (security information and event management) which are not developed here, are software to analyze and display event of every security tools.

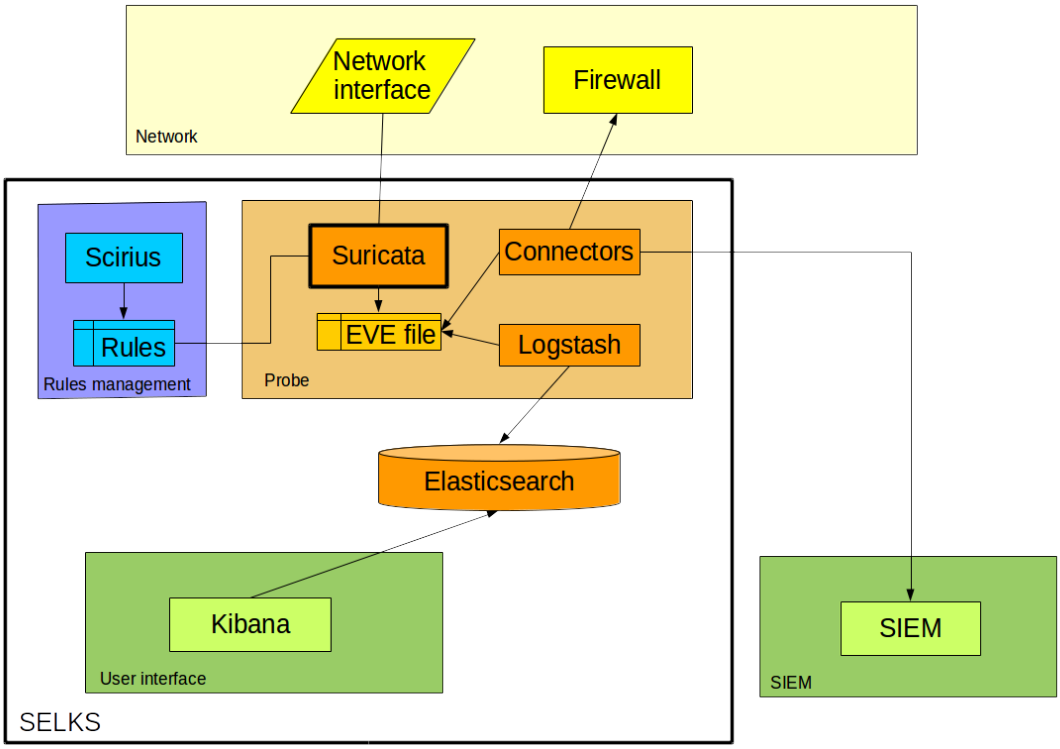

Figure A.2: Selks architecture

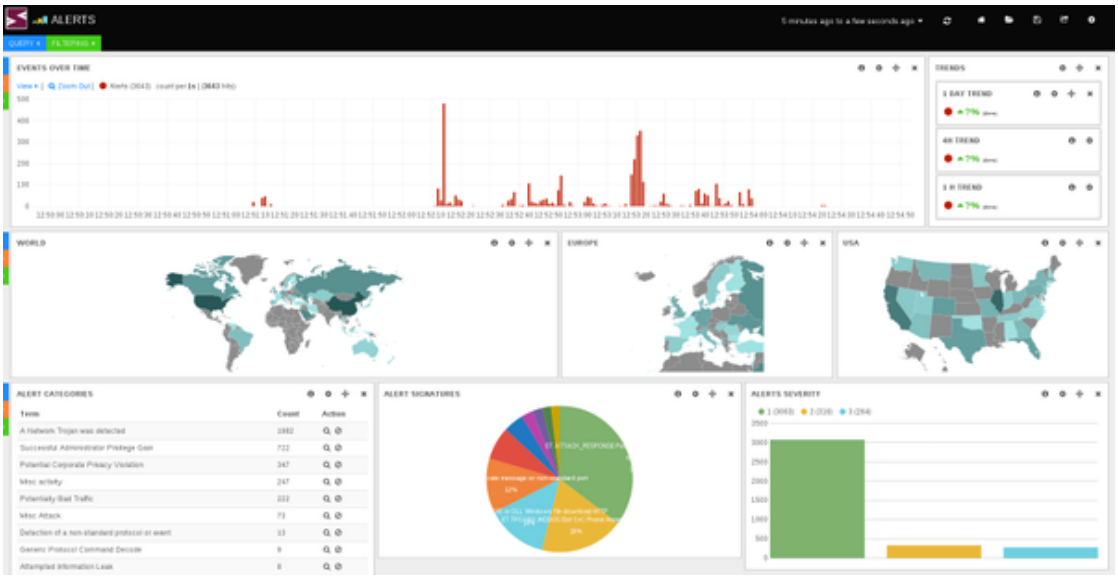

Figure A.3: Kibana interface

List of Figures

1.1 Hynesim logo 5
1.2 An example of a client Hynesim interface 6

4.1 Complex model in UML 12
4.2 Position of my project 13

5.1 Model of the application 14

6.1 Kanboard of this project 16
6.2 Gantt diagram of the project 17
6.3 Github server 18

7.1 Network infrastructure 20
7.2 Example of alerts 21

8.1 Data processing 23
8.2 My siem interface 24
8.3 Summary of the situation 24

A.1 screenshot of SELKS desktop 28
A.2 Selks architecture 29
A.3 Kibana interface 29

Bibliography

[1] 3ilson.org. Selks + esxi installation guide. Youtube, October 2016.

[2] Vangie Beal. intrusion detection system. Webopedia.

[3] Diateam. Hynesim. https://www.hynesim.org/.

[4] Solange Ghernaouti. *Sécurité informatique et réseaux*, volume 4, chapter 8.4. Dunod, 2013. At Ensta Bretagne code: I11 GHE.

[5] Frédéric Guillot. Kanboard. https://kanboard.net/.

[6] Jonathan Krier. Les systèmes de détection d'intrusions. Technical report, developpez.com, july 2006.

[7] Eric Leblond. Let's talk about selks. In *SSTIC conference*, June 2014.

[8] Eric Leblond. Suricata, dévoilez la face sécurité de votre réseau. *Gnu Linux Magazine France Hors-série*, 76:9, 2015.

[9] David Peterson. What is kanban. Technical report, KanbanBlog, 2009.

[10] StamusNetworks. Selks. `https://github.com/StamusNetworks/SELKS`, 2014.

[11] wikipedia. Security information and event management, 2017.