UNIVERSITY OF SANTIAGO DE COMPOSTELA

ESCOLA TÉCNICA SUPERIOR DE ENXEÑARÍA

Improvements in IDS: adding functionality to Wazuh

Autor:
Andrés Santiago Gómez Vidal

Directores:
Purificación Cariñena Amigo
Andrés Tarascó Acuña

Computer Engineering Degree

February 2019

Final degree project presented at the Escola Técnica Superior de Enxeñaría of the University of Santiago de Compostela to obtain the Degree in Computer Engineering

Ms. Purificación Cariñena Amigo, Professor Computing Science and Artificial Intelligence at the University of Santiago de Compostela and **Mr. Andrés Tarascó Acuña**, Managing Director at Tarlogic Security S.L.

STATE:

That the present report entitled *Improvements in IDS: adding functionality to Wazuh* written by **Andrés Santiago Gómez Vidal** in order to obtain the ECTS corresponding to the final degree project of the Computer Engineering degree was conducted under our direction in the department of Computer Science and Artificial Intelligence of the University of Santiago de Compostela.

For the purpose to be duly recorded, this document was signed in Santiago de Compostela on February TODO, 2019:

The director,	The codirector,	The student,
(Purificación Cariñena Amigo)	(Andrés Tarascó Acuña)	(Andrés Santiago Gómez Vidal)

Index

1 Introduction **1**
- 1.1 Motivation . . . 1
- 1.2 Objectives . . . 6
- 1.3 Structure of this document . . . 7

2 OSSEC and Wazuh **9**
- 2.1 Introduction . . . 9
- 2.2 Wazuh architecture . . . 11
- 2.3 Rules and decoders . . . 14

3 Requirements **17**
- 3.1 Use cases . . . 17
  - 3.1.1 Use cases actors . . . 18
  - 3.1.2 Use cases list . . . 18
- 3.2 Requirements analysis . . . 18
  - 3.2.1 Non functional requirements . . . 18
  - 3.2.2 Functional requirements . . . 18
  - 3.2.3 Domain requirements . . . 18

4 Project management **19**
- 4.1 Scope management . . . 19
  - 4.1.1 Description of the scope . . . 19
  - 4.1.2 Acceptation criteria . . . 20
  - 4.1.3 Increments . . . 20
  - 4.1.4 Products of the project . . . 21
  - 4.1.5 Exclusions . . . 21
  - 4.1.6 Restrictions . . . 22
- 4.2 Risk management . . . 23
  - 4.2.1 Risk metrics . . . 23
  - 4.2.2 Risk types . . . 24
  - 4.2.3 Risk identification . . . 24
  - 4.2.4 Risk analysis and planning . . . 25
  - 4.2.5 Risk supervision . . . 37

4.3 Time management 37
4.3.1 Metodology 37
4.3.2 WBS 37
4.3.3 Initial planning 40
4.3.4 Real planning 50

5 Technologies and tools **55**
5.1 Development technologies and tools 55
5.2 Pentesting technologies and tools 55
5.3 Documentation technologies and tools 55
5.4 Other technologies and tools 55

A Glossary **57**

B Bibliography **59**

List of Figures

1.1 Comparison by attributes of the most important ICSs[6] 5

2.1 The different parts of Wazuh[9] 9
2.2 Singlehost architecture 12
2.3 Distributed architecture 12
2.4 Communications and data flow 13
2.5 Filename structure of a log directory 14
2.6 Portion of the ruleset used by Wazuh[19] 15
2.7 Example of output for ossec-logtest 16

4.1 Planning simplification 41
4.2 "Beginning of the project" planning 42
4.3 "Increment 1: Common attacks in Windows Server" planning . . 43
4.4 "Increment 2: Use of data from Sysmon" planning 44
4.5 "Increment 3: Detection/action against ransomware" planning . . 45
4.6 "Increment 4: Adapt Wazuh configuration to typical requirements from enterprises" planning 46
4.7 "Increment 5: Explore solutions in problems with GPDR" planning 47
4.8 "Increment 6: Additional detection for GNU/Linux" planning . . 48
4.9 "Increment 7: VirusTotal integration" planning 49
4.10 "Closing of the project" planning 50
4.11 Planning simplification 51
4.12 "Beginning of the project" planning 52
4.13 "Updating tools of the project" planning 53
4.14 "Increment 1: Common attacks in Windows Server" planning . . 54

List of Tables

1.1 Simplification of the data flow 2

4.1 Probability classification of risks 23
4.2 Impact classification of risks 23
4.3 Method of calculation of exposition based on probability and impact 23
4.4 List of the risks of the project 24

Chapter 1

Introduction

This project was made in collaboration with the cybersecurity company Tarlogic SL, even though I am not a member of Tarlogic and have never worked with them in the past. It is key to note my lack of experience in cybersecurity (on a professional level) because is the reason for the bad planning estimations and the limit of the scope. Furthermore in this project there are no absolute constraints or objectives, as it was suggested as a case between investigation (with some coding) and cybersecurity auditing, so the scope can be reduced if the time remaining is too short.

1.1 Motivation

Cybersecurity nowadays is very complex and there are many sub-fields and expert tools and it could be argued that is impossible to guarantee that any system is totally safe. In this project to decide the technologies and tools to use we put ourselves in the shoes of an administrator of an enterprise system that wants to improve the security by detecting intrusions for the servers he works on.

Cybersecurity measures can be applied in multiple layers of the system, each with different tools, objectives, advantages and cost. In general the security of a system has the next parts:

1. **Firewall**: Control the inbound/outbound connections, on the **network layer**. In our case its objective is to reduce the amount of inbound connections, reducing the chance of intrusion.

2. **IPS**: Intrusion Prevention System to minimize the chance of intrusions, on the **network and host layers**. Provides active protection by actions.

3. **IDS**: Intrusion Detection System to mitigate the damage of intrusions, on the **network and host layers**. Provides passive protection by alerts.

The next table shows a **simplified** flow on how the information is processed by the security layers and methods. For example an IDS can monitor the network connections, scanning the whole packet (header and payload) and filing a report if needed, but has worse performance than a firewall because they only scan the header of the packet and just opt to reject them[1].

Table 1.1: Simplification of the data flow

Layer	Network	Network and Host	
Method	Firewall	IPS	IDS
Measures	Prevent	Prevent	Mitigate

Direction of the data flow $\longrightarrow$

We focus on IDS because we are more interested about host detection. Also IDS is less explored than IPS or Firewall and due to the advance in gathering and processing of data in the last years IDS has become much more viable and reliable.

IDSs are different from antivirus or antimalware because the first are systems **specialized** in detection and the latter usually focus on prevention, however prevention and detection are often meshed together because both are deeply related. There are some cases where a system specialized in detection offers some kind of mitigation functionality or a steam specialized in prevention offers some kind of detection functionality.

It is important to note that in cybersecurity the trend is for the attack to be created first and later some kind of measures, not necessarily by the same teams as they usually are specialized in each role. This means that defensive security that requires manual intervention always lags behind.
Nowadays there are lots of different attacks, so many that their detection could be almost impossible one by one, but most of them can be detected because they share patterns. If we can determine the patterns of an attack and code a way to detect them we can detect the threat. Due to all this some times is easier to detect the attack and take measures after the intrusion has taken place.

IDS work by analysing the key information available (programs, logs, network, etc) to determine if there has been an intrusion in the system. The details of the

process vary with each IDS but in general they work like an expert system:

- The source of the data is the system.
- The alerts are set by certain rules when they match.
- Rules do not need to throw an alert and there can be dependencies, allowing a stateful approach and complex analysis without false positives (the main annoyance of IDSs).

There are two types of IDS depending of the detection mechanism:

- Signature based: The IDS looks for specific data (signature), for example a string. This is often an efficient solution to known attacks, but is fundamentally useless against unknown attacks (attacks without a signature in the IDS database).
- Behaviour analysis: After a training period the IDS can detect when an event is rare (by probability) and correlate these suspicious occurrences to an intrusion.

In our case we take interest in the signature approach because is much more used and behavior analysis is more fit for network than host.

OSSEC is an HIDS (Host-based IDS) solution with detection based on rules and decoders. Both rules and decoders can be defined with numerous options and support dependencies and regular expressions.

- The decoders format the data for the rules.
- The rules determine there is a threat if the conditions are met.

OSSEC stands for **O**pen **S**ource HIDS **SEC**urity and is interesting for this project because the next qualities[2][3]:

- **Widely Used**: OSSEC is a growing project, with more than 5,000 downloads per month on average. It is being used by ISPs, universities, governments and even large corporate data centers as their main HIDS solution. In addition to being deployed as an HIDS, it is commonly used strictly as a log analysis tool, monitoring and analyzing firewalls, IDSs, web servers and authentication logs.

- **Scalable**: Because it is an HIDS and it uses **agents**. Each monitored host can either install the agent or use an agentless agent[4][5]. Agentless agents are processes initiated from the OSSEC manager, which gather information from remote systems, and use any RPC method (e.g. ssh, snmp rdp, wmi).

- **Multi-platform**: GNU/Linux, Windows, Mac OS and Solaris. This is important because most professional services are on GNU/Linux or Windows, but it is important to note that rules can only work in one operating system.

- **Free**: OSSEC is a free software and will remain so in the future; you can redistribute it and/or modify it under the terms of the GNU General Public License (version 2) as published by the FSF – Free Software Foundation.

- **Open source**: The code is open, so you can read, contribute and debug it all you want.

- **Rootkits detection**: This type of malware usually replaces or changes existing operating system components in order to alter the behavior of the system. Rootkits can hide other processes, files or network connections like itself.

- **File integrity monitoring**: To detect access or changes to sensitive data.

There are lots of alternatives to OSSEC for the scenario of a system administrator that wants to reinforce the security of the systems he is responsible for. There are free of charge and paid solutions and they do not need to be pure IDS as often they come in a full approach. For example the next table shows a comparison of the most important ICSs (Industrial Control Systems):

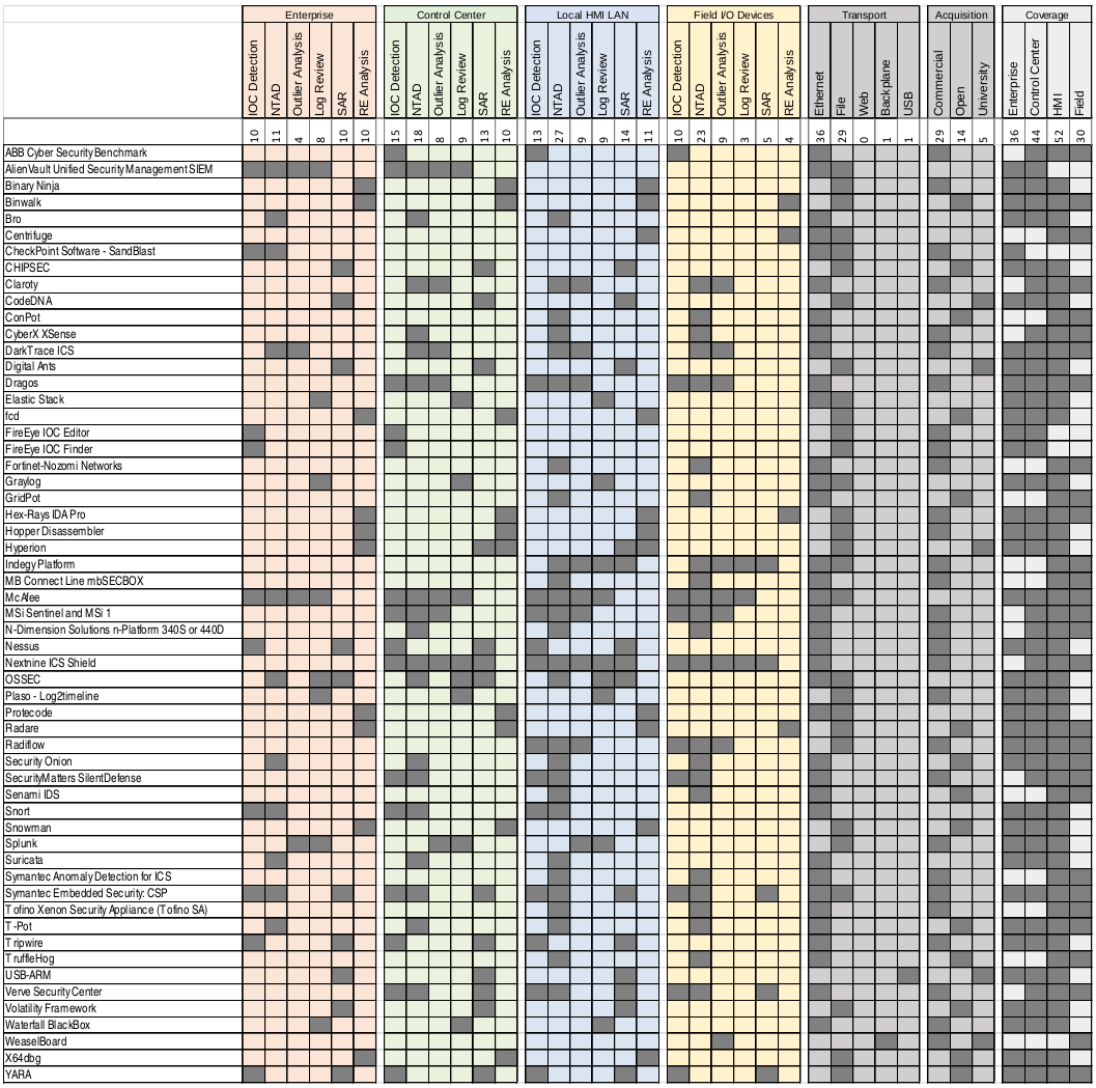


Figure 1.1: Comparison by attributes of the most important ICSs[6]

One of the problems of a comparison in a table like this is that it fails to show how much a tool excels or lacks in the features it shares with others, how easy it is to use and other factors that can decide the right tool. The most relevant alternative technologies to OSSEC for this project are[7]:

- Bro: Is an open source Network Intrusion Detection System (NIDS), so it does not have any other functionality and supports only Linux, FreeBSD, and Mac OS.
- Snort: Is the most popular open source IDS/IPS, but it is only for network and can be very expensive in processing power.

- Suricata: Another open source network security monitor with IDS and IPS capabilities and it provides hardware acceleration and multi-threading to improve the scanning speed at the cost of resources.

- Sagan: An open source HIDS, but only supports *nix operating systems (Linux, FreeBSD, OpenBSD, etc) and it lacks in features compared to OSSEC.

- YARA: Is not an IDS or IPS, it is just a tool that does pattern/string/signature matching, but it excels at it in performance, results and easiness to write the rules. YARA is being used widely in cybersecurity, for example by Avast, Kaspersky Lab, VirusTotal and McAfee Advanced Threat Defense[8]. We could build a system to use YARA to scan files but always combined with at least another tool, but we prefer to stick to a tested IDS.

In our case most of the attributes in the previous comparison do not matter and we chose OSSEC because the problems found on the alternatives and that OSSEC offers a reliable way to use an already done and thoroughly tested IDS and enhance it to our needs without much work. To even ease more this we will use Wazuh, a fork of OSSEC.

1.2 Objectives

The main objective is to improve intrusion detection in IDS. This can be accomplished in several ways:

- Adding or changing functionality of an already existing technology.

  Coding on core or additions.

  Configuration or input of the program.

- Develop a new technology or tools that result in a different detection system.

As mentioned before in this project we will use OSSEC through Wazuh to code rules and decoders, without the need to change any code of the program itself, which means this project can focus directly on detection without the need to create a detection system. Of course if in later stages of the project it would to be found that is convenient to modify the detection system itself it could be considered depending on the importance, the progress and the remaining time of the project.

1.3 Structure of this document

This document has TODO chapters:

- In **chapter 1**
- In **chapter 2**
- In **chapter 3**
- In **chapter 4**
- In **chapter 5**
- In **chapter 6**
- In **chapter 7**

Chapter 2

OSSEC and Wazuh

2.1 Introduction

Wazuh is different than base OSSEC in that it adds capabilities (a RESTFul API and rules and decoders) and is easier to install (it uses the ELK stack to gather and preprocess data, while OSSEC leaves that choice to the user).

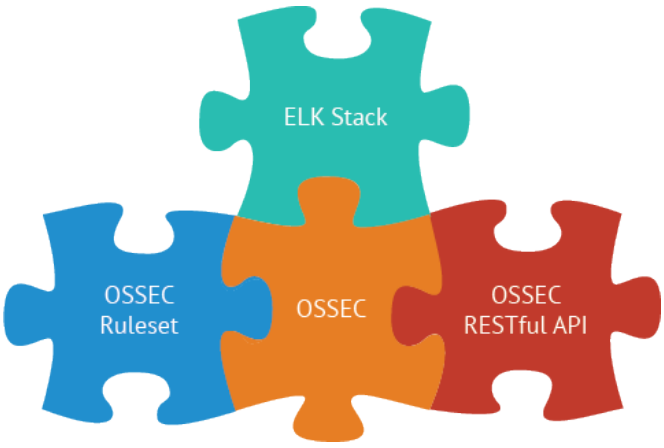


Figure 2.1: The different parts of Wazuh[9]

The most interesting qualities of Wazuh for this project are[10][11]:

- **Rootkits detection**: Rootkits are commonly used after an attack has suceeded to use the computer of the victim leaving no traces.
- **File integrity monitoring**: It can provide detection of intrusions by identifying changes in content, permissions, ownership, and attributes on the monitored files. It can be used to comply with GDPR (General Data Protection Regulation).

- **Scalability and multi-platform**: This means that the work on this project could really be used in real work environments.
- **Configuration management**: The configuration is managed by the Wazuh server (Wazuh manager) and the agents can be grouped, allowing custom, grupal or global gathering and detection for each agent.
- **Multiple sources of data**: The scanned data can be from logs, output of commands or databases.
- **Active response**: An automated remediation to security violations and threats, to mitigate more the possible damage. For example to stop the Internet connection to isolate a compromised system.
- **Improved ruleset**: This reduces the workload of this project, as it can serve as guidance and complement some of the rules and decoders that this project intends to create or modify.
- **Open source, free and easy to contribute to**: This is optional but nice, as it offers a chance to an unexperienced student to contribute in a real and useful project. The project is hosted on Github and Google Groups. In this project the contribution would be to the ruleset[12] and not to the core of Wazuh[13] or the documentation[14].

The RESTFul API interacts using OSSEC commands and would be interesting if this project were related to a tool issuing queries to Wazuh, but this is not the case. Anyway it is still something valuable to have as these kind of tools are very common nowadays.

Wazuh provides support and integration with multiple important tools and technologies:

- Docker container for OSSEC: An ossec-server image with the ability to separate the ossec configuration/data from the container.
- Puppet and Ansible: For massive deployment. This can be very helpful to setup a big environment mostly because even being no need to put configuration files in the agents for Wazuh often is necessary to configure other things and the process of registering agents can be tedious manually.
- Network IDS integration: Gives the option to use OwlH and integrate Suricata and Bro to generate alerts in Wazuh.
- VirusTotal: A free virus, malware and URL online scanning service that combines more than 40 antivirus solutions.

- OSQuery: Osquery can be used to expose an operating system as a high-performance relational database. This allows you to write SQL-based queries to explore operating system data.

The use of these depends on the scenario and the only one we take interest in is VirusTotal, because it can work as a secondary detection method for the most critical or complicated cases.

2.2 Wazuh architecture

A basic Wazuh setup has the next components[15]:

- Wazuh server: Runs the Wazuh manager, API and Filebeat (Filebeat is only necessary in distributed architecture). It collects and analyzes data from deployed agents.
- ELK stack: It reads, parses, indexes, and stores alert data generated by the Wazuh server. The ELK stack is flexible, highly configurable and very used in big data.
- Wazuh agent: Runs on the monitored host, collecting system log and configuration data and detecting intrusions and anomalies. It talks with the Wazuh server to which it forwards collected data for further analysis.

The main difference with the architecture of OSSEC is the ELK stack, because OSSEC leaves the choice of tools to the user. ELK stands for the combination of:

- Elasticsearch: Gets the data and allows search queries and analysis.
- Logstash: Transforms the data to the desired format. This step can make alike data from different log and output formats, trivializing the decoders work.
- Kibana: Shows the data in a web browser, with graphs and options like grouping and time interval. This is often easier than to write commands to scan the OSSEC log in the Wazuh server, as the data of interest tends to stay the same.

There are two possible architectures for this setup: having the ELK stack in the same machine that the Wazuh server (singlehost) or in a separated one (distributed). Each has advantages and disadvantages and in this project we will use

the singlehost because in our case there are no constraints and is easier to set up and is more efficient.

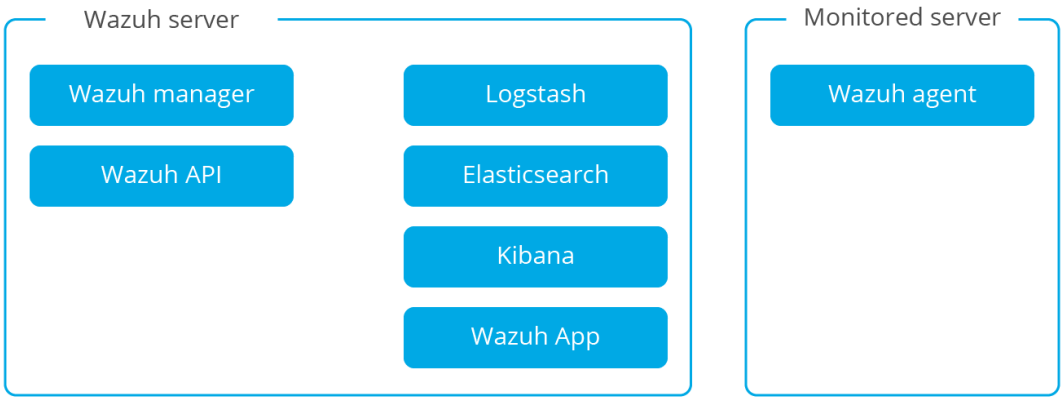


Figure 2.2: Singlehost architecture

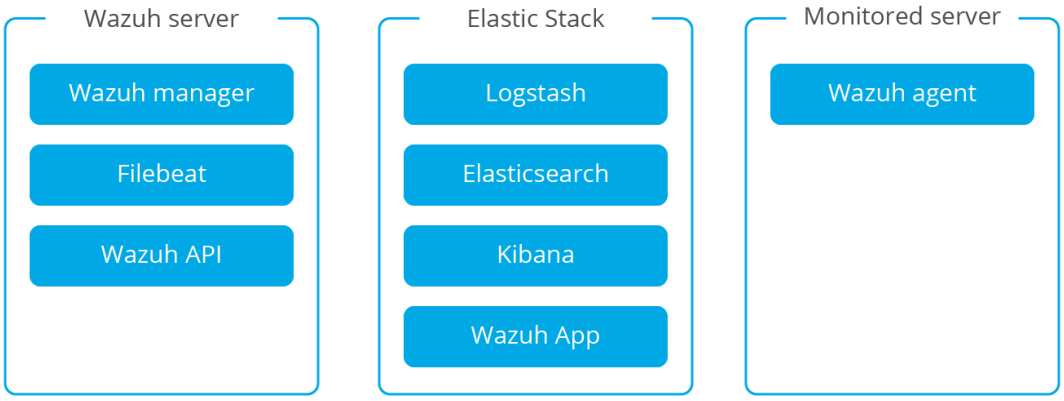


Figure 2.3: Distributed architecture

To understand better the communications and data flow in Wazuh we will now get into more detail on the process[16][17].

Wazuh agents use the OSSEC message protocol to send collected events to the Wazuh server over port 1514 (UDP or TCP). The Wazuh server then decodes and rule-checks the received events with the analysis engine. Events that trip a rule are augmented with alert data such as rule id and rule name. The Wazuh message protocol uses a 192-bit Blowfish encryption with a full 16-round implementation, or AES encryption with 128 bits per block and 256-bit keys.

Logstash formats the incoming data and optionally enriches it with GeoIP information before sending it to Elasticsearch (port 9200/TCP). Once the data is indexed into Elasticsearch, Kibana (port 5601/TCP) is used to mine and visualize the information.
The Wazuh App runs inside Kibana constantly querying the RESTful API (port 55000/TCP on the Wazuh manager) in order to display configuration and status related information of the server and agents, as well to restart agents when desired. This communication is encrypted with TLS and authenticated with username and password.

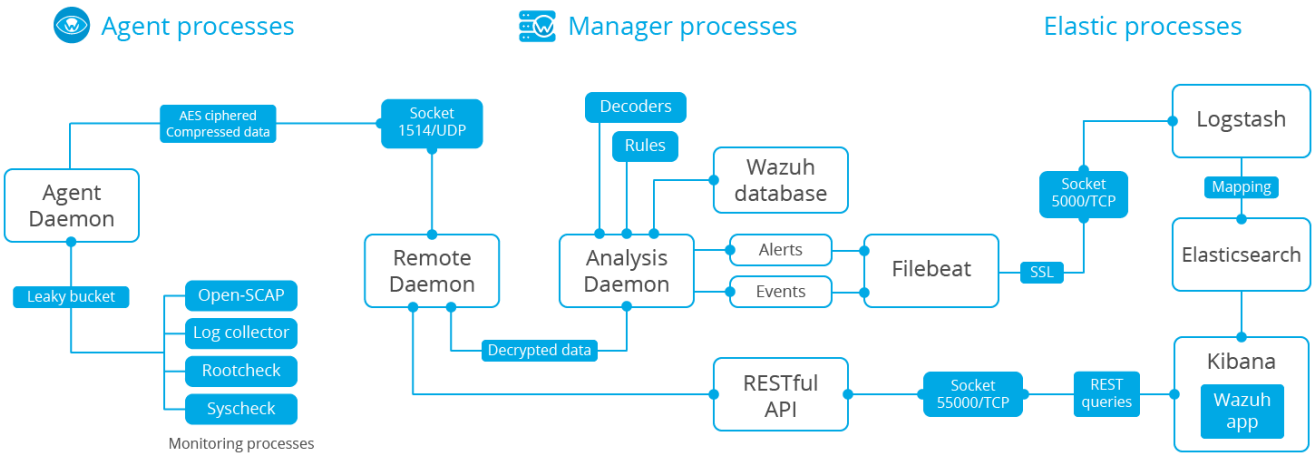


Figure 2.4: Communications and data flow

Both alerts and non-alert events are stored in files on the Wazuh server in addition to being sent to Elasticsearch. These files can be written in JSON format and/or in plain text format (.log, with no decoded fields but more compact). These files are daily compressed and signed using MD5 and SHA1 checksums. The directory and filename structure is as follows:

```
root@wazuh-server:/var/ossec/logs/archives/2017/Jan# ls -l
total 176
-rw-r----- 1 ossec ossec 234350 Jan  2 00:00 ossec-archive-01.json.gz
-rw-r----- 1 ossec ossec    350 Jan  2 00:00 ossec-archive-01.json.sum
-rw-r----- 1 ossec ossec 176221 Jan  2 00:00 ossec-archive-01.log.gz
-rw-r----- 1 ossec ossec    346 Jan  2 00:00 ossec-archive-01.log.sum
-rw-r----- 1 ossec ossec 224320 Jan  2 00:00 ossec-archive-02.json.gz
-rw-r----- 1 ossec ossec    350 Jan  2 00:00 ossec-archive-02.json.sum
-rw-r----- 1 ossec ossec 151642 Jan  2 00:00 ossec-archive-02.log.gz
-rw-r----- 1 ossec ossec    346 Jan  2 00:00 ossec-archive-02.log.sum
-rw-r----- 1 ossec ossec 315251 Jan  2 00:00 ossec-archive-03.json.gz
-rw-r----- 1 ossec ossec    350 Jan  2 00:00 ossec-archive-03.json.sum
-rw-r----- 1 ossec ossec 156296 Jan  2 00:00 ossec-archive-03.log.gz
-rw-r----- 1 ossec ossec    346 Jan  2 00:00 ossec-archive-03.log.sum
```

Figure 2.5: Filename structure of a log directory

2.3 Rules and decoders

They constitute the main part of this project and they can be used to detect application or system errors, misconfigurations, attempted and/or successful malicious activities, policy violations and a variety of other security and operational issues[10]. Wazuh is quite helpful with the features and documentation of the ruleset and in this project the already existing rules and decoders were a great help as examples.

Rules can be added in */var/ossec/etc/rules/* and decoders in */var/ossec/etc/decoders/* without any issue, but to change the already existing ones in */var/ossec/ruleset/rules/* or */var/ossec/ruleset/decoders/* is a bad idea because the next changes in those files from updates would overwrite them. The solution is to copy the code (actually only the id is needed) of the existing item to the folder where we can add new ones, make the desired changes add *overwrite="yes"*[18].

As mentioned before Wazuh adds its own ruleset over the one provided by the OSSEC project. The next table shows about 20% of the combined ruleset that Wazuh uses, where "Out of the box" means that the source was the OSSEC project.

OSSEC Ruleset		
Rule	**Description**	**Source**
amazon_rules	Amazon main rules.	Created by Wazuh
amazon-ec2_rules	Amazon Virtual Private Cloud (Amazon VPC) lets you provision a logically isolated section of the Amazon Web Services (AWS) Cloud where you can launch AWS resources in a virtual network that you define.	Created by Wazuh
amazon-iam_rules	AWS Identity and Access Management (IAM) is a web service that helps you securely control access to AWS resources for your users. You use IAM to control who can use your AWS resources (authentication) and what resources they can use and in what ways (authorization).	Created by Wazuh
apache_rules	Apache is the world's most used web server software.	Out of the box
apparmor_rules	AppArmor is a Linux kernel security module that allows the system administrator to restrict programs's capabilities with per-program profiles.	Out of the box
arpwatch_rules	ARPWatch is a computer software tool for monitoring Address Resolution Protocol traffic on a computer network.	Out of the box
asterisk_rules	Asterisk is a software implementation of a telephone private branch exchange (PBX).	Out of the box
attack_rules	Signatures of different attacks detected by OSSEC	Created by Wazuh
auditd_rules	The Linux Audit system provides a way to track security-relevant information on your system. Based on pre-configured rules, Audit generates log entries to record as much information about the events that are happening on your system as possible.	Created by Wazuh
cimserver_rules	Compaq Insight Manager Server	Out of the box
cisco-estreamer_rules	The FireSIGHT System Event Streamer (eStreamer) uses a message-oriented protocol to stream events and host profile information to the client application.	Created by Wazuh
cisco-ios_rules	Cisco IOS is a software used on most Cisco Systems routers and current Cisco network switches.	Out of the box
clam_av_rules	Clam AntiVirus (ClamAV) is a free and open-source, cross-platform antivirus software tool-kit able to detect many types of malicious software.	Out of the box
courier_rules	IMAP/POP3 server	Out of the box
docker_rules	Docker is an open-source project that automates the deployment of applications inside software containers.	Created by Wazuh
dovecot_rules	Dovecot is an open-source IMAP and POP3 server for Linux/UNIX-like systems, written primarily with security in mind.	Out of the box
dropbear_rules	Dropbear is a software package that provides a Secure Shell-compatible server and client. It is designed as a replacement for standard OpenSSH for environments with low memory and processor resources, such as embedded systems.	Out of the box
firewall_rules	FirewallD provides a dynamically managed firewall with support for network/firewall zones to define the trust level of network connections or interfaces. Default firewall management tool RHEL and Fedora.	Out of the box
firewalld_rules	Firewall events detected by OSSEC	Out of the box

Figure 2.6: Portion of the ruleset used by Wazuh[19]

Wazuh provides a way to manually test how an event is decoded and if an alert is generated with the tool */var/ossec/bin/ossec-logtest*[20], which is very useful for debugging. To use it you only need to introduce the data as it would be in the wild (in the logfile or command output). This command provides the option "-v" that shows which rules are tried and which trigger an alert.

For example for this input:

```
Mar  8 22:39:13 ip-10-0-0-10 sshd[2742]: Accepted publickey for root from 73.189.131.56 port 57516
```

We get the next output:

```
$ /var/ossec/bin/ossec-logtest

Mar  8 22:39:13 ip-10-0-0-10 sshd[2742]: Accepted publickey for root from 73.189.131.56 port 57516

**Phase 1: Completed pre-decoding.
       full event: 'Mar  8 22:39:13 ip-10-0-0-10 sshd[2742]: Accepted publickey for root from 73.189.131.56 port 57516'
       hostname: 'ip-10-0-0-10'
       program_name: 'sshd'
       log: 'Accepted publickey for root from 73.189.131.56 port 57516'

**Phase 2: Completed decoding.
       decoder: 'sshd'
       dstuser: 'root'
       srcip: '73.189.131.56'

**Phase 3: Completed filtering (rules).
       Rule id: '5715'
       Level: '3'
       Description: 'sshd: authentication success.'
**Alert to be generated.
```

Figure 2.7: Example of output for ossec-logtest

After version 3.0.0 (we are currently in 3.7) Wazuh incorporates an integrated decoder for JSON logs enabling the extraction of data from any source in this format. This can be very useful in many situations, for example trivializing the generation of alerts for Suricata (without the need for a decoder just for Suricata)[21].

Another interesting feature is to check if a field extracted during the decoding phase is in a CDB list (constant database). The main use case of this feature is to create a white/black list of users, IPs or domain names.[22].

Chapter 3

Requirements

The requirement specification is a full description of the software the project is to develop.
PMBOK[23] states that requirements are conditions or capabilities that a product must meet to satisfy the contract. The requirements expose the needs of the client, which have to be accomplished to finish the project successfully. In this project the requirements will be fullfilled in multiple stages along the project.
Note that the client in this case is Tarlogic even if the product is a contribution to an open source project.

This specification contains:

- **Use cases**: Functionalities that the software will provide.
- **Requirements**: Depending of their type they can describe features, data, relations, properties or any details necessary to explain the system without ambiguity, in a way it can be easily understood.

In this project the functional requirements are not included because they can be considered a redundant version of the use cases, because both describe the same functionalities. Uses cases were chosen over functional requirements because they were considered to be easier to understand and have greater detail. If this project had the need of a very complex requirement specification it would be interesting to have both, as each could help to understand the other better, but in this project the specification should be quite simple.

3.1 Use cases

A use case is a description of all the ways an end-user wants to "use" a system. These "uses" are like requests of the system, and use cases describe what that

system does in response to such requests. In other words, use cases describe the conversation between a system and its user(s), known as actors. Although the system is usually automated (such as an Order system), use cases also apply to equipment, devices, or business processes.[24]

3.1.1 Use cases actors

The actors are entities external to the system that interact with it. They can be other systems, persons or even time.

3.1.2 Use cases list

3.2 Requirements analysis

3.2.1 Non functional requirements

3.2.2 Functional requirements

As mentioned before these are omited because of the redundancy with use cases.

3.2.3 Domain requirements

Chapter 4

Project management

A project is temporary in that it has a defined beginning and end in time, and therefore defined scope and resources. And a project is unique in that it is not a routine operation, but a specific set of operations designed to accomplish a singular goal.
Project management, is the application of knowledge, skills, tools, and techniques to project activities to meet the project requirements[25]. It is important to note that the actions on each area of the project may affect other areas, increasing the difficulty of the management.

4.1 Scope management

The management of the scope of the project has the necessary processes to guarantee that the objectives are met. The scope management allows the project to start focused in what really matters, not losing time in irrelevant details or desirable additions that we can not implementent, by identifying and describing the necessary tasks.

4.1.1 Description of the scope

This project tries to improve the detection of intrusions on the already existing HIDS Wazuh. This kind of objective can be accomplished by very different approaches, because the software to work on can be used in many scenarios, is very related to other software and is in active development.
Though some increments can be considered difficult due to the amount of new technologies and tools there should be no problem to met the basic objectives because we have the freedom to adapt the scope at any time and there is more than enough time for the essential parts of the project.

4.1.2 Acceptation criteria

In order to the product to be accepted the essential requeriments need to have been accomplished before the time limit of the project.

4.1.3 Increments

The essential increments to the project are:

- Increment 1: Common attacks in Windows Server.
- Increment 2: Use of data from Sysmon. By itself is not really a must have, but it could make a different for certain attacks and the more data the better.
- Increment 3: Detection/action against ransomware.

These were chosen because we think Wazuh needs tangible security measures against common threats more than anything else at the moment. In other words, we reckon these points are were Wazuh lacks the most right now.

The rest of the increments are considered optional and can be removed if there is not enough time left. The order is based on the estimation of the relevance of the increment for Wazuh and our project. This means for example having in mind the time estimation for the increment without stretching.

- Increment 4: Adapt Wazuh configuration to typical requirements from enterprises. This is considered a very important increment because we think it could be a selling point for some enterprises, that probably do not want the same level of security for all their computers and the time to set it up (or at least from scratch). There is a chance that something like this already exists, in which case the investigation at the start of the increment should find it.
- Increment 5: Explore solutions in problems with GPDR. Tarlogic stated that they would like to have this increment specifically.
- Increment 6: Additional detection for GNU/Linux. This could have more or less the same impact as increment 1 for some clients, but Tarlogic was more interested in Windows and Wazuh seems to be more oriented towards GNU/Linux.
- Increment 7: VirusTotal integration. The problem of this increment is that VirusTotal's public API key has more limited features and has a 4 requests/minute limitation[26]. We asume the use of a public API key

because it would fit the profile of a client using Wazuh, which has no charge. Also the exploration on this increment could not really be considered more than a patch to Wazuh, without really improving it, but still it would be an effective workaround for the problems we can not solve right now with only Wazuh.

4.1.4 Products of the project

At the end of the project the next elements will be delivered:

-
-
-

4.1.5 Exclusions

As in any project of this kind we had to leave some ideas behind.

For example an interesting way to to take advantage from IDS is to set up a honeypot (a false server just to be compromised) and learn from the intrusions suffered, improving the defenses (firewall, IPS and IDS) for the real servers. There are some honeypot implementations that automate (for example with machine learning) the generation of rules for certain IDSs, but is not yet a trend because there are problems[27][28][29][30]:

- Experienced attackers have learned to avoid honeypots, because they are easy to identify due to the low security they have.
- Is not trivial to automate correctly the defense based on the information of the system, because its state can be very complex (for example due to more than one attack at the same time).

This automation would be a great solution to the need to update manually the rules and depending of the case it could even protect against day-zero vulnerabilities. Despite being interesting this was not even included as a possible increment because the complexity of the task. If this were included probably it would not have ended well because is something that even experts in cybersecurity have some trouble with at the moment.

There is always the risk of the intrusion disabling the security of the system.

This is more or less the same problem that cybersecurity has in any scenario and there is no way to guarantee that it will not happen. In this case the attacker would have to somehow not be detected or cut the IDS before it sends the alert but in a way that is not suspicious (for example shutting it down completely would be obvious for a central manager). So our approach is to trust the IDS and work on improving the detection of known attacks instead of the worst case scenario. If there was enough time we could have considered finding a solution for this problem.

Exploring a HIDS with behaviour analysis was also considered but rejected because it is more fit for a network approach. Still is a shame there is no enough time to explore IDS based on behavior analysis, because their protection against much zero-day attacks.

We focus on a host approach, leaving aside most of the detection capabilities for the network. This means less detection, a lower detection rate, less options to improve the detection process and later and worse performance in the analysis of network traffic. Having chosen to focus on HIDS the best way to have also a good NIDS process would be the use of a NIDS along with our HIDS.
Wazuh offers this kind of integration with Bro and Suricata and probably it would be possible to extend it to other NIDSs like Snort, but yet again we had to choose and this was not a priority at the moment.

4.1.6 Restrictions

Leaving aside the time constraint of about 417 hours, the two main factors to decide what improvements to choose for this project are a student without experience in proffesional cybersecurity and that we want some kind of inmediate results from this project. This is why while we could just have a pure research project (for example machine learning with IDS) instead we opted for a more traditional and safe approach. This is the reason why most of the increments were optional (due to the high probability of initial scope being too ambitious) but the first increments are considered vital to the project.

A minor restriction is to deliver correctly all the products of the project before the presentation date.

4.2 Risk management

4.2.1 Risk metrics

Chances of the risk happening	Probability
≥80%	High
Between 30% and 80%	Medium
≤30%	Low

Table 4.1: Probability classification of risks

Resource in Place / Effort / Cost	Impact
≥20%	High
Between 10% and 20%	Medium
≤10%	Low

Table 4.2: Impact classification of risks

Exposition		Probability		
		High	**Medium**	**Low**
Impact	**High**	High	High	Medium
	Medium	High	Medium	Low
	Low	Medium	Low	Low

Table 4.3: Method of calculation of exposition based on probability and impact

4.2.2 Risk types

4.2.3 Risk identification

Identifier	Name
R-01	Optimist planning, "best case" (instead of a realistic "expected case")
R-02	Bad requirement specification
R-03	Design errors
R-04	Lack of key information from sources
R-05	Lack of feedback or support from the security consultants of Tarlogic
R-06	The learning curve of some technologies is larger than expected
R-07	The unexplained parts of the project take more time than expected
R-08	Can not access source material
R-09	Unexpected changes to any of the software used in the project
R-10	Loss of work
R-11	Wrong management of the project's configuration
R-12	A delay in one task leads to cascading delays in the dependent tasks
R-13	The student can not find a way to code the detection of a certain occurrence
R-14	The quality of the product is not enough
R-15	Sickness or overwork
R-16	Performance issues
R-17	Unnecessary work
R-18	Optional requirements delay the project
R-19	Unexpected personal events delay the project

Table 4.4: List of the risks of the project

4.2.4 Risk analysis and planning

Identifier	**R-001**
Name	Optimist planning, "best case" (instead of a realistic "expected case")
Description	An optimistic planning at the start of the project does not take into account problems or delays, and so it does not allocate time for them.
Negative effects	Could mean the failure of the project if the objectives can not be accomplished in the time left. Cascading delays, because the work done would not fit the planning.
Probability	Medium
Impact	High
Exposition	High
Indicator	There are 3 consecutive delays, after the beginning of the project.
Prevention: Avoid	Allocate a bit more time than initially expected for each task, in case something goes wrong.
Correction: Mitigate	Redo the planning. Reduce the scope of the project, leaving out initially planned optional increments.

Identifier	**R-002**
Name	Bad requirement specification
Description	The requirements specified at the beginning of the project are not specific enough, are not needed or there are new requirements after the beginning of the project.
Negative effects	Possible failure of the project if the objectives can not be accomplished in the time left. Wasted time, due to lack of comunication in the requirement specification.
Probability	High
Impact	High
Exposition	High
Indicator	There are 3 changes in the requirements specification.
Prevention: Mitigate	Confirm that all the requirements have been identified at the beginning of the project. Assure that there is no ambiguity in the requirement specification.
Correction: Mitigate	Redo the requirement specification. Rework of related requirements and work based on them, including the need to test the results. Redo the planning. Reduce the scope of the project.

Identifier	**R-003**
Name	Design errors
Description	A design is not enough or is incorrect. This can be found in later stages, when it is clear that the implementation based on the design would not satisfy the requirements.
Negative effects	Having to redesign and maybe redo the work based on the design. Minor delays.
Probability	Low
Impact	Medium
Exposition	Low
Indicator	There are 3 designs that need rework.
Prevention: Mitigate	Use design patterns if needed (this project should have very simple designs, so it is possible that there is no need to use them). Make the design as simple and modular as possible.
Correction: Mitigate	Redesign and probably change and test the work based on the design.

Identifier	**R-004**
Name	Lack of key information from sources
Description	Not having key information from articles, documentation or manuals.
Negative effects	Minor delays. Loss of quality. Added difficulty, even if the work is done in time. Maybe rework and test the functionality, even completely, to follow the desired procedure.
Probability	Medium
Impact	Medium
Exposition	Medium
Indicator	The duration of the study of the attack and the related tools takes 50% than expected.
Correction: Mitigate	Ask the security consultants of Tarlogic for specific information. Possibly the need to rework completely some functionality.

Identifier	**R-005**
Name	Lack of feedback or support from the security consultants of Tarlogic
Description	Because I do not know enough of some technical aspects of cibersecurity to solve all the problems in this by myself in time, Tarlogic has promised to help (in a tutoring way) if a problem arises. This help could be critical to solve or get around some of the most complex problems, which probably happen to be critical points, needing to be dealt with to continue working on that stage.
Negative effects	Cascading delays.
Probability	Medium
Impact	Medium
Exposition	Medium
Indicator	A simple technical question takes more than 2 working days to be answered or a complex question takes more than 7 working days.
Prevention: Mitigate	Ask in a clear way and with as many details as possible. Ask during work hours, to ensure they are available.
Correction: Mitigate	Redo planning and possibly change the scope.

Identifier	**R-006**
Name	The learning curve of some technologies is larger than expected
Description	This is a critical need because not having enough knowledge can result in an inefficient approach to accomplishing the objectives.
Negative effects	Loss of quality. The work is more complicated.
Probability	Medium
Impact	Medium
Exposition	Medium
Indicator	The duration of the study of the technologies takes 50% than expected.
Correction: Mitigate	Redo planning and possibly change the scope. Ask the security consultants of Tarlogic for specific help. Maybe the need to rework completely some functionality.

Identifier	**R-007**
Name	The unexplained parts of the project take more time than expected
Description	There is not enough specification on what a tasks implies or not enough planning. This means that a part of the project is not understood as it should, and the work done is not what was expected or is not enough, needing more time to finish.
Negative effects	Wasted time that should have been easy to avoid. Loss of quality. Could mean the failure of the project if the objectives can not be accomplished in the time left.
Probability	Low
Impact	High
Exposition	Medium
Indicator	A task takes 25% more time than expected and when the causes are investigated it is revealed that there were ambiguous descriptions or planning.
Prevention: Avoid	Try to detail every part enough, having no obvious ambiguity.
Correction: Mitigate	Possible need to redo the specifications. Redo planning and possibly change the scope. Maybe having to redo related work.

Identifier	**R-008**
Name	Can not access source material
Description	All or part of the source material can not be accessed, probably because the only host of the resource is down.
Negative effects	In some cases this could mean a delay in a critical task, delaying the whole project for an unknown period of time.
Probability	Low
Impact	Medium
Exposition	Low
Indicator	There have been at least 10 failed attempts to download the source material, at least 5 with a computer A in a network X and at least 5 with a computer B in a network Y.
Prevention: Avoid	When possible choose the source with the best uptime.
Correction: Mitigate	Redo planning and possibly change the scope. Possible need to cut out the part of the project that depends on this source. Maybe find another source or wait to the original source to be accessible again.

Identifier	**R-009**
Name	Unexpected changes to any of the software used in the project
Description	Changes to base software could affect this project directly or indirectly: programs could fail or not work as expected. This could mean any software changes, from simple syntax to API changes. Is possible that these changes would eliminate the need of planned or already done work. In a project that does not work in a bleeding edge environment, like this, this occurrence should be very rare and even if it were to happen it would have to interfere with the part of the software this project uses, which (as this is not bleeding edge) normally would be backwards compatible.
Negative effects	Minor delays. Unnecessary work.
Probability	Low
Impact	Low
Exposition	Low
Indicator	The software is not working as expected due to a change in another software version.
Prevention: Mitigate	When possible use software that follow good design guidelines and try to be backwards compatible. Be informed about the roadmap and future functionalities of these software projects.
Correction: Mitigate	Need to adapt the software to work as expected or remove the related functionalities.

Identifier	**R-010**
Name	Loss of work
Description	Due to a bad configuration management or something else, there is a loss of work related to this project.
Negative effects	Need to do again the work already done but lost. Depending of the time needed to recover the work, there could be minor or very big delays, planning, changes to the scope of the project and even its failure.
Probability	Low
Impact	High
Exposition	Medium
Indicator	The need to replicate already done work is greater than 30 minutes.
Prevention: Mitigate	Take snapshots of key status for each virtual machine. Automate backing up the data and store the copies both in a cloud storage service and in a local disk.
Correction: Mitigate	Recover the last backup available of the work. If needed work even outside schedule and in holidays.

Identifier	**R-011**
Name	Wrong management of the project's configuration
Description	The project's configuration is inefficient or lacks work. For example due to unclear changes or taking too long to commit changes.
Negative effects	Maybe the failure of the project if the objectives can not be accomplished in the time left. Possibly wrong baselines or identification of the configuration elements. It could be that it takes more time than expected to manage the project. The project suffer delays because the need to redo management work and/or planned tasks.
Probability	Medium
Impact	High
Exposition	High
Indicator	There are 3 delays because of the configuration of the project.
Prevention: Avoid	The configuration of the project should be just complex enough (whithout ambiguity, to ensure a proper management), but not too much complex (which would be hard to follow). Use of familiar and standard tools, like Git. Optionally use an easier to manage lifecycle. Study of the configuration management done in previous final degree projects, to get a proper idea of its scope and details.

Identifier	**R-012**
Name	A delay in one task leads to cascading delays in the dependent tasks
Description	A task gets delayed and one or more tasks depends on its completion to start, so they get delayed too.
Negative effects	Cascading delays.
Probability	Medium
Impact	Medium
Exposition	Medium
Indicator	At least 2 tasks are delayed, due to only one of them needing more time.
Prevention: Avoid	When planning, avoid task dependencies whenever possible. Optionally use a lifecycle based on increments.
Correction: Mitigate	Redo planning and possibly change the scope.

Identifier	**R-013**
Name	The student can not find a way to code the detection of a certain occurrence
Description	It could be that the knowledge of the student is too limited or the problem has too much logical or mathematical difficulty. Another possibility is that the event is impossible to detect with the current technologies. If so, this impossibility could be hard to assure too, due to the complexity of nowadays technology.
Negative effects	High difficulty to estimate the time needed to detect the event. Cascading delays.
Probability	Low
Impact	Low
Exposition	Low
Indicator	Finding a way to detect the occurrence takes 30% more time than planned.
Prevention: Mitigate	Have as much information on the problem as possible, the more detailed the better.
Correction: Mitigate	Ask the security consultants of Tarlogic for help. Demonstrate that it is possible to detect it.

Identifier	**R-014**
Name	The quality of the product is not enough
Description	The final result is does not comply the quality standard set for this project.
Negative effects	The incorporation to the official repository gets rejected. Redo planning and possibly change the scope. Analysis of the changes needed to improve the quality.
Probability	Low
Impact	High
Exposition	Medium
Indicator	Getting 10 suggestions to rework functionality.
Prevention: Avoid	Follow design patterns. Follow the design guidelines of the official repository when possible.
Correction: Mitigate	Need to redo and test work. Pass some kind of quality control.

Identifier	**R-015**
Name	Sickness or overwork
Description	The health of the student deteriorates to the point it affects the project.
Negative effects	Probably the quality of the project drops. Possibly delays, that could be hard to specify their limit. Analysis of the changes needed to improve the quality. In the worst case scenario the project can not continue and fails.
Probability	Medium
Impact	High
Exposition	High
Indicator	There is an unexpected delay because the functionality is not done but there has not been any important issues that could explain it but there is a clear deterioration of the student health.
Prevention: Avoid	Stay healthy by following a regular schedule for work and exercising, that includes multiple rest periods. Optionally maintain a diet.
Correction: Mitigate	Go to the doctor and follow any instructions to improve the recovery.

Identifier	**R-016**
Name	Performance issues
Description	The program is too heavy for the environment and takes too much resources, because there are not good enough optimizations or the problems are poorly approached.
Negative effects	Minor delays.
Probability	Low
Impact	Low
Exposition	Low
Indicator	The program takes 30% more resources that at the beginning of the project.
Prevention: Mitigate	If possible use efficient algorithms and check the efficiency after the testing is done for each increment.
Correction: Mitigate	Analysis of faster ways to solve the problem. Code and test a faster solution.

Identifier	**R-017**
Name	Unnecessary work
Description	Resources are wasted in work that latter is not used. This could happen because multiple reasons, like wrong assumptions or balancing of the remaining time of the project.
Negative effects	Minor delays.
Probability	Low
Impact	Low
Exposition	Low
Indicator	There is at least one functionality not necessary or useful for any requirement.
Prevention: Avoid	In the design stage make sure that everything is really needed.
Correction: Mitigate	Evaluate again if the work planned is really needed.

Identifier	**R-018**
Name	Optional requirements delay the project
Description	Optional requirements get too much time or are treated as vital.
Negative effects	The task related to these requirements get too much resources. Vital requirements get less resources, making the project loss value.
Probability	Low
Impact	Low
Exposition	Low
Indicator	There is at least one functionality from an optional requirement, when the project is behind schedule and there are vital requirements not yet accomplished.
Prevention: Avoid	The optional requirements are planned as optional: they are only done if there is enough time left.
Correction: Mitigate	Redo the planning.

Identifier	**R-019**
Name	Unexpected personal events delay the project
Description	There are unplanned occurrences that need considerable time from the student, for example family matters.
Negative effects	Time loss, resulting in a quality drop and possibly in a smaller scope. It can be hard to specify when the event will end, resulting in uncertainty and the failure of the project in the worst case scenario. Even more if is about a chronical or serious sickness from a family member. Vital requirements get less resources, making the project loss value.
Probability	Medium
Impact	High
Exposition	High
Indicator	The student stops to work on the project for more than 2 planned weeks, to attend personal matters.
Prevention: Avoid	Always be organized and try to predict time consuming events.
Correction: Mitigate	Redo the planning. Use personal time like holidays and weekends to work on the project to compensate. In extreme cases the project may be put on hold or even fail.

4.2.5 Risk supervision

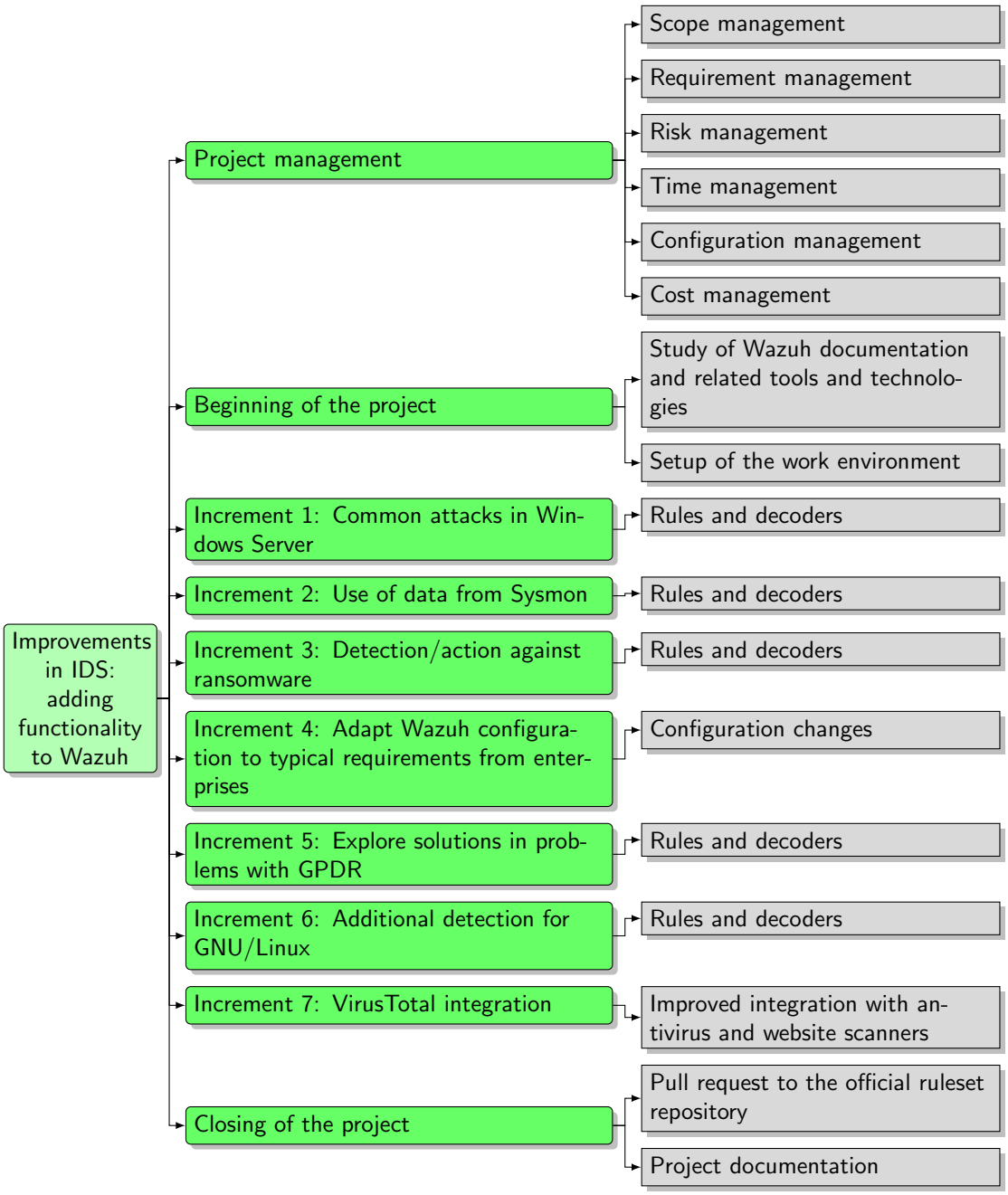
Identifier	**R-019**
Name	Unexpected personal events delay the project
Date of the beginning of the problem	05/12/2018
Date of the solution of the problem	10/02/2019
Actions	After a delay of 2 weeks it was clear that the student could not meet the original planning, or at least without rushing and suffering significant quality loss. The project was put on hold and the student notified the tutors, who agreed to the next deadline. The student kept working on the project (researching) from time to time.
New probability	Low (before was Medium)
New impact	High (same as before)
New exposition	Medium (before was High)
New prevention: Avoid	Another person takes responsibility of the family member, freeing the student.

4.3 Time management

4.3.1 Metodology

4.3.2 WBS

The Work Breakdown Structure is a decomposition of the project into smaller components (tasks).



WBS dictionary:

1. **Project management**

    a **Scope management**: Scope explanation, set the restrictions of the project and determine what is going to be turned in at the end of the project.

   b **Requirement management**: Analysis, requirement specification and probably a traceability matrix.

   c **Risk management**: Identification, analysis, classification, planning and supervision of risks.

   d **Time management**: Planning (initial and real), any planning changes and necessary measures.

   e **Configuration management**: Documentation on the management of changes and control version.

   f **Cost management**: Cost estimation (direct and indirect) of software, hardware and resources.

2. **Beginning of the project**

   a **Study of Wazuh documentation and related tools and technologies**: Is the base for multiple aspects of the project and if it is done correctly it can mean less hours in related work.

   b **Setup of the work environment**: Installation and basic configuration of the virtual machines of the project, like having a functional Wazuh environment.

3. **Increment 1**

   a **Rules and decoders**: The objective is to be able to detect common attacks in Windows Server (specifically 2016 and 2019), but it should be backwards compatible and depending on the difficulty it could be worth to ensure support for Windows 10 Pro too. This rules are the final product of this increment, which probably will need more time than any other increment, because its heavy study and testing.

4. **Increment 2**

   a **Rules and decoders**: It will need a preliminary study of Sysmon and the ways to use its data to improve detection in certain situations. It is possible that this increment will modify rules and decoders of the previous one.

5. **Increment 3**

   a **Rules and decoders**: This increment tries to produce rules and decoders to detect ransomware and launch alerts and maybe actions against the attack, like rollback to a previous backup or try to stop the attack from repeating in a short period of time.

6. **Increment 4**

   a **Configuration changes**: Adapt Wazuh to the typical requirements from enterprises. This means that an enterprise could choose from a set of templates, with different security profiles.

7. **Increment 5**

   a **Rules and decoders**: Most should be focused on detecting changes on the protected files. Part of this increment should be the investigation on normal problems of these technologies and recent innovations and solutions.

8. **Increment 6**

   a **Rules and decoders**: There would be preliminary study to do, but the increment should be about expanding the already done work in the field, probably focusing in services and security technologies like SELinux or AppArmor.

9. **Increment 7**

   a **Improved integration with antivirus and website scanners**: The idea is to improve the detection as much as possible with the help of VirusTotal malware scanners, which is updated consistently and so it would mean a consistently updated detection for a system with Wazuh without the need to write new rules and decoders. Obviously there is a difference in the scope and objectives of these technologies, which can be redundant, but this could be certainly interesting in some cases.

10. **Closing of the project**

    a **Pull request to the official ruleset repository**: There is a fundamental need to investigate the correct way to organize the the forked repository for a pull request to an official repository like this. In any case the status of the fork should be checked before and there should be a high amount of commits and use a different branch for each functionality, allowing an easier way to select what to admit or not in the official repository.

    b **Project documentation**: The memory and presentation of the project and whatever other documentation if necessary.

4.3.3 Initial planning

The tasks marked in red are essential to the project, meanwhile the ones marked in cyan are considered optional and only will be done if there is enough time

left. The tasks marked in yellow are normal, and they are used when there is no need to distinguish between essential and optional.

The next Gantt diagram shows the initial planning, from the draft proposal (31/10/2018) to the end of the project (20/02/2019).
Furthermore the last two weeks are marked with a grey overlay to mark that there are only about 17 weeks before the due date of this project (in February). This difference is because the estimation of the tasks was made by the student and so it is not reliable, which means that it could be optimistic or pessimist. Thus the need to either reduce tasks or have more that there were expected to fit.

Figure 4.1: Planning simplification

The rest of the Gantt diagrams are organized in days, for a more detailed planning.
It is important to note that these plannings could change during the project, either because controlled measures or any unexpected reason.
The order they are implemented could change too and that is the reason because these diagrams have not a set date for start and end, yet.

In other words, they could be described as the models for the final Gantt diagrams.

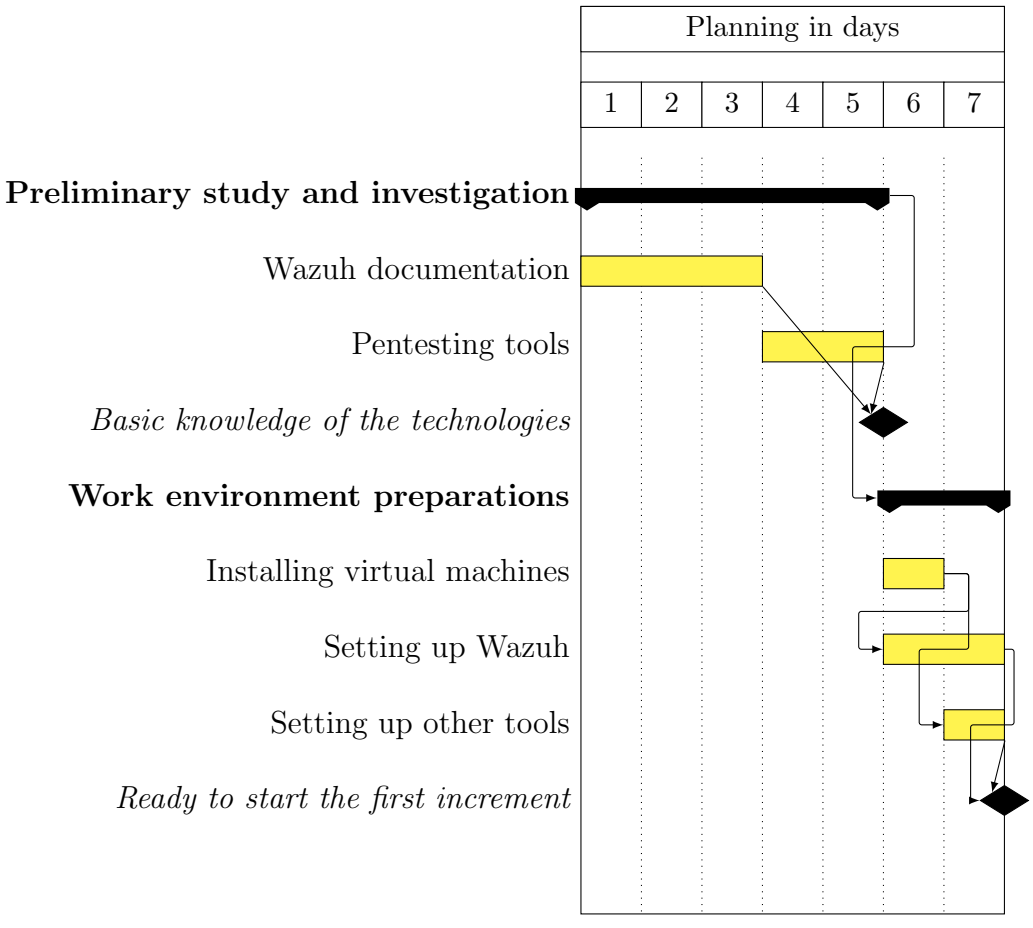


Figure 4.2: "Beginning of the project" planning

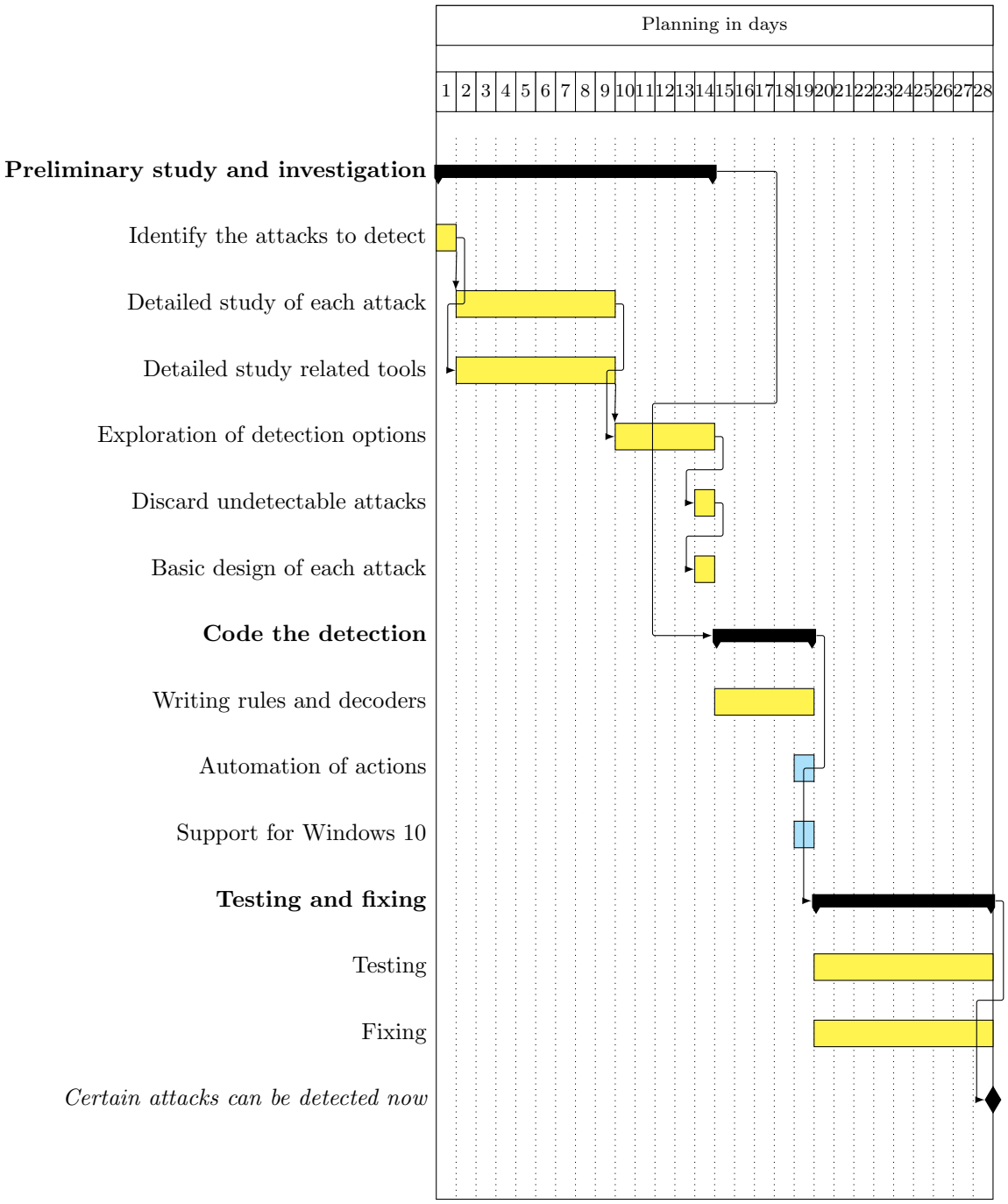


Figure 4.3: "Increment 1: Common attacks in Windows Server" planning

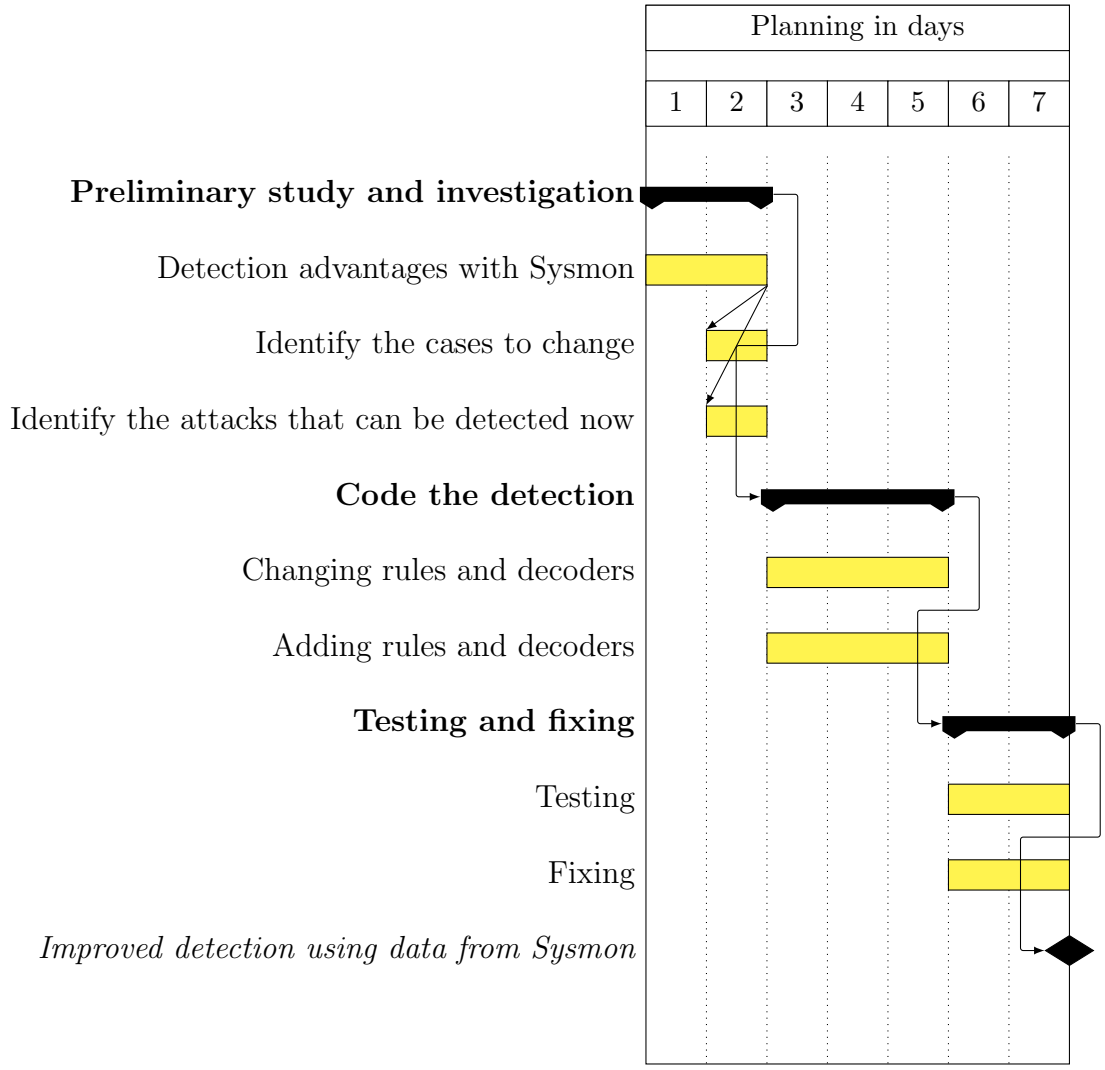


Figure 4.4: "Increment 2: Use of data from Sysmon" planning

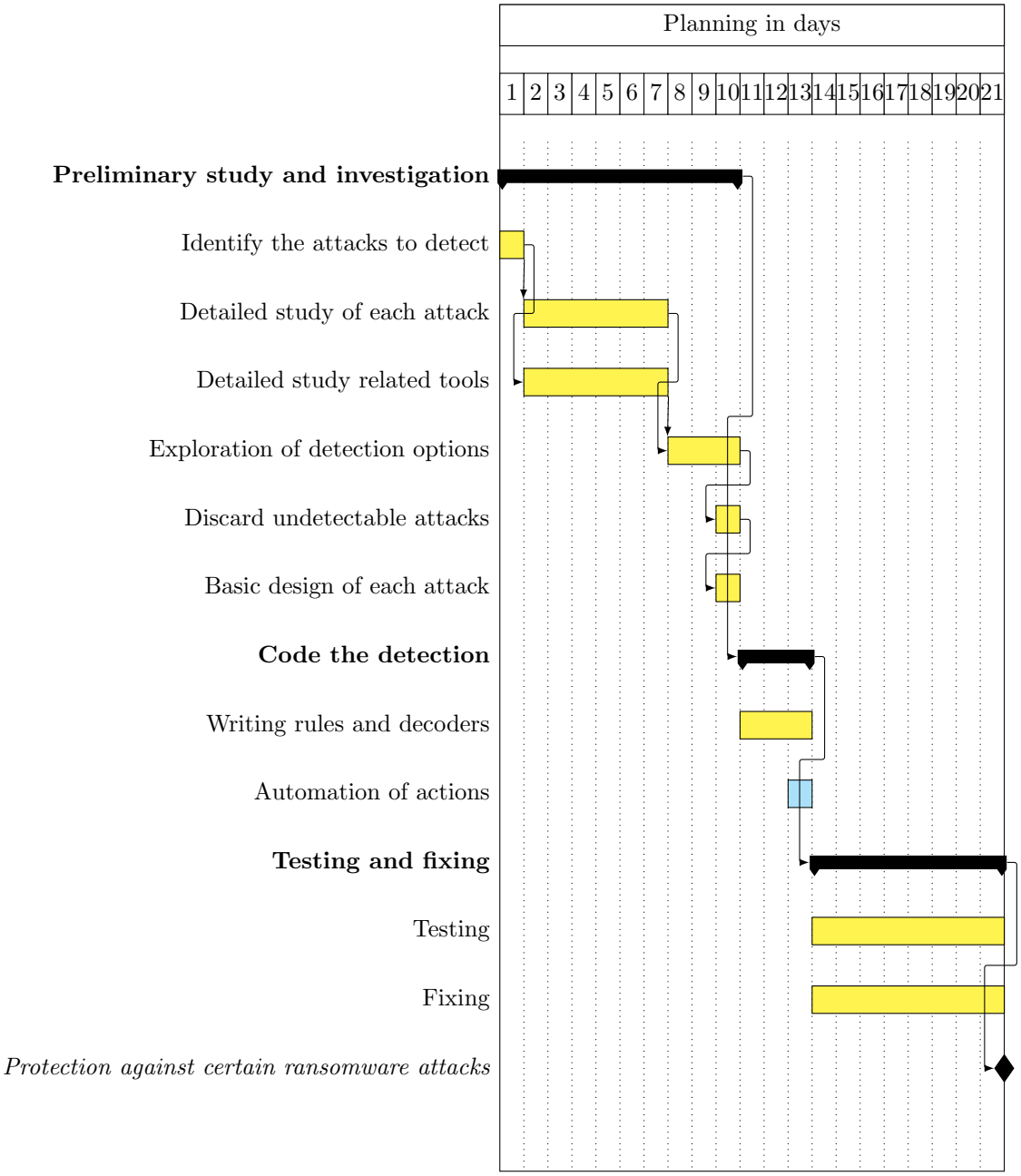


Figure 4.5: "Increment 3: Detection/action against ransomware" planning

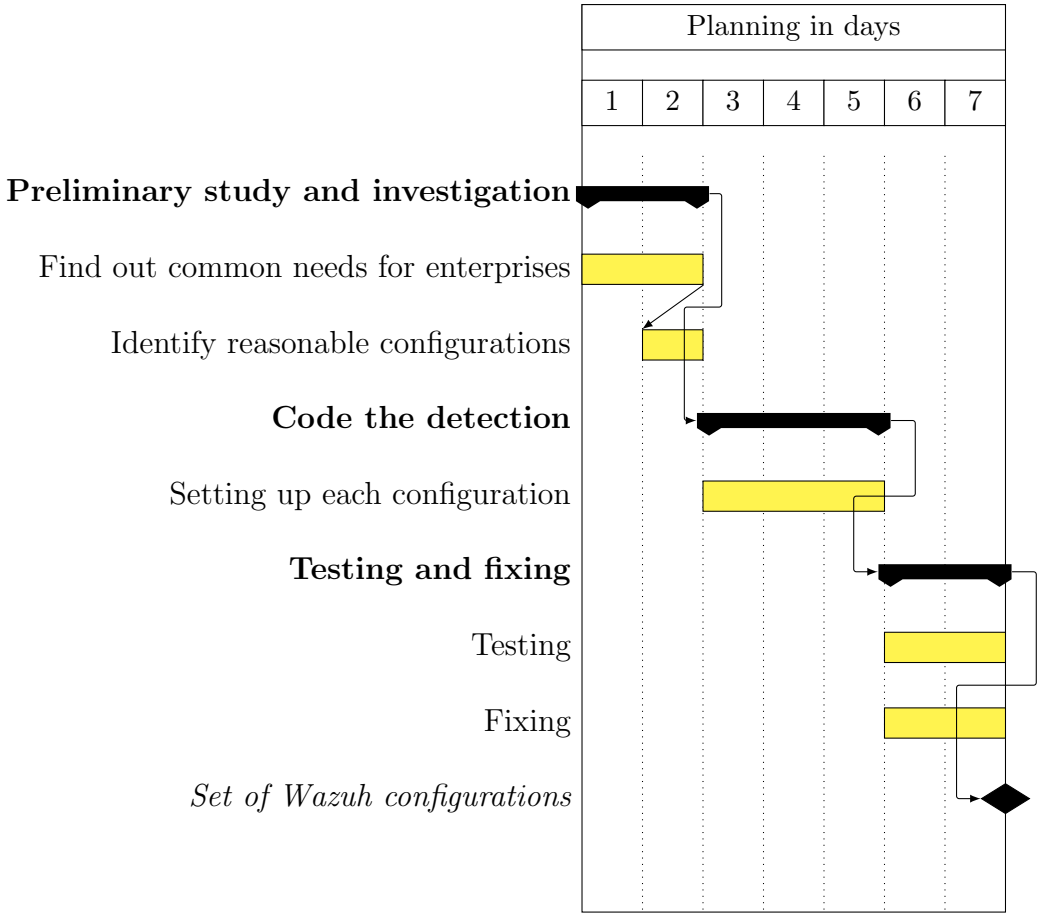


Figure 4.6: "Increment 4: Adapt Wazuh configuration to typical requirements from enterprises" planning

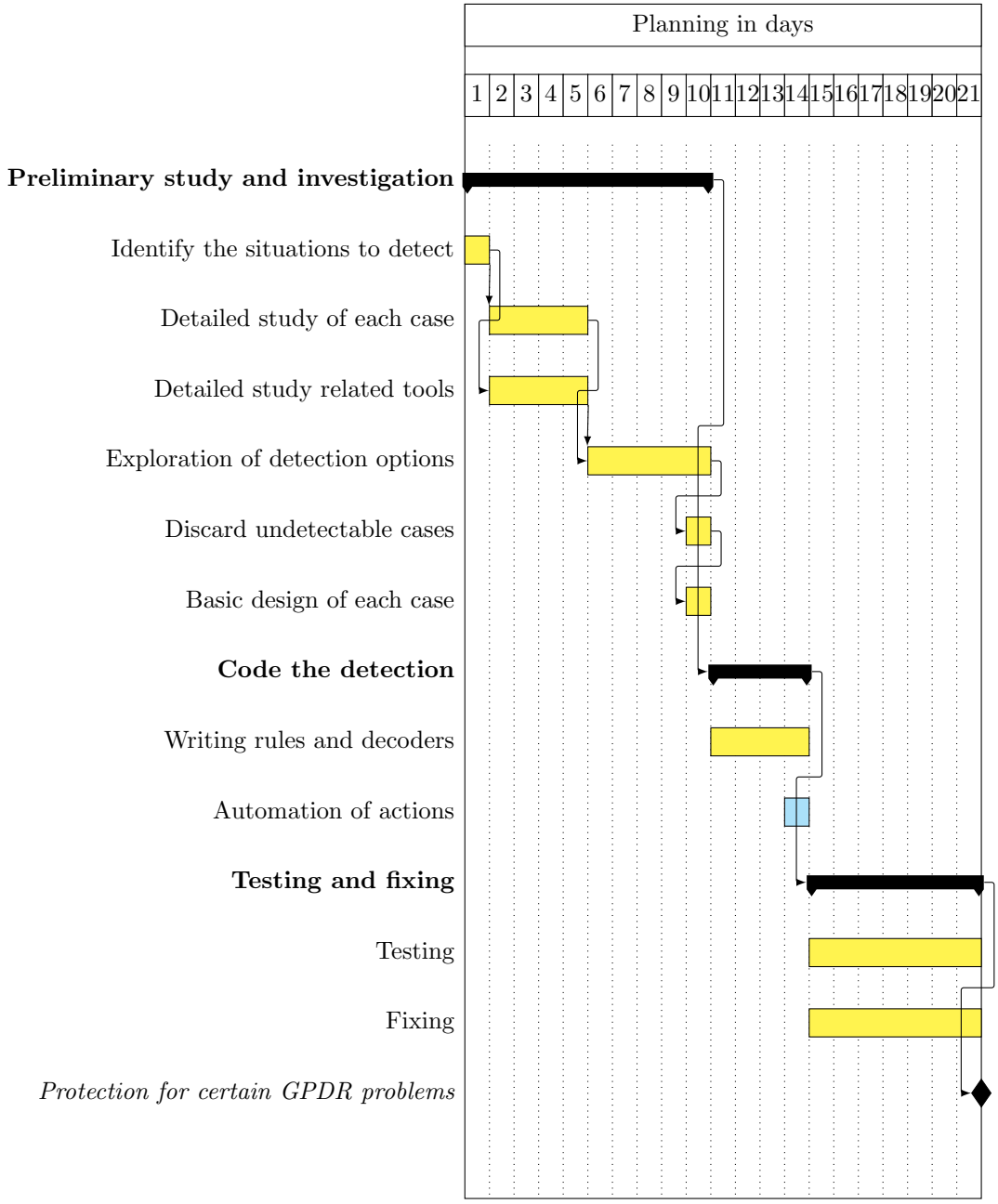


Figure 4.7: "Increment 5: Explore solutions in problems with GPDR" planning

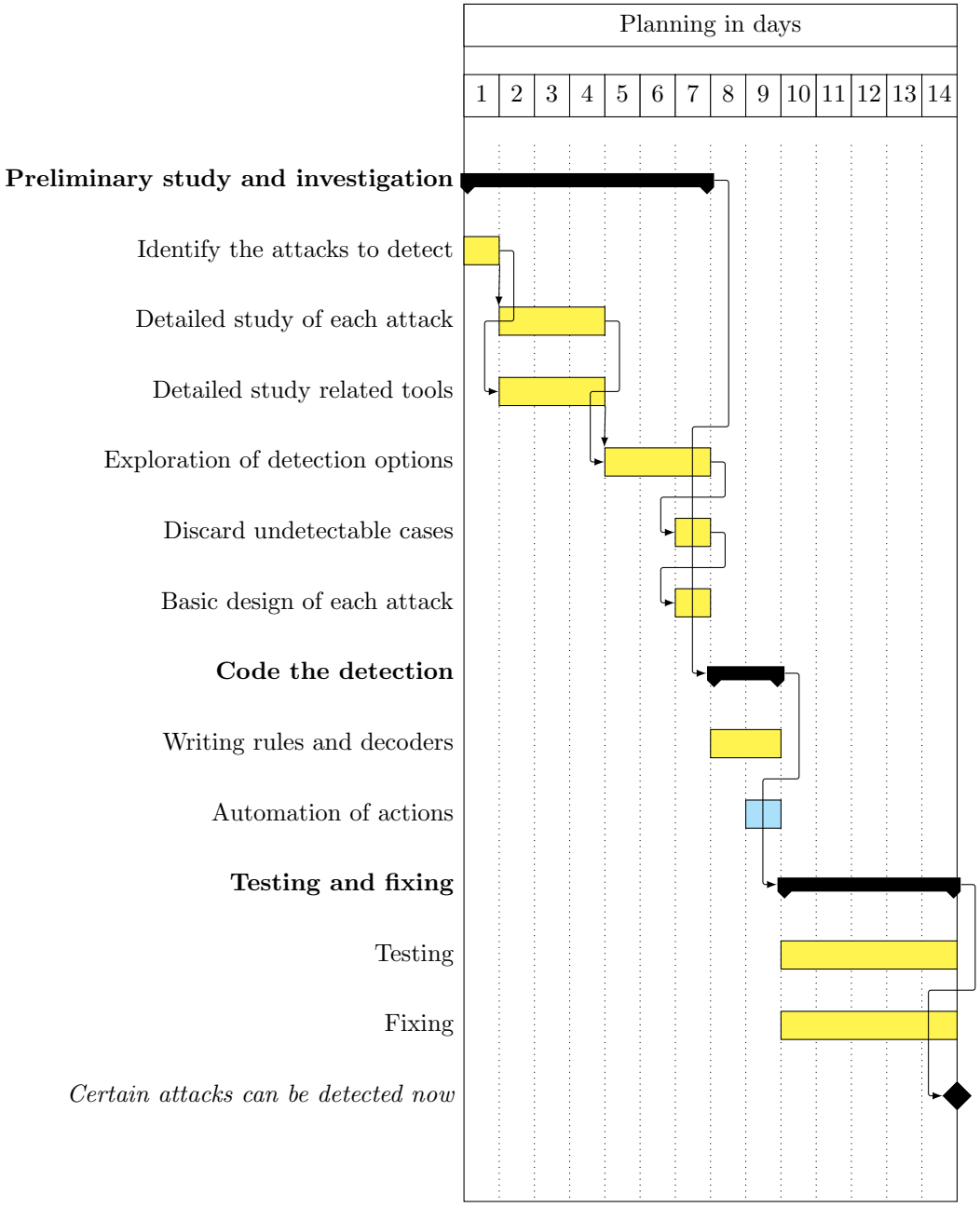


Figure 4.8: "Increment 6: Additional detection for GNU/Linux" planning

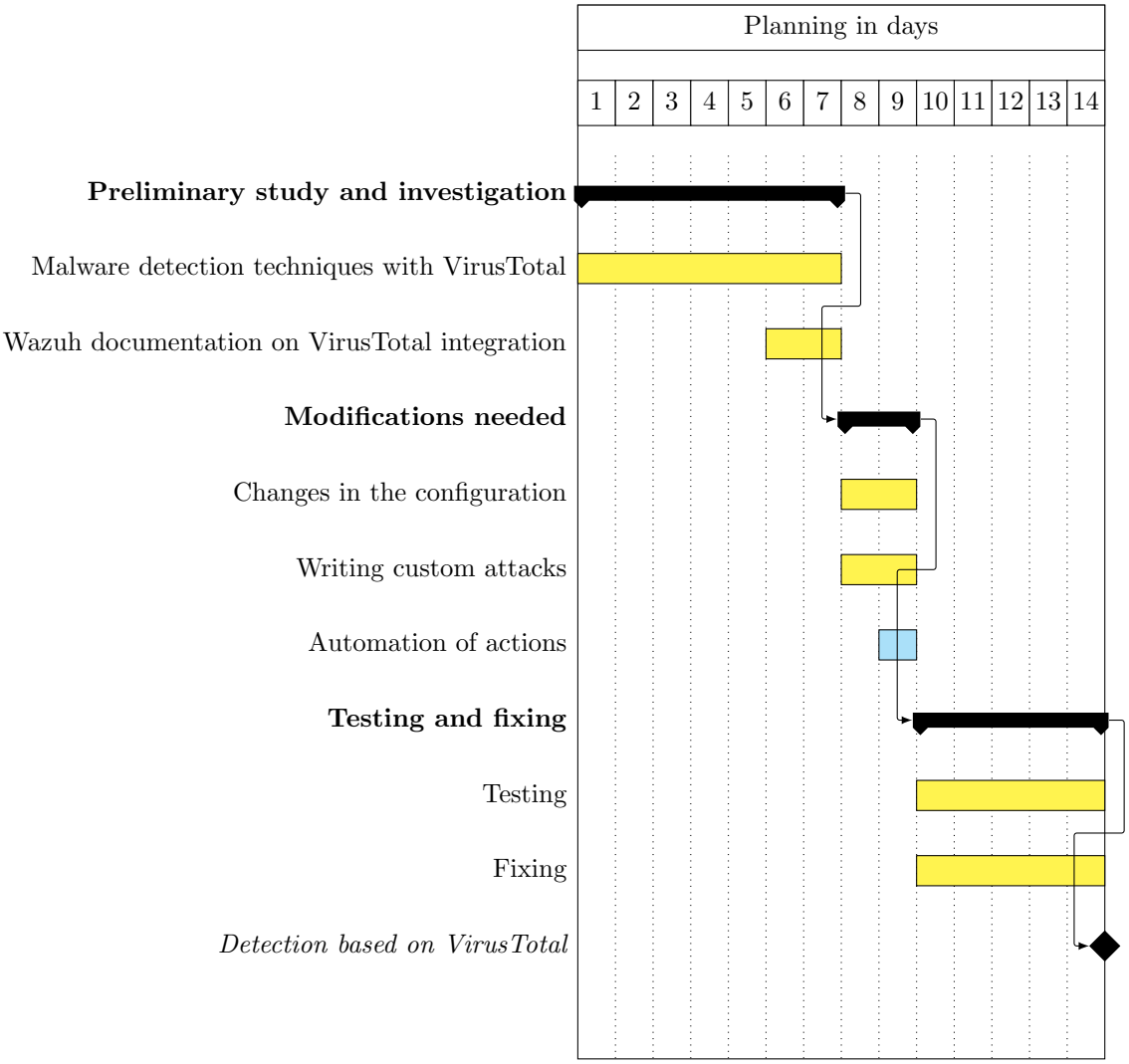


Figure 4.9: "Increment 7: VirusTotal integration" planning

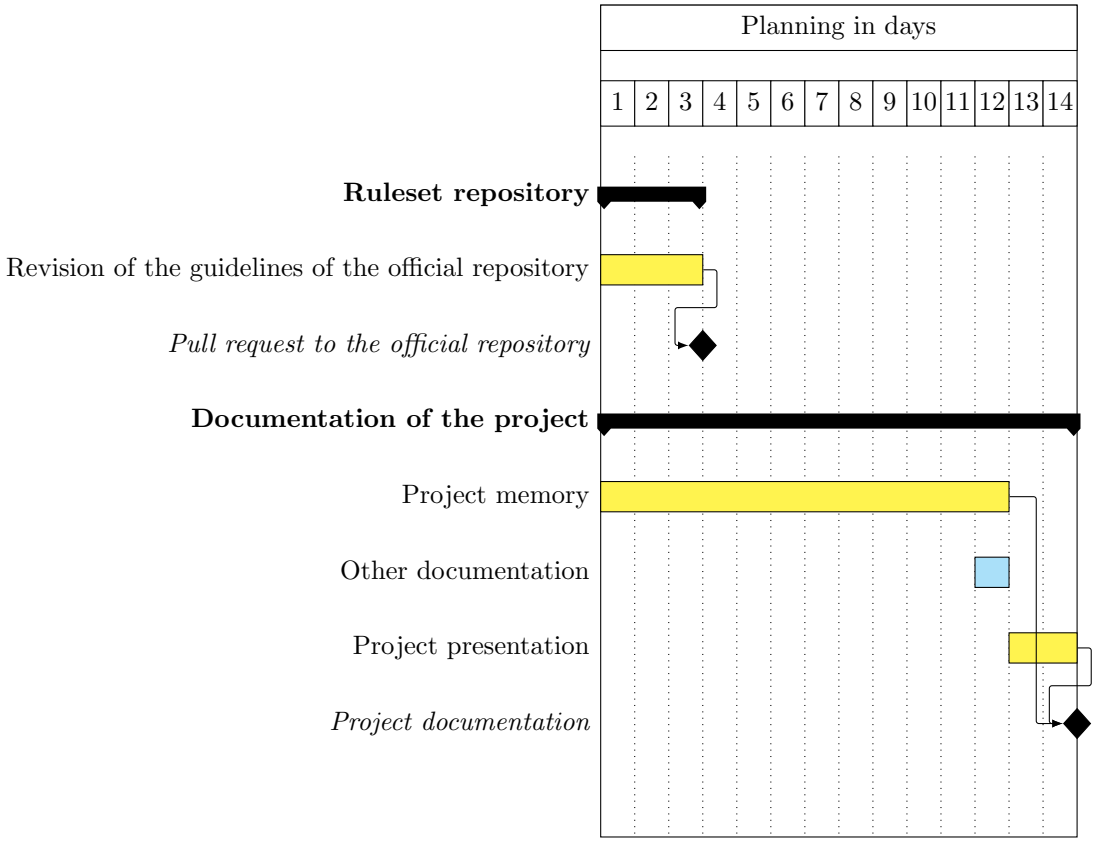


Figure 4.10: "Closing of the project" planning

4.3.4 Real planning

Due to changes on the scope and a 3 month delay due to personal matters of the student there is a big difference with the initial planning.

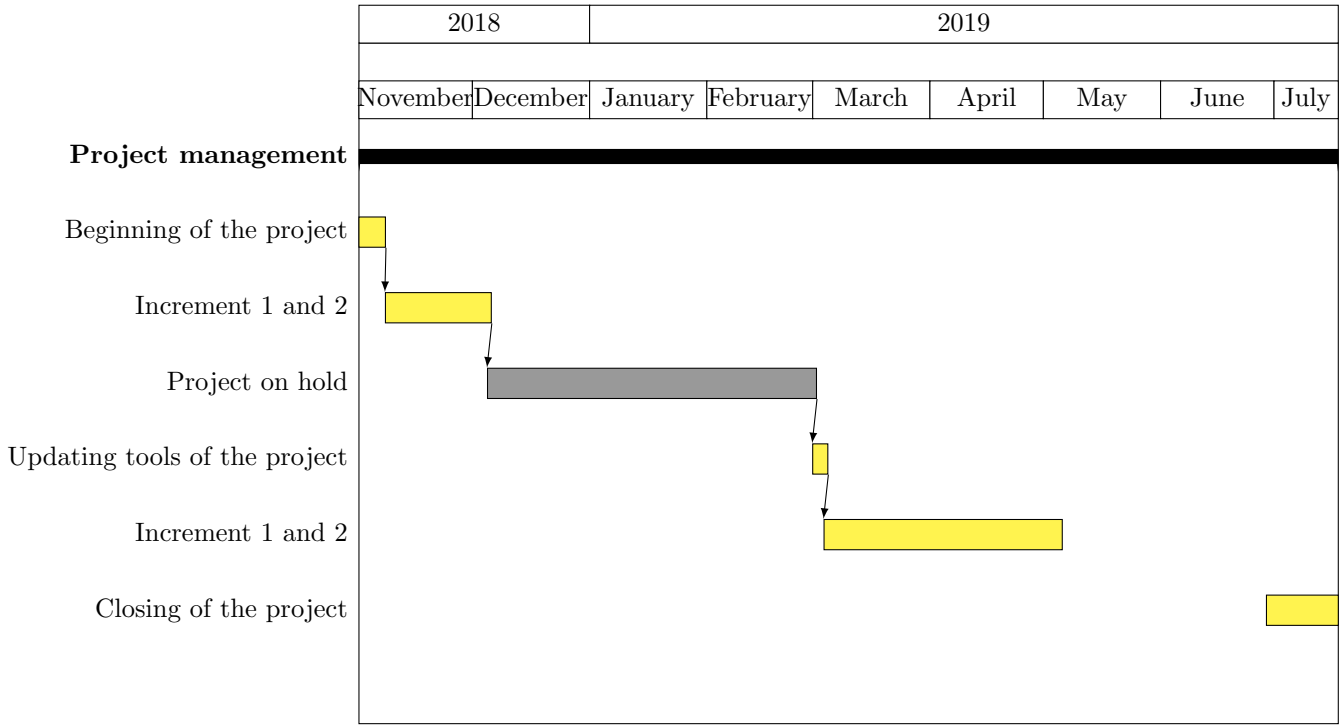


Figure 4.11: Planning simplification

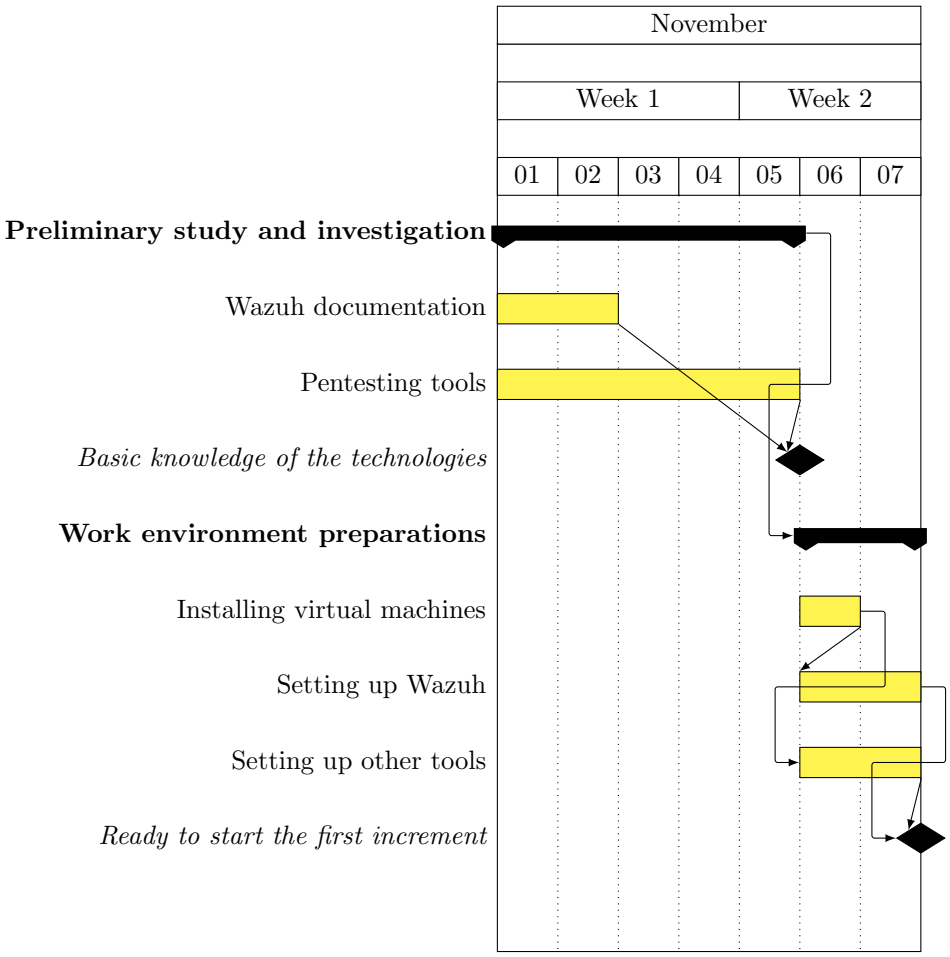


Figure 4.12: "Beginning of the project" planning

This was not planned beforehand but it seemed like a good idea to leave some time to investigate exactly what did change while the project was on hold. Also a new setup of critical installations, because the documentation of the process before was lacking in details.

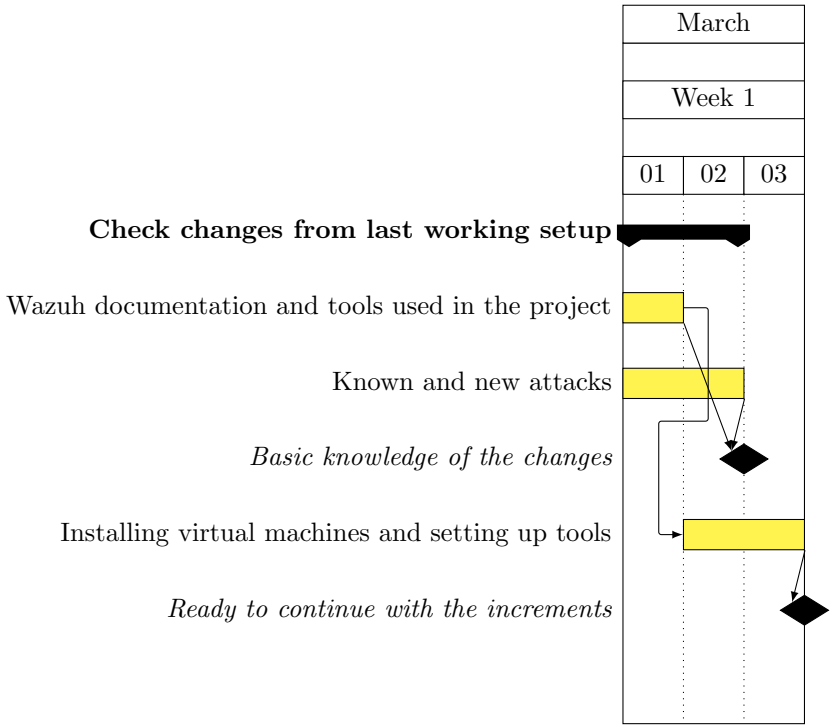


Figure 4.13: "Updating tools of the project" planning

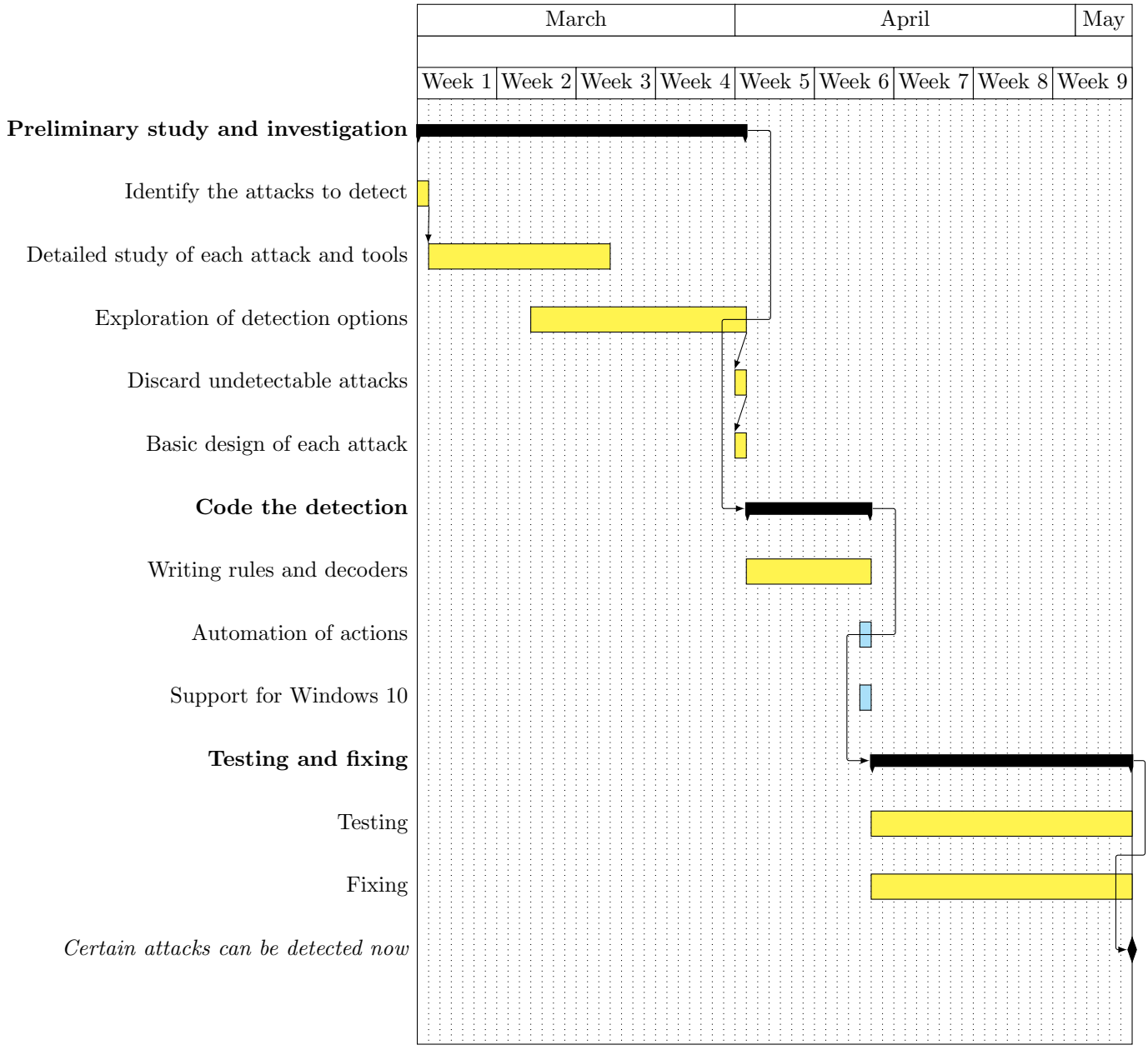


Figure 4.14: "Increment 1: Common attacks in Windows Server" planning

Chapter 5

Technologies and tools

5.1 Development technologies and tools

5.2 Pentesting technologies and tools

powershell, metasploit, mimikatz

5.3 Documentation technologies and tools

git, vim, latex, screenreplay, youtube

5.4 Other technologies and tools

Appendix A

Glossary

API: Application Program Interface. Is a set of subrutines, functions and procedures from a library to be used by other software.

AES: Advanced Encryption Standard. Popular symmetric encryption algorithm with different key lenghts.

CDB: Short for constant database. File format and library for item creation and reading in a database at fast speeds.

ELK: Elasticsearch, Logstash and Kibana. Stack used for Wazuh to gather and transform data.

GPDR: General Data Protection Regulation. Regulation in EU law on data protection and privacy for all individuals within the European Union and the European Economic Area. In practice in this project means law protected files against changes.

HIDS: Host-based Intrusion Detection System.

ICS: Industrial Control System. They are control systems for critical tasks. Normally they are used for industrial control, but in this project we consider any purpouse, like data analysis.

IDS: Intrusion Detection System. Mitigates the damage of intrusions, providing passive protection by alerts.

IPS: Intrusion Prevention System. Minimizes the chance of intrusions, providing active protection by actions.

NIDS: Network-based Intrusion Detection System.

OSSEC: **O**pen **S**ource HIDS **SEC**urity. Is an HIDS solution with detection based on rules and decoders.

TLS: Transport Layer Security. Cryptographic protocol designed to provide communications security over a computer network with hybrid (symmetric and asymmetric) cryptography.

YARA: Tool that does pattern/string/signature matching, with great in performance, results and easiness to write rules.

Appendix B

Bibliography

Books

[23] *Guía de los Fundamentos para la Dirección de Proyectos (Guía del PMBOK) , 5 a edición*. Project Management Institute, 2013. ISBN: 9781628250091.

Articles

[6] Carl M. Hurd and Michael V. McCarty. "A Survey of Security Tools for the Industrial Control System Environment". In: (June 2017). DOI: 10.2172/1376870.

[24] R. (2004) Larson E. & Larson. "Use cases: what every project manager should know". In: *Paper presented at PMI® Global Congress 2004—North America, Anaheim, CA. Newtown Square, PA: Project Management Institute* ().

[27] Syed Ali Raza Shah and Biju Issac. "Performance Comparison of Intrusion Detection Systems and Application of Machine Learning to Snort System". In: *Future Gener. Comput. Syst.* 80.C (Mar. 2018), pp. 157–170. ISSN: 0167-739X. DOI: 10.1016/j.future.2017.10.016. URL: https://doi.org/10.1016/j.future.2017.10.016.

[28] Daniel Zammit. "A machine learning based approach for intrusion prevention using honeypot interaction patterns as training data". In: (2016).

[29] Dr.Najla B. Al-Dabagh and Mohammed A. Fakhri. "Monitoring and Analyzing System Activities Using High Interaction Honeypot". In: *International Journal of Computer Networks and Communications Security* (2014).

[30] Stephan Riebach, Birger Tödtmann, and Erwin P. Rathgeb. "Combining IDS and Honeynet Methods for Improved Detection and Automatic Isolation of Compromised Systems". In: (2005).

Online

[1] *Difference between IDS and IPS and Firewall.* URL: https://security.stackexchange.com/questions/44931/difference-between-ids-and-ips-and-firewall (visited on 11/15/2018).

[2] *About OSSEC.* URL: https://www.ossec.net/about.html (visited on 11/11/2018).

[3] *Wazuh documentation: OSSEC and Wazuh additional functionality.* URL: https://documentation.wazuh.com/current/getting-started/use-cases.html (visited on 11/11/2018).

[4] *Agentless monitoring in OSSEC.* URL: https://www.ossec.net/docs/manual/agent/agentless-monitoring.html (visited on 11/11/2018).

[5] *OSSEC Agents.* URL: https://www.ossec.net/docs/manual/agent/index.html (visited on 11/11/2018).

[7] *10 Top Intrusion Detection Tools for 2018.* URL: https://www.comparitech.com/net-admin/network-intrusion-detection-tools/ (visited on 11/15/2018).

[8] *Who is using YARA.* URL: https://github.com/VirusTotal/yara (visited on 11/15/2018).

[9] *Technologies that form Wazuh.* URL: https://wazuh.com/wp-content/uploads/2015/11/Jigsaw4.png (visited on 11/11/2018).

[10] *Wazuh documentation: Welcome to Wazuh.* URL: https://documentation.wazuh.com/current/index.html (visited on 11/15/2018).

[11] *Wazuh documentation.* URL: https://documentation.wazuh.com (visited on 11/11/2018).

[12] *Wazuh ruleset.* URL: https://github.com/wazuh/wazuh-ruleset (visited on 11/11/2018).

[13] *Wazuh: Github.* URL: https://github.com/wazuh/wazuh (visited on 11/11/2018).

[14] *Wazuh documentation.* URL: https://github.com/wazuh/wazuh-documentation (visited on 11/11/2018).

[15] *Wazuh documentation: Installation guide.* URL: https://documentation.wazuh.com/current/installation-guide/index.html (visited on 11/11/2018).

[16] *Wazuh documentation: Architecture.* URL: https://documentation.wazuh.com/current/getting-started/architecture.html (visited on 11/15/2018).

[17] *Wazuh data flow.* URL: https://documentation.wazuh.com/current/_images/wazuh_data_flow1.png (visited on 11/15/2018).

[18] *Wazuh documentation: Custom rules and decoders.* URL: https://documentation.wazuh.com/current/user-manual/ruleset/custom.html (visited on 11/15/2018).

[19] *OSSEC-Wazuh Ruleset list.* URL: https://wazuh.com/resources/OSSEC_Ruleset.pdf (visited on 11/15/2018).

[20] *Wazuh documentation: Testing decoders and rules.* URL: https://documentation.wazuh.com/current/user-manual/ruleset/testing.html (visited on 11/15/2018).

[21] *Wazuh documentation: JSON decoder.* URL: https://documentation.wazuh.com/current/user-manual/ruleset/json-decoder.html (visited on 11/15/2018).

[22] *Wazuh documentation: CDB lists.* URL: https://documentation.wazuh.com/current/user-manual/ruleset/cdb-list.html (visited on 11/15/2018).

[25] *What is Project Management?* URL: https://www.pmi.org/about/learn-about-pmi/what-is-project-management (visited on 11/17/2018).

[26] *VirusTotal FAQ.* URL: https://www.virustotal.com/en/faq/ (visited on 11/18/2018).