ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA
SCUOLA DI INGEGNERIA ED ARCHITETTURA
CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA

ICT Security: defence strategies against targeted attacks

Tesi in
Progetto di Reti di Telecomunicazioni LM

Relatore
Prof. Franco Callegati

Presentata da
Daniele Bellavista

Correlatore
Ing. Marco Ramilli

Sessione Terza
Anno Accademico 2012-2013

to CeSeNA, Anesec and Room 404,

Contents

Chapter 1

Introduction

In March 2013, the guardian published an article with a shocking title: *Is 2013 the Year of the Hacker?*[1]. It states that 1.5 million peoples are impacted by cyber attacks every day with an increasing rate and that even industry giants like Facebook or Dropbox have been hit by cyber attacks.

Similar articles became common in international and national journals, presenting the cyber crime, often associated with the term *hacker* or *hacking*, as a rising and dangerous threat that may hit every person, society or nation. Governments and industries are also showing a raising awareness to security risks. Nowadays, the investment in cyber security defences is becoming popular by promoting security researches, bug hunting and by developing new standards.

The current cyber defence race is justified by the continuous cyber-attacks that are massively performed since the Morris worm. Although the most famous cyber security event of 2013 was the *global surveillance disclosure*, when Edward Snowden leaked several top secret documents, several cyber attacks were performed in the last year, for instance:

- **Spamhaus attack**: the second largest DDoS attack ever. Nine day of DNS spoofing starting from March 18th (the primate for a DDoS attack goes to the CloudFlare attack, dated February 10th, 2014 with 400Gbps).
- **Department of Energy breach**: performed in July, this data breach was possible by exploiting backdoors installed after a previous attack performed in February.
- **Columbian Independence Day Attack**: on July 20th, 30 Colombian government websites were defaced or shut down.
- **New York Times Attack**: the Syrian Electronic Army targeted the New York Times' website in August 2013.

[1]http://guardian.co.tt/business-guardian/2013-03-11/2013-year-hacker

- **Adobe data breach**: before the Snowden leak, it was the most famous data breach of 2013. In September, a successful attack took millions of user credentials and, more important, source code of some Adobe products.
- **Data-Broker botnet**: in September, a botnet was discovered in the data-broker servers. The malware remained undetected from detection systems for over a year.

The list could be expanded and not only with industrial giants or governments, but also with average network users, infected by malware, because operating systems and software like browsers and PDF readers are constantly analyzed to find exploitable bugs. Moreover, the massive diffusions and connections of smartphones expanded the security context to both mobile OSs and applications. One notorious example is Whatsapp, which protocol has been hacked several times.

The increasing rate and scope of cyber attacks can be attributed to evolution of computer technology, especially when dealing with internet connectivity. Cloud computing and BYOD solutions, for instance, enlarged the circle of thrust, creating situations that widen the attacker entry points and making the defence work more difficult.

My work, however, doesn't concern the study of new security solutions in the context of new technologies, but in the study of strategies against targeted attacks, which differ from the normal cyber attacks. While it may seem that the cyber crime simply followed the technological evolution, the cost reduction of resources, software development and the information spreading led to new conceptual scenarios like *hacktivism*, *industrial espionage* and *cyber-warfare*, where the attacker has a specific target to compromise. Strictly speaking, they are not new scenarios, but the modern context enlarged the actor set, by bringing targeted attacks within the reach of a wider range of people.

Targeted attacks represent the major threat against companies, governments and military organizations. In the last few years, the term *targeted attack* started to be used and now it's becoming one of the major topic in security communities. Many security solutions claim to offer protection against targeted attacks and academic and professional researcher starts to study the problem as well as approaches to detect and counter targeted attacks.

This thesis aims to analyze differences between targeted and opportunistic attacks, to examine existing defense approach and to propose some extensions when dealing with targeted attacks. The method is then applied in the real world, by performing analysis and mitigation of cyber-attacks performed against an Italian company, which I will refer to as `ACME Corporation`.

Chapter 2

Cyber Attacks

This chapter describes what a cyber attack is and how it's performed, by analyzing existing taxonomies and common attack scenarios. Then opportunistic attacks and targeted attacks are compared, highlighting the main differences between them.

2.1 Malware

A malware is any file containing executable code, such as binary files, scripts and documents with active content (e.g. PDF), that performs malicious actions. They are used in cyber attacks, with various motivations (see section 2.2). In this section, I describe types of malware and techniques commonly used by malicious programs.

2.1.1 Types of malware

Malware programs have been classified basing on the observed behavior in the wild[10]. Each category isn't mutually exclusive: modern malware have a high level of complexity and exhibit multiple different behaviors. The following list is intended to list features and strategies that can be seen in malware programs:

Worm - A worm is an independent program that can propagate itself to other machine, usually by means of network connection. The Morris Worm, developed in 1986, was one of the first computer worm capable of distributing over the internet[42]. Nowadays worms exhibit complex behavior, for instance the Conficker worm propagate itself using flaws of windows systems and creates a botnet used for malicious activities. Conficker is also capable of updating itself to newer version[32].

Virus - A virus is a piece of code, incapable of running independently, which adds itself to other programs in order to be executed[42]. Viruses, first developed between 1981-1983[1], don't propagate through the network; they rather infect files in network shares or removable media to spread to other computers.

Trojan Horse - A trojan horse is program that executes a benign task, but secretly performs a malicious behavior[1]. Trojan horses have been reported since *1972*, but they are common even now as the form of browser plugin, downloadable managers, and so on[10]. A **Remote Access Trojan** (RAT) is a trojan that installs a backdoor to allow a remote control of the system.

Spyware - A spyware is a program, incapable of replication, which passively records and steals data from the infected host. A spyware may steals usernames and password by using a passive keylogger, email addresses, bank accounts, software licence, and so on[1]. A less dangerous spyware, called **adware**, is a malware that steals market-oriented information, such as visited websites.

Bot - A bot is a malware that allow the attacker to remotely control the infected system[10]. Bots take orders from the command and control center (aka C&C) and, for instance, can be used to deploy distributed denial of service attacks (DDoS) or bitcoin mining.

Rootkit - A rootkit is a malware capable of hiding itself from user and analysis application. It can operates at any level, user space, kernel space or even by creating a virtual machine below the operating system[10, 3].

Logic Bomb - A logic bomb is a program that executes a malicious payload when a triggering condition becomes true[1]. For instance, the trigger can be a date, the presence of a file or a mouse click (like the Upclicker malware[41]).

Dropper - A dropper is a malware that *drops* other malware on the system. The dropper can be persistent and continuously drop malware or it can delete itself once the dropped content is installed.

Malware Toolkit - A toolkit is not malicious per se, but it's capable of generating malware versions by specifying desired features and behavior, such as exploits and targeted. Famous malware toolkits are the BlackHole Exploit Kit (which creator was arrested in 2013[26]) and the Zeus Agent Toolkit.

2.1.2 Malware infection techniques

When designing a malware, the attacker must consider how to infect the system, hide the malware from the user, detect defence systems and optionally evade them. For more details on malware analysis, see section 3.6.

In order to evade signature-based detection and static analysis, a malware can implement one or more of the following techniques:

Encryption - The malware body or part of it is usually encrypted, or obfuscated, to hide the malicious code and to disrupt antivirus signatures. The encryption can be static or use a key.

Oligomorphism - Oligomorphism is an improvement of encryption. At each infection the malware changes the encryption key and/or the encryption algorithm from a predefined set[1].

Polymorphism - A polymorphic malware implements the same concepts of oligomorphism, but it uses dynamically generated encryption function, but modifying its code, by using a mutation engine[1]. The mutation engine can operate in various ways, for instance by using equivalent instruction, changing the used registers, by introducing concurrency, junk code or by inverting instructions.

Metamorphism - Metamorphism is the process of changing at each infection the malware code, inclusive of the mutation engine itself. A metamorphic malware doesn't need to encrypt its body to evade signatures, because the code itself is changed at every infection[1].

Packing - Packing is another approach to hide the malware code and to evade signatures. The malware is compressed, encrypted and wrapped by a second program called packer. The duty of the packer is to instantiate a valid environment for the malware, by loading libraries, creating memory segments and API wrappers. Then, the packer decrypts the malware and executes it without writing the code on the disk. The packing operation doesn't need a mutation engine or complex encryption step for parts of the malware body. On the other hand, an unpacked malware is vulnerable to dynamic analysis (see section 3.6).

Encryption techniques are deployed also to thwart static binary analysis. If strings, values and the malicious code are encrypted, the instruction flow of the malware is nearly impossible to follow. To overcome these techniques, modern defence systems perform dynamic analysis by executing the malware inside an emulated system. In such cases, a malware have to detect the emulated environment and to exhibit malicious code only in real machines. This is a typical behavior of logic bombs: if the emulated environment is not detected, then execute the malicious code is executed. Common dynamic analysis evasion includes:

Red pills - Red pills are detection methods, which exploit flaws in the CPU emulation. A red pill invokes certain assembly instructions that are known to produces different results inside the emulated environment.

System alternation detection - Emulated systems may present or lack of devices, services or processes that are respectively missing or present in real environment.

Sleeping malware - A simple counter action to automatic analysis is to wait for it to finish. Real-time defence systems usually perform a time-limited analysis. If the malware sleeps for long enough, the detection could be bypassed.

Wait for user interactions - Like the Upclicker malware, a malicious action could be triggered only when a user input is detected. In emulated environment the human user is missing and the defence system can be detected by the lack of its actions. Advanced defence systems emulate user interaction to fool such behaviors.

Multi-stage attack - Multistage attacks are organized in a way that the downloaded malware is not complete. Its infection process is divided into multiple stages: at each stage the malware downloads another component, which precedes the infection[24, 35].

2.2 Opportunistic and targeted attacks

When dealing with targeted attacks it's important to identify when an attack can be considered opportunistic and when targeted in order to distinguish truly dangerous targeted attacks among the countless opportunistic events that may resides in the network.

2.2.1 Opportunistic attacks

In opportunistic attacks, the attacker has a general idea of what to do and chose its victim among the possible exploitable targets. Recurring motivations for opportunistic attacks may be:

- Build or expansion of an illegal infrastructure: the attacker wants to build or extends an illegal infrastructure for phishing, spam or botnet. The target is irrelevant since the desired outcome is to infect as many hosts as possible.
- Financial gain: bitcoin mining, ransonware and credit card stealing are common objective of opportunistic attackers.
- 15 minutes of fame: any vulnerable website or service is defaced or taken down by a solo attacker or an organization.

Opportunistic attacks' scenarios have in common the absence of a specific target, resulting in a large number of riskless attacks, as confirmed by the Verizon data breach analysis[11, 20, 21, 22], which states that:

1. More than 70% of attacks as opportunistic.
2. On average, a system on the internet is scanned after 11 minutes from the connection.
3. 97% of opportunistic attackers try one port and 81% send a single packet

When the attack involves the use of malware, the malicious software must be designed to infect a generic target, without a priori knowledge of OS version and defence systems involved. It must also be fast to produce to maximize the duty time between the zero-day and the bug correction. Moreover, since the target is unknown, the attacker has to take in account that the malware could infect a honeypot, or will be analyzed in sandboxes. In order to avoid the automatic identification the malware must contain anti-analysis techniques.

These problems led to the diffusion of *malware toolkits* that automatize the malware generation. The result is a set of malware, equal in behavior but with different signatures, which are generated every day by the thousands, captured in honeypots and marked in a few weeks.

2.2.2 Targeted attacks

Targeted attacks differ from opportunistic because of the choice of the target. This is the crux that influences every step of the attack process.

As first difference, motivations for targeted attacks share always an implicit assumption: the attacker is not going to give up, since the objective is reached only by attacking the specific target.

- Financial Gain: the attacker has financial benefits by stealing information or by disrupting a service, for instance by executing a DoS on the main web-site of the target.
- Activism: instead of money, emotions guide the attacker. The objective may be the same as the financial gain, but the attacker is driven for instance by political or religious believes.
- Cyber warfare: the attacker launches an assault against a nation, targeting strategic objective.

The second difference is the customization. Targeted attacks vector contains evidence that the attackers has specifically selected the recipient of the attack [44]. Emails, for instance, contain evidence of an advanced knowledge of the target beside address and name: format, terminology or the plausibility of the content is a sign of strong information gathering. Also malware differs from opportunistic attacks. Targeted malware are specifically designed to bypass the target defence systems and doesn't belong to any known family, thus, no signature is present. Unlike opportunistic attacks, a targeted malware requires more skills for its design and development.

The third difference is organization. An opportunistic attack is an end unto itself, while targeted attacks are usually part of an attack campaign, aimed to cause a persistent damage to the target. Once the target is compromised and the objective is reached, the attacker installs a remote access toolkit inside the host to guarantee a persistent access. In such situation, the targeted attacks can be called advanced persistent thread (APT).

2.3 Cyber attack process

At the turn of 21th century, due to the increase of the internet, illegal activities involving network and computer were impacting a high number of users. The increasing connectivity also opened attack paths from across the world: with the right resources, cyber warfare operations could be undertaken, threatening the target nation's security.

While attack taxonomies existed since the Morris Worm, when the CERT was founded, early works were focused on the classification scheme of security flaws, like the Bishop's taxonomy of UNIX system and network vulnerabilities [2, 34]. Newer taxonomies faced the problem of which intentions and objective drive the attacker and how to relate them in a larger attack campaign. They still however bind too much to the technical details of vulnerabilities and security flaws which fail in the classification of a structured and organized targeted attack.

Over the last few years targeted attacks evolved into specialized and persistent attacks aimed to steal information, compromise the infrastructure and leave backdoors to guarantee a persistent access. Moreover, attackers and targets may have relationships with organization or nations, that could help understand what is the real objective and how to defend it [34, 44].

These new scenarios, more similar to cyber warfare than cyber attacks, cannot be handled by the classical taxonomies, thus the security community produced methods to profile attack processes rather than to classify them basing on technical properties. Taxonomies moved to a higher level, to be capable of describing multiple actors, the relation between them and the attackers' modus operandi.

2.3.1 Howard’s Common Language: a first attack process taxonomy

One notorious taxonomy is the Common Language, published by John Howard in 1997 [16]. Howard expanded the existing CERT taxonomies by modelling a primitive attack process. It defines the concept of *attacker*, who wants to achieve a set of *objectives* by means of computer *attacks*. A group of attacks sharing the same attacker, objective and timing is called incident.

Objective: purpose or end goal of an incident.

Attacker: an individual who attempts one or more attacks in order to achieve an objective. Basing on the objective, the attacker can be classified using various term such as hacker, spy or terrorist.

Attack: a series of steps taken by an attacker to achieve an unauthorized result.

For the taxonomy to be complete, the attack is divided into steps which formalize an attack classification. An attack starts with a *tool*, which exploits a *vulnerability*. The exploit allow a series of *actions* on the *target*, which may lead to an *unauthorized result*. A single action on a target is called *event*.

Tool: a means to exploit a vulnerability. As Howard states, this is the most difficult connection because of the wide variety of methods.

Vulnerability: a weakness in a system allowing unauthorized action.

Action: a step taken by a user or process in order to achieve a result.

Target: a computer or network logical entity or physical entity.

Unauthorized result: a consequence of an event that is not approved by the owner or administrator.

2.3.2 Cyber warfare taxonomy and military doctrine

In 2001, the Lieutenant Colonel Lionel D. Alford Jr. published an issue, proposing a military-oriented taxonomy for cyber crime acts. He recognized the danger of a targeted attack toward modern software-controlled components and urged to develop an appropriate defence plan. *“Cyber warfare may be the greatest threat that nations have ever faced. Never before has it been possible for one person to potentially affect an entire nation’s security. And, never before could one person cause such widespread harm as is possible in cyber warfare.“* [23].

Inspired by military taxonomies, Lt. Col. Alford identified the main stages that occur or may occur during a cyber attack:

Cyber Infiltration (CyI): during this stage, the attacker identifies all the system that can be attacked, that is, every system that accepts external input. The infiltration can be both physical and remote.

Cyber Manipulation (CyM): following infiltration, an attacker can alter the operation of a system to do damage or to propagate the infection.

Cyber Assault (CyA): following infiltration, the attacker can destroy software and data or disrupt other systems functionality.

Cyber Raid (CyR): following infiltration, the attacker can manipulate or acquire data and vital information.

A cyber attack can be in the context of **cyber warfare (CyW)** or **cyber crime (CyC)**, being the attack intent to affect national security or not. Also, the taxonomy includes some primitive motivation of actors and their intent. The doctrine distinguish between **intentional cyber warfare attack (IA)** and **unintentional cyber warfare attack (UA)**. The responsible of the cyber attack are called **intentional cyber actors**.

Despite the military terms, the taxonomy describes actions that occur during targeted attacks, being the target a military organization or not. Moreover, the inclusion of both physical and remote access to the computer and network infrastructure is a common point with targeted attacks taxonomies.

2.3.3 AVOIDIT an extensible taxonomy

AVOIDIT is an acronym for Attack Vector, Operational Impact, Defense, Information Impact and Target. It's an often cited taxonomy in the security community because it first introduces the concept of impact of an attack.

AVOIDIT use five extensible classifications:

Classification by Attack Vector: an attack vector is defined as a path by which an attacker can gain access to a host. The path is defined as a vulnerability exploited by the attacker.

Classification by Operational Impact: the operational impact determinates what an attack could do if succeeded. The classification is based on a custom mutually exclusive list, for instance web resource, a malware installation or Denial of Service.

Classification by Defence: classification by defence classifies the attack basing on the defence strategy against the pre- and post- attack. The defence can involve both a *mitigation* of the attack damage and a *remediation* to the vulnerability.

Classification by Informational Impact: the informational impact determinates how the targeted sensitive information has been impacted by the attack.

- Distort - The information is modified.
- Disrupt - The access is changed or denied, for instance as a consequence of a DoS.
- Destruct - The information is deleted.
- Disclosure - The information is accessed from an unauthorized viewer.
- Discover - Previously unknown information is discovered by the attacker.

Classification by Attack Target: the target of the attack is any attackable entity, such as users, machines or applications.

The AVOIDIT taxonomy allowed to classify complex attacks, by considering both the various attack vectors and the information impact. It lacks however of physical attacks classification[40] which is an important aspect in targeted attacks.

2.3.4 Kill chain for targeted attacks

The kill chain for targeted attacks was never published, but proposed in a Withe Paper by Lockheed Martin's company [18]. Authors propose a military-inspired intrusion kill chain, with respect to computer network attack (CNA) or computer network espionage (CNE). The model describes the stages the attacker must follow as a chain, thus profiling already performed attacks or identifying current attacks.

A defence strategy based on kill chain assumes that any deficiency will interrupt the entire process.

1. **Reconnaissance:** intelligence and information gathering phase.
2. **Weaponization:** create the malware and inject it in a *weaponized deliverable.*
3. **Delivery:** transmission of the weapon by means of an attack vector.
4. **Exploitation:** the malware acts, exploiting the system and compromising the machine.
5. **Installation:** the malware installs a RAT (remote access tool) or a Backdoor to allow the attacker an access to the environment.

6. **Command&Control:** the malware establishes a connection to the attacker C&C, getting ready to receive manual commands.

7. **Actions on Objectives:** the attacker takes actions to achieve his objective.

2.3.5 Modus Operandi for industrial espionage and targeted attacks

The paper "Industrial Espionage and Targeted Attacks: Understanding the Characteristics of an Escalating Threat" in 2012 [44], doesn't propose a taxonomy, but rather a modus operandi for targeted attacks. Authors recognize the complexity of a targeted attack, and that the targets can be also small industries and not only large corporations or governments.

The paper identifies a typical modus operandi, based on forensic investigation of multiple targeted attacks:

Incursion: in this phase the attacker attempts to penetrate a network.

Discovery: once inside, the attacker analyzes the network to spread the infection and identify the real objective.

Capture: when a worthy computer is infected, the attacker *installs* an advanced RAT (remote access toolkit) to improve the stealth capabilities.

Data Exfiltration: the attacker uses the RAT to steal documents, password and blueprints. The attacker can also use the advanced capabilities to spread deeper into the network.

2.4 Attack process details

This section describes common approach used when performing cyber attacks. It's important to understand how a cyber attack is performed in order to design a proper strategy.

2.4.1 Information Gathering

The first stage of every attack is the information gathering, which provides the intelligence needed by every subsequent phase.

Since recon is the very first phase, the attacker will try not be detected, because an early detection could warn the target and disrupt the entire attack. From the defender side, this is also the most difficult phase to detect, since most of the information can be gathered in legal ways.

A list of critical information needed by the attacker could be[9]:

Employee's personally identifiable information: phone numbers, addresses, biometric records and so forth can be used for the selection of the right target and attack vector.

Network layouts: web and mails servers, their location, software version and system fingerprints reveal necessary information for exploitation, installation and C&C phases.

Company files: reports, databases and source code, for instance, could reveal software and useful data.

Company information: information such as partners, services, mergers and acquisitions, facilities plants could delineate social engineering approach and attack vectors via physical infiltration.

Organizational information: for a successful social engineering attack and in order to identify the correct target the attacker needs technical staff and C-team names, organizational charts and corporate structure details.

Work interaction: emails content templates, inter-communication protocols, security and authentication mechanisms are essential information for the attack design.

Tons of public information can be obtained by searching in employee's social networks profiles, public calendar entries and via specific Google searches. Private information can be harder to obtain and often requires the use of social engineering against the company personnel.

2.4.2 Attack vector

Most attacks are performed by creating a malware to be executed inside the target network. However those malware need to be disguised, to bypass both security programs and humans by creating a weaponized deliverable. A typical weaponization it is performed by embedding the malware inside a valid program, thus creating a trojan, or by exploiting vulnerability of pdf readers or browser plugins to embed executable code inside documents and web pages.

Once the weaponized deliverable is ready, it must be sent into the target machine by means of an attack vector. This phase is the very begin of the infiltration process, the attacker will use every gathered information to

During delivering, the attacker uses the available intelligence to design the correct method to reach the objective.

A list of commonly used vector is the following [5]:

Email: using a spoofed address and a fake signature, the attacker can play a member of the C-team or a supplier, attaching the weapon to the message.

Message: similar to email, but using some chat-based protocols, for instance Skype, used in many organization.

File Sharing: malicious software may be distributed by means of a sharing system protocol.

Vulnerabilities: internet services, antiviruses, browsers, OSs and other program that expose themselves to the internet, may have vulnerabilities capable of granting access to an attacker. This vector is dangerous because it can be completely undetected by the logging systems.

Passwords: although this can be considered a vulnerability itself, a weak password is a standalone and easy-to-use vector. A weak password can be broken regardless of any update or security system a sysadmin could install.

Another possible attack path is the physical interaction. Using social engineering or disguises, the hacker can try to elude the physical security systems and deploy manually the malware. A common path is also the physical deliver of a memory support, such as an USB stick and the use of social engineering skill to connect it to some host.

2.4.3 Entry point infection and propagation

Once deployed the attacker has an infection to a host that may not be the target. The infection must be spread into the target network, by using automated or manual methods.

2.4.4 Action on Objectives

Once the target is infected, the attacker can take the necessary action to fulfil the objective. Usual actions involve keylogging, data downloading or attacks toward other targets.

Chapter 3

Methodologies for attack prevention, detection and response

ICT security is the result of a competitive evolution between cyber attacks and systems trying to counter them. Taxonomies and process profiling, described in the previous chapter, were designed to understand and identify cyber attacks, in order to create defence systems able to counter them.

As first cyber attacks taxonomies were based on technical vulnerabilities, first defence approaches were guidelines for developers on how to prevent, identify and fix security vulnerability in software and networks and also policy enforcement techniques to prevent unwanted behaviors[7]. However, defence methods based on these taxonomies failed to be effective against complex attacks, which cannot be detected nor prevented by patching single vulnerabilities.

In 1986, the same year of the Morris Worm and of the Howard's taxonomy, Dorothy E. Denning published an intrusion detection model based on real-time monitoring of the system's audit record[8], defining the concept of *intrusion detection and prevention system*. IDPS are widely used nowadays, however they are proving ineffective against targeted attacks[15, 12, 27, 44], and part of the security community is moving toward the concept of assumption of breach[14], which states that a company must be aware that any IDPS can stop a targeted attack, so it may invest in a fast and reliable incident response plan.

The following sections describes the common methodologies used as defence against cyber attacks, which are access control models, firewalls, intrusion detection and prevention systems, intrusion response systems, vulnerability assessment and details on malware analysis.

3.1 Access Control Models

Access control is the process of protecting system resources against unauthorized accesses, by modelling limitations on interaction between *subjects* and *objects*[39]. The act of accessing to a resource may involve the use, read, modification or physical access to it. A subject is authorized to access a resource if a policy that permits it exists.

In the field of computer security, access control manages the interaction of users to network and software resources. Generally speaking, users are mapped to multiple subjects and access control assumes that users are already identified and authenticated[37].

The security community produced multiple models for access control, but only three have been widely recognized and used[37], which are *Discretionary Access Control*, *Mandatory Access Control* and *Role Based Access Control.*

3.1.1 Discretionary Access Control

Discretionary access control (DAC) governs the access of a subject to an object by using a policy determined by the owner of the object.

DAC is suitable for a variety of systems and application for its flexibility, however it doesn't impose any restriction on the usage of objects, that is the flow of information is not managed[37]. For instance, a user capable of reading a file may allow other non authorized users to read a copy of the file without the owner permission.

An example of discretionary access control is the UNIX file modes, where each file has an owner and a group. The owner sets a subset of permissions (read, write and execute) for itself, the group and other users.

3.1.2 Mandatory Access Control

Mandatory access control (MAC) governs the access of a subject to an object by using a policy imposed by the system administrator.

Originally, MAC was tightly coupled to the multi-level security policy[29]. Each object has a security level and a subject a clearance level. Access to objects must satisfy the following relationships[37]:

Read down: a subject's clearance must dominate the security level of the object being read.

Write up: a subject's clearance must be dominated by the security level of the object being written.

This approach imposes some degree of control in the information flow. The write up property imposes that higher-level object cannot be written into lower-level objects.

However the original definition of MAC was insufficient to handle complex situation as it ignores critical properties such as intransitivity and dynamic separation of duty[29]. These problems led to a more generic definition of MAC, where the policies are controlled by the system administrator.

3.1.3 Role Based Access Control

Role Based Access Control (RBAC) governs the access of a subject to an object by means of *roles*, identified in the system. The RBAC model is rich and extendible, because "it provides a valuable level of abstraction to promote security administration at a business enterprise level rather than at the user identity level"[36]. In order to unify ideas from multiple RBAC models and commercial products, the NIST published a unified model in [36], defining RBAC in a four step sequence model. Each step increases the functional capability of the previous step:

1. **Flat RBAC:** users are assigned to roles, permissions are assigned to roles and users acquire permissions by being members of roles.
2. **Hierarchical RBAC:** introduce role hierarchies, with inheritance (activation of a role implies activation of all junior roles) or activation (no activation assumptions) or both.
3. **Constrained RBAC:** imposes a separation of concerns, by limiting the simultaneous activation of certain roles.
4. **Symmetric RBAC:** include the ability of review a permission-role association with respect to a defined user or role.

3.2 Firewalls

A firewall is a network security system, which analyzes the network traffic enforcing policy based on various protocol headers. Packets that do not match policy are rejected. The survey on firewall written by Inghan and Forrest in [19] defines as firewall a machine or a collection of machines between two networks meeting the following criteria:

- The firewall is at the boundary between the two networks.
- All traffic between the two networks must pass through the firewall.
- The firewall has a mechanism to allow some traffic to pass while blocking other traffic (*policy enforcing*).

Firewalls are able to separate a portion of the network, creating a zone of trust, for instance by separating the inner network from the outside network. A trust boundary is needed for security reasons[19]:

- Masking some security problems in OS, by denying the access to internal hosts from the outside.
- Preventing access to certain information present in the outside network, for instance by blocking access to certain web sites.
- Preventing information leaks by analyzing the information contained in the outgoing traffic.
- Logging for audit purposes.

Firewalls can operate as packet filtering or as application layer filtering. Packet filtering firewalls analyze the traffic from the data-link layer to the transport layer of the TCP/IP stack, while application layer firewalls enforce policies at the application layer.

3.2.1 Packet Filters Firewall

Packet filters firewalls use a set of rules to decide whether or not to accept a packet. Packet filters operate up to the transport layer of the ISO/OSI stack and make decisions with rules based on a subset of the following information[19]:

- Source/Destination interface.
- Whether the packet is inbound or outbound.
- Source/destination address.
- Options in the network header.
- Transport-level protocol.
- Flags/options in the transport header.
- Source/destination port or equivalent (if the protocol has such construct).

Packet filters can also perform a better filtering at the transport level, by keeping the *state* of the connection. This kind of firewalls is called stateful filter. For instance, a stateful firewall can keep track of established TCP connection and some ICMP communication[19], such as the “echo request”.

Packet filters have high performance and doesn’t requires user interactions, however they are incapable of mapping the network traffic to a user[19]. They can only map the traffic with the host IP, which can lead to the bad practice of using IP addresses and DNS names for access control.

3.2.2 Application Layer Firewall

An application layer firewall works like a packet filter firewall, but operates in the application layer of the ISO/OSI stack. These firewalls are able to interpret various application protocols, such as FTP, HTTP or DNS.

There are two categories of application layer firewalls:

Network-based firewall: the firewall is a computer system inside the network, which operates as proxy server, or reverse proxy for various applications/services. A typical example is a web application firewall (WAF).

Host-based firewall: the firewall operates over the network stack, monitoring communication directly made by the application. Unlike network-based firewalls, a host-based firewall can filter per-process packets, but it can protect only the host were it is installed.

3.3 Intrusion detection and prevention systems

Intrusion detection is the process of monitoring a set of resources, such as computer systems or networks, and analyzing them to detect intrusions[38]. Intrusion prevention is the ability of stopping ongoing attacks or preventing them from happening, by modifying the attack context.

Intrusion detection and prevention systems (IDPS) are software or hardware system capable of detecting and preventing intrusions, which are attempts to compromise the confidentiality, integrity and availability, or to bypass the security mechanism of a computer or network[38, 28]. IDPS, often referred as IPS, are a superset of *Intrusion Detection Systems* (IDS), which offer only the detection capability. The following sections, however, will focus on IDS and how to classify them.

3.3.1 Intrusion detection model

The intrusion detection model proposed by Denning's in 1986 is independent of system, application, vulnerability and intrusion. It defines the objects, used by subjects under a particular behavior. Any action, expected or unexpected, is recorded.

Subjects: initiators of activity on a target system.

Objects: resources managed by the system, e.g. files, commands, devices, etc...

Audit records: generated by the target system in response to actions performed or attempted by subjects on objects. Each audit must have a timestamp.

Profiles: structures that characterize the behavior of subject with respect to objects in terms of statistical metrics and models of observed activity. Profiles are automatically generated and initialized from templates.

Anomaly records: generated when abnormal behavior is detected.

Activity rules: actions taken when some condition is satisfied, which update profiles, detect abnormal behavior, relates anomalies to suspected intrusion and produce reports.

Although the model is designed for a intrusion detection expert systems[8], it can used to describe others IDS. However, the security community didn't create a standard model or ontology for IDS, resulting in the commercialization of various products with different features and capability. Thus, the security researchers tried to find a comprehensive taxonomy for the existing IDS.

3.3.2 Taxonomy for intrusion detection systems

The production of taxonomies for intrusion detection systems had a burst in the 2000s. Many papers propose classifications based on methodology and technology, and other papers were published as surveys and reviews of existing taxonomies. In an attempt to organize issues and classes, Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin and Kuang-Yuan Tung published a comprehensive review of IDS in 2012[28]. The following taxonomy is the result of their work.

Intrusion detection systems has four characteristic, each of them has multiple viewpoints. An IDS classification is performed by identifying every viewpoint of each characteristic.

- **System Deployment:** defines on which technology the IDS is designed to work. Thus, it determinates which *objects* can be monitored,

- **Data source:** defines which kinds of data are used and how they are gathered. Data is the base of *audit records*.

- **Timeliness:** define the IDS detection granularity and how it reacts. It's a particular aspect of *activity rules*.

- **Detection Strategy:** defines methodologies and strategies for detection and processing. It defines *profiles*, *activity rules* and how to generate *anomaly records*.

System Deployment

System deployment concerns with what the system is supposed to monitor and how it's interconnected with the infrastructure. The deployment consists of three different viewpoints: *technology type*, *networking type* and *network architecture*.

The *technology type* determinate what events the IDS can collect and inspect. There are four classes of *technology type*[28]:

- **Host-based IDS (HIDS)**: monitors and collects information from hosts and servers using *agents*.
- **Network-based IDS (NIDS)**: captures network traffic of specific network segments through *sensors* and analyzes protocols and communication of applications.
- **Wireless-based IDS (WIDS)**: similar to a NIDS, but sensors capture traffic from wireless domain.
- **Network Behavior Analysis (NBA)**: works like an NIDS, but inspect the traffic to identify attacks with unexpected traffic flows.
- **Mixed IDS (MIDS)**: is an IDS with multiple technologies.

With *network architecture*, an IDS is classified basing on how it detects attacks[28]:

- **Centralized:** the IDS detects attack basing on data from a single monitored system.
- **Distributed:** the IDS is capable of detect structured attacks that involve multiple systems. The distributed approach can be *parallelized*, *grid-based* or *cloud-based*.
- **Hybrid:** the IDS uses multiple approaches.

Finally, the *networking type* identifies how the IDS is interconnected with the system[28], which can be *wired*, *wireless* or *mixed*.

Data Source

Data source determinates how the IDS gather data and which data it can handle. An IDS collect data by means of *collection components*, which can be software based (*agents*) or hardware based (*sensors*).

Data from components can be *collected* in a centralized or distributed fashion, depending on the IDS has a single server which gathers all data or it has intermediate gatherers.

The last viewpoint is the *data type*, strongly related to the *technology type*:

- **Host logs:** related to the host activities, they include *system commands*, *system accounting*, *system logs* and *security logs.*
- **Application logs:** logs from a single application.
- **Wireless network traffic:** includes all the traffic gathered in the wireless domain.
- **Network traffic:** in addition to network packets, it includes also SNMP information.

Timeliness

Timeliness determinates how the IDS behaves in the time dimension:

- **Time of Detection:** can be "*real time/on-line*" or "*non-real time/off-line*".
- **Time granularity:** describes when the IDS process the data for sign of attacks. It can be *continuous*, *periodic* or *batch.*
- **Detection-response:** determinates if the IDS is also an IDPS. It can be *passive*, that is the IDS take no countermeasures against the detected attack or *active* if the IDS tries to prevent the attack.

Detection Strategy

Once the data is gathered, the *detection strategies* determinate how the information is produced. The *detection discipline* viewpoint establishes how the state of insecureness is handled; if an IDS recognize two states (secure and insecure), the IDS is called *state based*, while if it recognize the transition from secure to insecure, and vice versa, the IDS is called *transition based.* Both state and transition based IDS can evaluate the state using an *non-obstructive* or *stimulating* evaluation, depending on the IDS infers the system status by the passive monitoring of the systems or if it obtains other data by interacting with the system.

The *processing strategy* viewpoint, as for the *data collection* viewpoint of *data source*, can be centralized or distributed.

Finally, the *detection methodology* is one of the most important aspect, because it determinates how the IDS recognize a threat [28]. There are three possible methodologies:

- **Signature-based:** the IDS search the logs for known strings or pattern associated with a threat. It is also known as *Knowledge-based detection* or *misuse detection.*

- **Anomaly-based:** the IDS detects anomalies, which are deviations from a known behavior (profile). A profile can be either static or dynamic. It is also known as *behavior-based detection*.

- **Specification-based:** the IDS understands various protocols and keep track of their state, searching from anomalous behaviors. It's called *stateful protocol analysis* and differs from anomaly-based detection, because the protocol profiles are generic and low-level.

3.3.3 IDS comparison

Intrusion detection systems are often specialized in particular field and technologies, thus comparison between them results in a pro-cons list. However, knowing the strength and limitation of different IDS is necessary to build a complete defence system. The major comparison are done basing on *technology type* (see table 3.2) and *detection methodologies* (see table 3.1)[28].

Signature-based	
Pros	• Simple and effective against known attacks. • Detail contextual analysis.
Cons	• Ineffective against unknown attacks or variants of known attacks. • Little understanding of states and protocols. • Need to keep signatures and patterns up to date.
Anomaly-based	
Pros	• Effective against new and unforeseen vulnerabilities. • Less dependent on OS. • Facilitate detection of privilege abuse.
Cons	• Weak profiles accuracy due to observed events being constantly changed. • Difficult to trigger alerts in right time.
Specification-based	
Pros	• know and trace the protocol states. • Distinguish unexpected sequence of commands.
Cons	• Resource consuming due to protocol state tracing and examination. • Unable to detect attacks conforming to the protocol. • Too much dependent on OSs or APs.

Table 3.1: Pros and cons of intrusion detection methodologies[28]

HIDS	
Pros	• Can analyze end-to-end encrypted communication.
Cons	• Lack of context knowledge results in difficult detection accuracy. • Delays in alert generation and centralized reporting. • Consume host resources. • May conflict with existing security controls.
NIDS	
Pros	• Can analyze the broadest scopes of AP protocols.
Cons	• Cannot monitor wireless protocol. • High false positive and false negative rates. • Cannot passively detect attacks within encrypted traffic. • Difficult analysis under high loads.
WIDS	
Pros	• Narrow focus allows a more accurate detection. • Only technology able to supervise wireless protocol activities.
Cons	• Cannot monitor application, transmission and network level protocol activities. • Physical jamming attacks against sensors are possible.
NBA	
Pros	• Able to detect reconnaissance scanning. • Can reconstruct malware infections. • Can reconstruct DoS attacks.
Cons	• The transferring flow data to NBA in batches causes a delay in attacks detection.

Table 3.2: Pros and cons of IDS technology types[28]

3.3.4 Intrusion prevention capabilities

The previous taxonomies was focused on intrusion detection, because intrusion prevention is strongly related to the detection and may be seen as additional capabilities, rather than a factor for further classification[28, 38].

The NIST guide to intrusion detection and prevention systems[38], describe the prevention capabilities for each IDS technology type class.

Network-based IDPS and **network behavior analysis IDPS** operates in similar ways, by altering the network communication. Such IDPSs can:

- End the current TCP session (session sniping).

- Perform inline firewalling by dropping or rejecting suspicious network activity.
- Throttle bandwidth usage, in case of DoS attacks, file sharing or malware distribution.
- Alter malicious content, by active sanitization of the payload.
- Reconfigure other network security devices, such as firewalls, routers and switches to block certain activities or route them toward a decoy or also to quarantine an infected system.
- Trigger a response plan (see section 3.4).

A **wireless-based IDPS** operated directly by terminating the malicious wireless session and preventing a new connection to be established, or, if a wired link is present, The WIDPS can instruct router and switches connected to the access point to block the suspicious network activity.

A **host-based IDPS** agent operates directly on the host, thus it can perform multiple prevention actions:

- Prevent malicious code to be executed.
- Prevent some application to invoke certain processes, for instance to prevent network application to execute shells.
- Prevent network traffic to be processed by the host.
- Prevent suspicious outgoing traffic to be sent to the network.
- Reconfigure the local firewall.
- Dynamic policy enforcement.
- Prevent access, read or usage to some files, which could stop malware installation.

3.4 Intrusion Response Systems

Intrusion detection systems are in continuous development, and new techniques for detection and prevention are always refined. However, in case of attack detection, an IDS doesn't offers a full support for an automated counter-measure to the attack[43].

From an high-level prospective, the process of intrusion response involve the following actions[13, 30]:

- Contain the effect of an attack, in case of a multi-staged attack or an APT.

- Recover affected services.
- Take longer term actions, such as reconfiguration or patching, to prevent further attacks.
- Gather and handle evidences for law enforcement.
- Minimize the effect of information leaks.

The process of response to an ongoing or concluded attack is a high heterogeneous, system dependent problem and received less attention from the security community[43, 13, 14]. However, the increasing complexity of systems and attacks, made the manual intrusion response inadequate and the cost necessary to maintain an effective team for incident response raised.

An intrusion response system (IRS) is a system capable of automatize and ease the incident response steps. In 2007, Stakhanova proposed a taxonomy for IRS[43] to provide a foundation for further works on intrusion response.

3.4.1 Taxonomy for Intrusion Response System

Intrusion response systems can be classified according to the *activity of triggered response* and to the *degree of automation*[43].

Activity of Triggered Response

The activity of triggered response determinates how an IRS responds when a new attack is detected. The response can be passive or active, according to the IRS notifies the system administrator and provides attack information or if the IRS attempts to contain the damage or to locate or harm the attacker. Common passive responses are:

- Notify the administrator by generating an alarm.
- Generate a comprehensive report of the attack.
- Enable additional IDS or logging activity.
- Enable forensic and information gathering tools.

An active response can operates on hosts or network. Host-based response actions can be:

- Deny access to files.
- Allow to operate on fake files.
- Restore or delete infected files.

- Restrict user activity or disable the user account.
- Shutdown or disable compromised services or host.
- Terminate suspicious process.
- Abort or delay suspicious system calls.

While network-based response actions can be:

- Enable or disable additional firewall rules.
- Block suspicious incoming/outgoing network connection.
- Block ports or IP addresses
- Trace connection to perform attacker isolation or quarantine
- Create a remote decoy or honeypot.

Degree of automation

Basing on the degree of automation an IRS can be categorized as:

Notification system: provides information about the intrusion. The system administrator has the duty of perform and intrusion response basing on the notification information.

Manual response system: provides a higher level of automation, by presenting the attack log and a set of pre-determined responses to be launched by the administrator.

Automatic response system: opposed to the manual response system, the automatic response system provides immediate response through an automated decision-making process.

Automatic response support is very limited, but many IRS offers high level of automation. Thus the automatic response systems can be further classified by the *ability to adjust*, the *time instance of the response*, the *cooperation ability* and by the *response selection mechanism.*

The *ability to adjust* determinates if an IRS is capable of dynamically adapting the response selection to the changing environment during the attack time. A *static* approach it's simple and easy to maintain but requires periodical upgrades and the true effectiveness of the response plan is truly tested only during an attack. On the other hand, an *adaptive* approach can operates in many ways, for instance by adjusting the system resource devoted to intrusion response or by reasoning on previous success or failure responses.

The *time instance of the response* classifies an IRS basing on when the response plan is activated basing on the attack time. A *proactive* approach tries to foresee the incoming intrusion before the attack has effected the resource. A *delayed* approach applies the response plan once the attack has been confirmed. Although a proactive system can execute more effective response actions, its design is complex and a high false rate of the attack foreseen can trigger inappropriate responses.

The *cooperation capability* relates a set of IRS by their intercommunication. A *cooperative* IRS, in contrast with an *autonomous* IRS, can cooperate with other systems to provide a global detection and response.

The last characteristic is the *response selection mechanism*, which determinates how the IRS selects the response plan basing on the alarm and the environment conditions. A *static mapping* may be seen as an automated manual response system: the system administrator maps a response plan to certain alarms and the IRS automates the response actions. However this approach seems to be infeasible for large systems with vast and complex threat scenarios. A *dynamic mapping*, instead, takes into account also custom metrics, such as confidence or severity of attack. The response plan is then formulated at real-time basing on both alarms and environment. Finally, a more business oriented approach is the *cost-sensitive mapping*, which determinates the optimal response basing on the balance between the intrusion damage and response cost.

3.5 Vulnerability Assessment

Vulnerability are the gateways by which threads are manifested[4], in other words, vulnerability are attack paths for both opportunistic and targeted attackers. The vulnerability assessment is the process of identifying, quantifying and prioritizing the vulnerabilities in a system.

Vulnerability assessment should be performed for external and internal network reachable systems and singular hosts, in order to protect against remote attack and malware spreading and infection. Possible vulnerabilities that can be encountered can be:

- **Application misconfiguration -** for instance by allowing unauthenticated users to perform write actions or by permitting the execution of external applications.
- **Unpatched system -** system vulnerabilities are published by means of bugs and *CVE* (Common Vulnerabilities and Exposures) entries. Once a vulnerability has been exposed the system is to consider vulnerable until a new patch is released and applied.
- **Default configuration -** many applications come with a default configuration, such as passwords, opened ports or seeds, which can be

easily exploited.

The action of securing the company systems doesn't scale well with its size and complexity, thus the vulnerability assessment should be expanded in an organized vulnerability management process able to control information security risk[33]. An example of vulnerability management process is[33]:

1. **Preparation:** the process starts by organizing vulnerability scans into meaningful and limited chunk of systems. The preparation should also determinate which kind of scan to employ (i.e. external, internal or host-based).

2. **Initial vulnerability scan:** during this phase, the scans defined in the preparation phase are executed. The result is a list of vulnerabilities.

3. **Remediation phase:** after the vulnerabilities are obtained, each one of them must be quantified and prioritized for remediation. The feasibility, the risk and the amount of work required depends on the type and rank of the vulnerability, the kind of the affected system/application and by which corrective measure is required (e.g. patch or reconfiguration?). The result of this phase is a work plan.

4. **Implement remediating actions:** the remediation work plan is applied. If a corrective action fails, alternatives must be discussed.

5. **Rescan:** a second scan is performed to test the mitigation plan. If a mitigation action wasn't successful, new corrective actions must be discussed and applied.

3.6 Malware analysis

Executables and active documents can exhibit malicious behaviors, threatening the safety of the system (see section 2.1). Defence systems try to detect malware by performing an automatic analysis.

The first approach is the *static analysis*, which analyzes the malware without executing it. From static analysis, information such as strings, call graphs, parameters value and known signature can be extracted[10], however this approach can be drastically complicated by packing, self-modifying code, ciphers and code disallignment.

Therefore, the security community is continuously developing advanced methods to analyze malware actions while executing the malware itself[10]. This approach is called *dynamic analysis*.

3.6.1 Dynamic analysis information

The dynamic analysis detects malicious software by observing the malware behavior and detecting malicious actions. Malicious actions are identified on how the malware interact with the environment, thus many activities can be observed:

Filesystem access - Malware usually drop other stages or modified version of them, by writing executable on the disk. They can also access configuration files or documents for discovering vulnerabilities or stealing data.

Network activity analysis - The majority of malware communicates with remote C&C or tries to scan the network to spread the infection. By analyzing the network communication, the analyst can obtain information on remote location, protocol and spreading methods.

Process interactions - A malware can interact with other process to hide its execution, self delete the executable or exploit a running process.

Inter-process synchronization - Inter-process synchronization requires the creation of files or named mutex. Although not a malicious action by itself, it can provide additional information.

Registry/System configuration access - The system configuration can be read or written for various reasons. Example of suspicious behavior can be to search information about installed programs, such as the antivirus, or insert the executable inside the autostart list.

3.6.2 Dynamic analysis techniques

Information about malware behaviors can be obtained using multiple approaches. This section describes techniques on how to gather such information.

System Calls Monitoring

In order to act on the system, a malware needs to access the system calls of the OS. Windows, for instance, offers a set of *API* (e.g. functions in `kernel32.dll`), a set of *native API* and *system calls*, which are direct invocation of the kernel. The native API is a bridge between the API and the system calls.

By analyzing the sequence of API calls and their parameters, the analysis system can track the malware behavior. The call monitoring is performed by means of function call hooks[10], which can be implemented using:

- **Binary rewriting:** binary rewriting consists in rewriting the first byte of every called code with a `jmp` to a hook function. The hook function must re-implements the overwritten instructions and returns to the original code[17].
- **Debugging:** a program can debug the malware inserting breakpoints at every call.
- **Dynamic linked libraries replacing:** the DLL used by the malware can be replaced with custom DLL simulating the expected behavior.

Information Flow Tracking

The tracking on the information flow is an orthogonal approach to the monitoring of the system calls[10]. The idea is to mark (taint) a memory location with a label and follow its usage and relationships during the execution.

The information flow tracking uses the following concepts[10]:

- **Taint source:** introduces new labels into the system.
- **Taint sink:** react to tainted inputs.
- **Direct data dependency:** dependency created after a direct assignment or arithmetic operations (see figure 3.1).
- **Address dependency:** created when a tainted value is used as address for memory read or write operations (see figure 3.2).
- **Control flow dependency:** conditional assignments of a variable based on tainted values, causes a control flow dependency to be created (see figure 3.3).

Pure dynamic information flow tracking can be easily evaded by using a semantic dependency[10] (see figure 3.4). The value of `x` influences the value of `v` with an implicit dependency, which can be discovered only by analyzing the unexplored branch. This can be done with branch static analysis, or using multipath techniques[10].

```
// x is tainted
a = x;
b = x * 3;

// Know both a and b are tainted and dependent with x

```

Figure 3.1: Direct data dependency

```
// x and y are tainted

a = x[0];
// Know a is tainted and dependent with x

b = v[y];
// Know b is tainted and dependent with y

```

Figure 3.2: Address dependency

```
// x is tainted
if (x == 0)
  a = 1;
else
  a = 0;

// Now a is tainted and dependent with x

```

Figure 3.3: Control flow dependency

```
// x is tainted
a = 0; b = 0;
if (x == 1) a = 1; else b = 1;
if (a == 0) v = 0;
if (b == 0) v = 1;

// If x is 1, a is tainted and dependent with x but v not
// If x is 0, b is tainted and dependent with x but v not

```

Figure 3.4: Tainting evasion

Multipath analysis

Multipath analysis is an extension of information flow tracking, proposed by Moser in 2007[31]. It permits to analyze multiple execution paths by recording dependency and constraints between memory locations and forcing a new consistent state at each branch.

Usually the *tainting sources* are values obtained from the environment, such as network communication, date or files. When registering a dependency between a tainted value and a new memory location, the system also registers the constraints between the values. When the program reaches a conditional jump based on a tainted value, the system tries to force each branch. The consistency of the variables is obtained by solving the constraint problem. This approach is expensive; in fact Moser system allows only linear constraints.

Multipath analysis allows bypassing logic bombs triggers, by forcing the conditions and running the malicious code.

3.6.3 Dynamic analysis implementation

Malware analysis techniques can be implemented in various environments[10]:

Analysis in User-space/Kernel-space - The malware is analyzed in user space or kernel space, gathering information such as invoked functions, API calls and memory values. However, it's difficult to analyze the information flow and an advanced malware could be able to detect or evade the analysis program.

Memory and CPU emulation - The malware is launched into a sandbox, preventing any side-effects on the system. Each instruction is analyzed and performed into a virtual environment; therefore the analysis system must simulate all the mechanisms offered by the operating system. A missing or bad emulated component can be exploited from the malware to evade the sandbox.

Full system emulation - The analysis system emulates a complete environment with CPU, memory, peripherals, network interfaces and an operating system. This approach permits the emulation of different architecture as well, for instance ARM. The analyst has a physical level access to the system and the malware must find flaw into the CPU emulation (commonly referred as *red pills*) to detect the emulation.

Virtual Machine - A virtual machine differs from an emulated system because the host and the guest architecture are identical and the VM allows the guest system to execute non privileged instructions. This feature results in a performance improvement.

Another important aspect is the network simulation[10]. The analysis system could simulate a network with basic services (DNS, mail, IRC, FTP, etc...) and redirect every request to them. This method is safer, but it cannot deal with custom protocols nor can simulate valid message such as the C&C commands exchange. Another solution is to allow a filtered access to the internet, by limiting the network traffic and spreading. This approach allows the malware to receive commands from the C&C or to enable additional hidden behaviors, by triggering the download of other malware components.

Chapter 4

Defence strategies against targeted attacks

In chapter 3 I mentioned that intrusion detection and prevention systems are proving to be ineffective against targeted attacks and that companies are investing in intrusion response plans. Intrusion Response Systems offers an improved level of protection that may be suitable for targeted attacks, in particular to advanced persistent threat, because they made the legit assumption that an attack needs time to be performed and thwarting it during the early stages could contain the damage. However, also automatic intrusion response systems rely on attack detection, otherwise the response plan will never start.

Many recent papers claim that targeted attacks are hard to detect using conventional defence systems[15, 12, 27, 44, 14]. In some cases, the breach occurred long before its discovery and the investigation involved a difficult forensic process.

But are targeted attacks really capable of bypass detection? To answer this question, Ramilli Marco, Mella Luca and I performed a targeted attack against `ACME Corporation` to prove the ineffectiveness of conventional defence systems.

In section 4.1 I describe the targeted attack we performed. In section 4.2, I analyze the results of the attack, in particular why the defence systems failed to detect it. Finally, in section 4.3 and 4.4 I describe the developed defence approach against targeted attacks, while in section **??** I list some tools to be used as support.

4.1 A real targeted attack

The targeted attack was aimed to bypass the security systems and infect the executives' personal computers with a malware. Once inside, each malware had to scan the pc for project files and log the keys pressed. Table 4.1

describes the attack process using the kill chain model and table 4.1 classifies the attack according to the *AVOIDIT* taxonomy.

The attack started with an information gathering phase. The objective was to learn:

- Which antivirus is installed or is likely to be installed in the hosts.
- If and which intrusion detection system guards the perimeter.
- External website vulnerabilities.
- Involved person, clients.
- Email addresses and communication format.
- Physical defence.
- Policy for physical access.

By using scrapers, web scans and some social engineering, we discovered a few information about the network and the IDS, but we failed to gather information about the antivirus. Then, basing on the available information, we designed a malware composed by four stages:

1. *The first stage* is a dropper, which serves as a social engineering entry point, by fooling the user to execute it. Once started, the first stage works drops the second stage and execute it.
2. *The second stage* implements stealth and evasion techniques, to hide itself from users and defence system. The main evasion is realized with a Logic Bomb: the second stage has an encrypted third stage, which is decrypted when the trigger is asynchronously activated by the attacker. Once the third stage is decrypted, it's executed.
3. *The third stage* is designed to be injected into other processes, as part of exploits or remote execution, however our malware didn't perform any exploitation and directly execute the third stage. The third stage contacts the C&C and downloads the final stage.
4. *The fourth stage* installs a backdoor on the host, allowing the attacker to execute commands, keylogging and to download or upload files.

Our malware is exposed against the antivirus only for the first stage and second stage, because they are saved the disk, while other stages resides only in memory. The malware was new and unknown, so no signature existed and the only possible detection would be by suspicious behavior. Thus, we designed and tested the second stage so that it won't rise any alarm with the majority of antivirus systems both when it waits for the trigger and when the fourth stage is downloaded.

To further bypass the analysis of both fourth stage and communications, we used an HTTPS connection with the C&C. Unless the system performed TSL inspection, and according to the information gathering phase it didn't, any encrypted communication couldn't be analyzed.

Reconnaissance	Web scan, physical presence, social engineering interactions via mails and in person.
Weaponization	Executable archive with real files and malware.
Delivery	Physical deliver or drive-by-download of the malware. Trigger activated via email.
Exploitation	None.
Installation	User-space RAT installation.
Command&Control	Immediate connection to the remote C&C using HTTPS.
Actions on Objectives	File downloads, file upload, keylogging.

Table 4.1: Attack description using the KillChain model (section 2.3.4)

Attack Vector	Social Engineering
Operational Impact	Installed Malware, Trojan
Defence - Mitigation	Remove from network
Defence - Remediation	Computer security awareness course for personnel
Informational Impact	Disclosure
Target	User

Table 4.2: Attack classification using the AVOIDIT taxonomy (section 2.3.3)

4.2 Why did defence systems fail?

When we performed the first tests and then the attack, we were rather surprised. The malware bypassed advanced defence systems, without the use of any exploitation.

The answer to *why did defence systems fail?* is complex and involve several aspect, so I broke the question down into the following subquestion, one for each attack stage:

- **Why wasn't the recon phase detected?** Web scans may results as burst of accesses to the website. It's difficult to discriminate normal traffic from reconnaissance. Moreover the scraping on external websites cannot be detected.

- **Why wasn't the malware detected?** The NIDS checked the downloaded executable using automatic behavioral analysis and signatures detection, same for the antivirus on the host. The malware didn't have known signatures, and it was designed to be multi-staged: the malicious actions won't begin until a condition is triggered by the attacker.

- **Why was the malware executed?** Using advanced social engineering, untrained personnel can be lured in executing the malware.

- **Why weren't the C&C communication detected?** The malware used an HTTPS connection toward a website with unknown reputation. No SSL proxy was used, thus encrypted communication cannot be examined. However, basing on further tests, even if TSL inspection were present, a custom http-based protocol would have evaded the signature analysis on payload and communication.

Although the questions covered different aspect of the attack, three major causes can be identified:

- **Signature detection:** detection by signature can be very efficient and effective against opportunistic attacks, but targeted attacks use custom and unknown malware, for which there is no signature.

- **Failed to inspect TSL/SSL connection:** targeted attacks have to manage few connections, so they can develop and use less efficient protocols which are harder to detect. Modern IDS can inspect SSL connection, but it's not enabled by default because it introduces network delay and problems with certificate recognition.

- **High level of social engineering:** emails and communication are studied to be realistic, it's easy to perceive them as real.

After performing the attack, I analyzed common attack process and defence systems to answer the question: can targeted attack be prevented or detected?

Prevention against targeted attack is hard to employ. Our attack, for instance, didn't perform any exploitation and passed undetected through many active defence systems. If real time detection fails, the prevention cannot act. Moreover, our malware compensated the absence of system exploitation with "human exploitation". Although a refresher course for employees about computer security and social engineer, may have compromise the attack, other entry points, such as via BYOD[1], are harder to control.

[1]Bring your own device

Since prevention requires detection, my work focused on how to detect a targeted attack, or at least on how to increase the change of detection. A design of an automatic real-time protection requires a different data set and a different work and, more important, it couldn't be immediately deployed inside `ACME Corporation`. Thus I have identified some key aspect that can be used by an analyst to pinpoint possible targeted attacks.

Since the strategy had to be deployed inside a enterprise context, I had to consider many problems:

- **Legacy systems:** some systems cannot be upgraded, nor host defence systems can be installed on them.
- **Number of executables and pdf docs:** executables and pdf documents may be malicious, however big enterprises downloads hundreds of them every day. Manual analysis of each file is unfeasible.
- **Presence of external devices:** many devices and even servers can reside inside the private network but not be under the direct control of the enterprise. The rest of the network must be protected from them and vice versa.
- **Network and host resources:** defence systems may not use excessive bandwidth or host resources.
- **Defence against opportunistic attacks:** targeted attacks may be an higher threat, but they are in lower number than opportunistic attacks, which still remain a constant menace.
- **Prevention, detection and response cost:** a security problem can be seen as a ticket. It must be opened, described and quantified with a risk evaluation value, so that the administration team can prioritize the cleanup operations.

After identifying some content, the approach evolved into a conceptual framework for targeted attack detection in synergy with a business process: targeted attacks are too complex and context-dependent, thus also the defence should be complex and context-dependent. My approach is not another defence system, but rather a service.

In the following sections I will refer as *organization* the teams resposible of the targeted attacks detection and as *company* the monitored client.

4.3 WASTE: Warning Automatic System for Targeted Events

`WASTE` is a framework which is capable of rising warnings by analyzing events generated by existing defence systems. The core idea of `WASTE` is that defence systems are capable of recording and analyzing events related to targeted attacks, however they fail to automatically mark them as malicious. However, if such events could be marked as suspicious, meaning that an expert judgment is required, the framework generates a *warning* for an analyst team (composed by human beings) that analyzes the events to infer what an automatic system cannot do. As a consequence, `WASTE` itself cannot be an install-and-forget system. It requires constant monitoring and improvement, to create, improve or delete *detection methods*, which are extremely dependent from the deployment context.

The design of `WASTE` changes from company to company. Requirements, scenarios and policies are hard to predict and generalize and I leak of the required experience. Thus, I propose `WASTE` as a conceptual framework, by modeling the core ideas only.

Figure 4.1 describe the core use cases of `WASTE`. The framework has three main actors:

- **Analyst Team:** responsible for the analysis of *warnings* and the proposal of new *detection methods*.
- **Detection Team:** fix the issue of `WASTE` and manage the *detection methods*.
- **Defence System:** already existing automatic defence system to be inquired.

Use cases describe what is expected from the system:

1. **Propose detection method:** the analyst team can propose detection methods basing on their experience and the detection team will receive the proposal.
2. **Notify Warnings:** The analyst team is interested in receiving the warnings generated by the system.
3. **Recover related events:** Given a warning, the analyst team must recover which events are correlated.
4. **Examine defence system:** the defence system is examined to detect new warnings.
5. **Mantain WASTE:** check the system functionality and fix issues.
6. **Manage WASTE:** the detection team manage the detection methods inside WASTE, by improving, creating, removing or disabling detection methods.

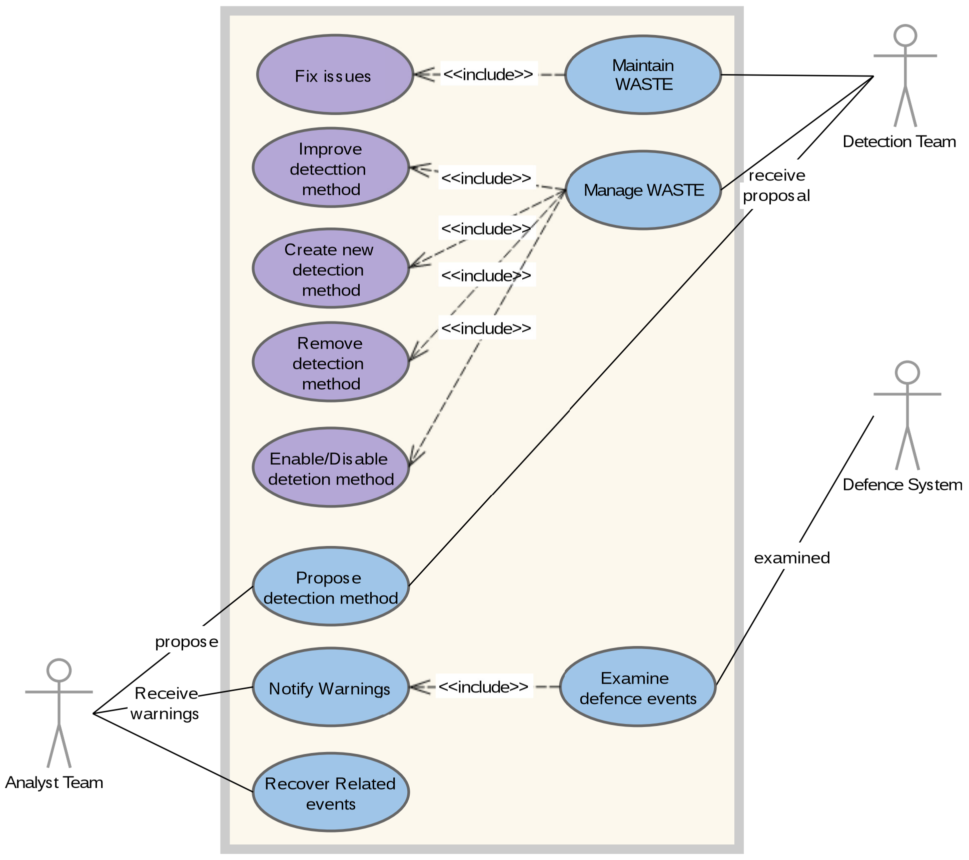


Figure 4.1: `WASTE` use cases

From the use cases I propose a core domain model, to better formalize the requirements. Figure 4.2 shows the core system entities, figure 4.3 shows the interaction of the detector, figure 4.4 the interaction of the detection team with the system to fulfil the use case *Manage WASTE*, figure 4.5 shows the interaction between the two teams to propose new detection methods. Finally, figure 4.6 shows the behavior of the detector.

The core aspects of `WASTE` are the detectors. To write good detectors permits to obtain meaningful warnings, thus rising the possibility of detecting a targeted attack. Approaches for detectors can be inspired to the cyber attacks models described in chapter 2 or defence approaches taxonomies of chapter 3.

I used some concepts of the kill chain model (section 2.3.4) and the modus operandi (section 2.3.5), basing on our attacks and those described in other papers and white papers[18, 27, 44, 11], to identify some key aspects for detecting targeted attacks.

In particular I have considered email-driven attacks (section 4.3.1), executable and document analysis (section 4.3.2) and network communications

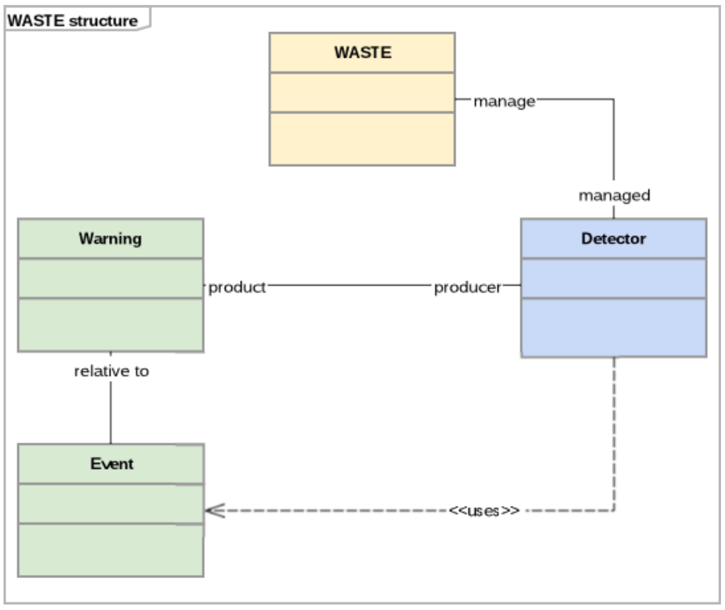


Figure 4.2: `WASTE` domain model: structure

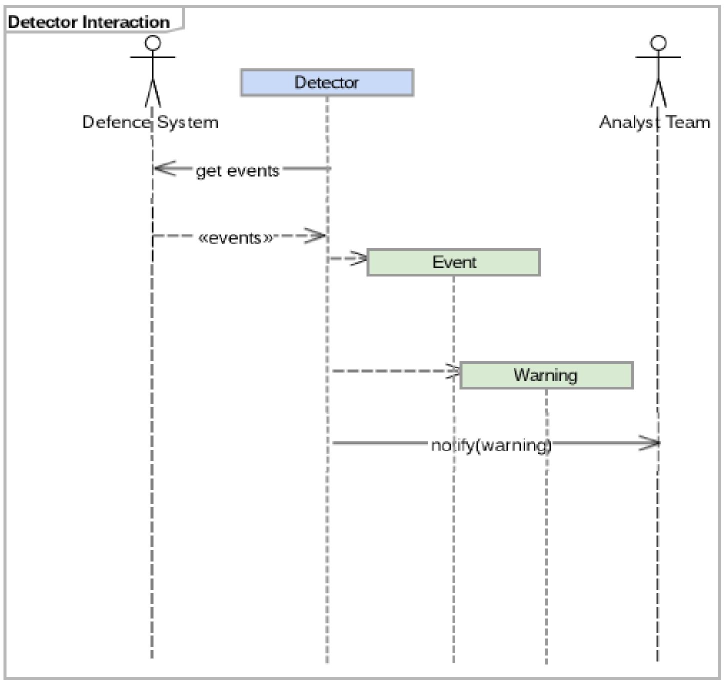


Figure 4.3: `WASTE` domain model: detector interaction. Fulfil use cases 2, 3 and 4

(section 4.3.3).

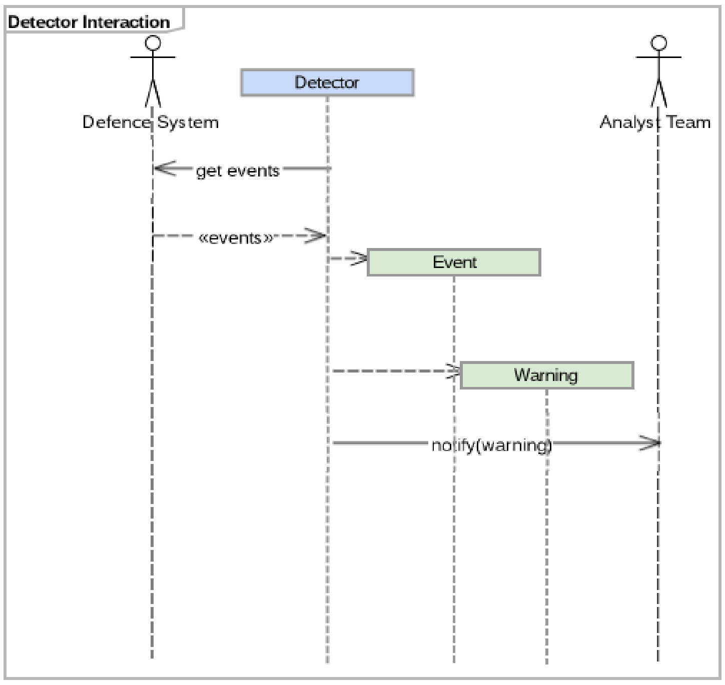


Figure 4.4: `WASTE` domain model: WASTE interaction. Fulfil use case 6

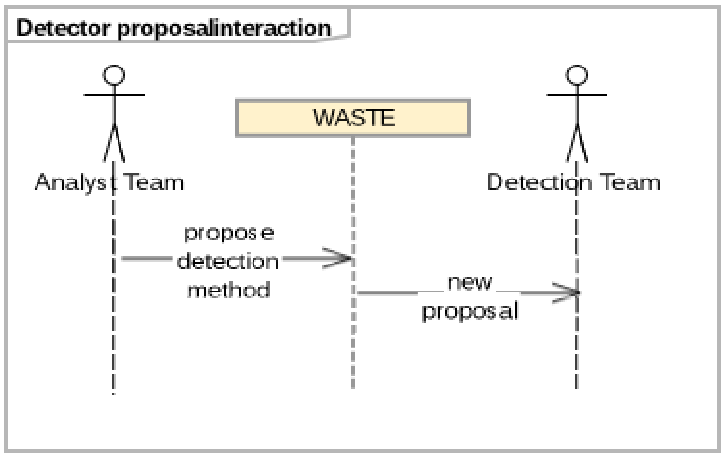


Figure 4.5: `WASTE` domain model: proposal interaction. Fulfil use case 1

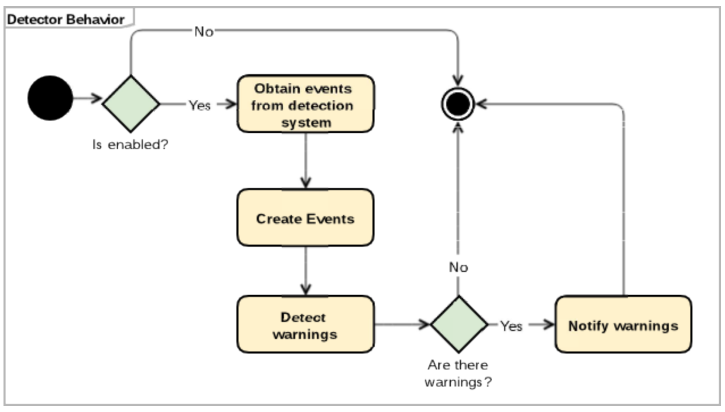


Figure 4.6: `WASTE` domain model: detector behavior

4.3.1 Email-driven attacks detection

According to the Verizon data breach report[11], emails attack vector for malware spreading was used in more than fifty percent of the attacks. Also our attack and those described in [18] and [27] used email as vector.

An email attack can occur in three different ways:

1. **Spoofing -** Spoofing is a social engineering technique, the attacker spoof the sender address with a valid email address. He cannot receive any response, but he has a better chance to fool the receiver.

2. **Similar Domain -** Another social engineering technique. The attacker uses a domain that is similar to an allowed name, for instance by sending a mail from `@acrne.com`, instead of `@acme.com` [9]. The domain can be either spoofed or registered. The latter case permits the attacker to receive responses.

3. **External Domain -** An attacker could just use a valid external domain, such as `gmail.com` or `yahoo.com`, as it happens in [18]. The social engineering attack resides in the message content.

Spoofed and emails with similar domain can have a great impact if undetected. Figure 4.7 shows an email header sent from `Daniele_Bellavista@acme.it` to `target@gmail.com`, while figure 4.8 shows a similar email with same sender and destination but it wasn't sent from `@acme.it`, but using a *postfix* mail server.

GMail identified the spoofed email as real, because the spam filters didn't match any signatures and the source IP is not blacklisted. However, my approaches assume the presence of an analyst team, which is alerted when suspicious activities occurs and can act according.

Real mail and fake mail, besides some missing or additional headers, can be distinguished by the `Received: from` header. The real email has (real IPs have been masked):

```
Received: from mail.acme.it ([192.168.66.55])
```

The spoofed email is very likely to have a different IP address:

```
Received: from mail.acme.it ([192.168.100.100])
```

When receiving an email, the analyst team can perform three automatic checks before beginning the manual analysis:

- **Reverse DNS query -** the displayed hostname usually differs from the domain name bound to the IP address. However if the email isn't spoofed, it's very likely that the hostname and the IP domain name resides in the same domain, for instance `mail.acme.it` and `someserver.acme.it`.

```
Delivered-To: target@gmail.com
Received: by 10.68.237.229 with SMTP id xxxxxxxxxxxxx;
        Sat, 8 Mar 2014 06:15:38 -0800 (PST)
X-Received: by 10.14.206.137 with SMTP id
            xxxxxxxxxxxxxxx.xx.xxxxxxxxxxxxx;
            Sat, 08 Mar 2014 06:15:37 -0800 (PST)
Return-Path: <Daniele_Bellavista@acme.it>
Received: from mail.acme.it ([192.168.66.55])
        by mx.google.com with ESMTPS id
        xxxxxxxxxxxxxx.xxx.2014.03.08.06.15.36
        for <target@gmail.com>
        (version=TLSv1 cipher=ECDHE-RSA-AES128-SHA
        bits=128/128);
        Sat, 08 Mar 2014 06:15:37 -0800 (PST)
Received-SPF: neutral (google.com: 192.168.66.55 is
        neither permitted nor denied by best guess
        record for domain of Daniele_Bellavista@acme.it)
        client-ip=192.168.66.55;
Authentication-Results: mx.google.com;
       spf=neutral (google.com: 192.168.66.55 is
       neither permitted nor denied by best guess
       record for domain of Daniele_Bellavista@acme.it)
       smtp.mail=Daniele_Bellavista@acme.it
Received: from SERVER.acme.local
       ([1:2:3:4:5:6]) by
       acme-Server.acme.local
       ([1:2:3:4:5:6%7]) with mapi id
       xx.xx.xx.xx; Sat, 8 Mar 2014 15:15:34 +0100
From: Daniele Bellavista <Daniele_Bellavista@acme.it>
To: "target@gmail.com" <target@gmail.com>
Subject: This is the real message
```

Figure 4.7: Real email header

- **Hostname distance check -** if the email address has an hostname similar to a known one, it could be a social engineering approach. Hostname distance check can be implemented using *Approximate String Matching*, against a database of known hosts.

- **IP reputation -** IP addresses associated with mail servers are likely to have a good reputation, that is many DNSBLs (DNS blackhole lists) repute it has good. On the other hand, a normal IP address is likely to have a neutral or bad reputation. IP reputatio can be used both in Spoofing and Similar Domain cases.

Figure 4.9 shows a proof of concept, written in python of the email checks. Output example are shown in figure 4.10.

The Spoofing and Similar Domain cases can be pinpointed easily if those

```
Delivered-To: target@gmail.com
Received: by 10.68.237.229 with SMTP id xxxxxxxxxxxxx;
        Sat, 8 Mar 2014 06:50:25 -0800 (PST)
X-Received: by 10.14.180.2 with SMTP id
        xxxxxxxxxxxxxx.xx.xxxxxxxxxxxxx;
        Sat, 08 Mar 2014 06:50:24 -0800 (PST)
Return-Path: <Daniele_Bellavista@acme.it>
Received: from mail.acme.it ([192.168.100.100])
        by mx.google.com with ESMTP id
        xxxxxxxxxxxxxxx.xxx.2014.03.08.06.50.23
        for <target@gmail.com>;
        Sat, 08 Mar 2014 06:50:24 -0800 (PST)
Received-SPF: neutral (google.com: 192.168.100.100 is
        neither permitted nor denied by best guess
        record for domain of Daniele_Bellavista@acme.it)
        client-ip=192.168.100.100;
Authentication-Results: mx.google.com;
       spf=neutral (google.com: 192.168.100.100 is
       neither permitted nor denied by best guess
       record for domain of Daniele_Bellavista@acme.it)
       smtp.mail=Daniele_Bellavista@acme.it
Date: Sat,  8 Mar 2014 15:49:54 +0100 (CET)
From: Daniele_Bellavista@acme.it
Subject: This is the spoofed message
```

Figure 4.8: Spoofed email address

conditions are satisfied. However, emails falling into the External Domain case or sent from a valid internal address (e.g. as part of an insider attack or following a mailbox exploitation) are hard to be marked as suspicious without a context. The defence strategy needs to be adjusted basing on the specific policies or conventions inside the company, for instance:

- *Can email addresses be categorized basing on importance and risk?* Email address with high importance could be monitored with restrictive criteria.

- *Which are common attachments file type in internal emails?* Are executable attached often or only PDF or CAD documents?

- *Are attachment expected from external domains? If so can those domains be listed?* A company may usually accept documents from suppliers from their mail server, but not from external domains.

- *Is the analyst team allowed to manually process the email content?* A negative answer implies the request of authorization for every received warning, introducing an additional delay in the detection time.

```python
#!/usr/bin/env python2
import httplib
import sys
import socket
import re
import Levenshtein

ip = sys.argv[1]
domain = sys.argv[2]
email = sys.argv[3]

# Reverse DNS query
real_domain = socket.gethostbyaddr(ip)[0]
# Equal domain level
level = 0
for e1,e2 in zip(reversed(real_domain.split('.')),
    reversed(domain.split('.'))):
  if e1 != e2:
    break
  level += 1
if level < 2:
  print " [!] Different domain (" + real_domain + ")!"

# Reputation check using www.reputationauthority.org
h1 = httplib.HTTPConnection('www.reputationauthority.org')
h1.request('get', '/lookup.php?ip=' + ip)
resp = h1.getresponse()
data = resp.read()
h1.close()
# Obtaining the reputation as a number from 0 to 100 (max trust)
val = re.compile('images/([0-9]+)\.gif').findall(data)[0]
val = 100 - int(val)
if val < 60:
  print ' [!] Reputation: '+str(val)+'/100'

# Hostname distance check
known_hosts = ['google', 'acme', 'yahoo']
hostname = re.compile('.*@(.*)\.\w+').findall(email)[0]
for h in known_hosts:
  ratio = Levenshtein.ratio(hostname, h)
  if ratio < 1 and ratio > 0.5:
    print ' [!] Host similarity with ' + h

```

Figure 4.9: Sample code for domain name and IP reputation check

4.3.2 Executable and documents analysis

Malware are essential parts of an advanced attack and their detection is of primary importance. A malware could be deployed by a direct link down-

```
# Valid email domain, good reputation, good email address

 $ ./check_sender.py\
      192.168.66.55 mail.acme.it Daniele_Bellavista@acme.it

# Different email domain, poor reputation, good email address

 $ ./check_sender.py\
      192.168.100.100 mail.acme.it Daniele_Bellavista@acme.it
 [!] The host has a different domain (web.ispnet.it)!
 [!] Reputation: 15/100

# Different email domain, poor reputation, similar email address

 $ ./check_sender.py\
      192.168.100.100 mail.acme.it Daniele_Bellavista@acrne.it
 [!] The host has a different domain (web.ispnet.it)!
 [!] Reputation: 15/100
 [!] Host similarity with acme
```

Figure 4.10: Example usage of `check_sender.py`. See figure 4.9

load, as email attachment or by physical interaction (e.g. from an USB stick).

The analysis of malware deployed using an USB stick requires an host-based IDS, while a NIST can analyze email attachments and downloads.

There are multiple file type to check:

- **Portable executable and windows scripts -** the most common malware.
- **Executable and linkable format and shell scripts -** malware for linux are rare can be part of a targeted attack.
- **OS X applications and shell scripts -** same for linux, OS X can be infected as part of a targeted attack.
- **APK, XAP, APPX and iOS Apps -** also mobile device could be targeted.
- **Browser extensions and flash applications**
- **PDF documents -** PDF may contains exploits and shellcodes.
- **Office documents -** can contain macros and exploits.
- **Archives -** an archive, especially if protected with a password, could hide malware.

Common antivirus scan or dynamic malware analysis are needed, because they can identify opportunistic malware, but fails to detect targeted attacks malware.

The theoretical solution is to perform a manual analysis of each file, but it can't be employed in the real world because of the high number of samples. For instance `ACME Corporation` downloads every days about a thousand PDF documents and a thousand executables.

My proposal is studied to work side by side with dynamic analysis, in order to provide additional hints or warnings for the analyst team. The detectors inspect the defence system events to identify suspicious activities. I have identified some detection approaches that can be retrieved from download information, metadata and the dynamic analysis result:

- **Source IP and Host:** as for emails source IP, the host and IP of the downloaded malware can be analyzed for reputation and similar names. Host reputation, however, is a lesser indication than mail reputation, because a neutral reputation can be accepted.
- **Filename:** a common social engineering technique is to create an executable with a double extension (e.g. `.doc.exe` or `.pdf_____.exe`). Also files called `Invoice.exe` or `Important Document.exe` are suspicious.
- **Icon analysis:** for executable files, another indicator is to check the executable icon for similarity with document icons, such as PDF or DOC icons.
- **Metadata analysis:** document metadata, such as comments, could serve as a social engineering technique or may contains extra data for a multistaged attack.
- **Analysis on the analysis:** dynamic analysis permits to extrapolate key feature of the sample. Although the behavior may not results suspicious, some behavior may need further analysis. A detector could rise a warning if:
  - **Common sandbox detection:** some bypass action are recorded by the analysis. A warning could be the presence of behaviors associated with sandbox detection techniques, especially if such evasion works against the malware analyzer used in the company.
  - **Scan for files or process:** another suspicious behavior can be the scan for a particular file or process. This can be an indicator of a multistaged attack.
  - **Presence of active code inside Documents:** although legit documents can contains active code or plugins, their presence could lead to further investigation.

4.3.3 Network communications

For each step trough the attack process, it becomes harder to detect a targeted attack. A network communication detector aims to identify possible C&C communications and data exchange. Thus, it's essential to have deployed a TSL inspection, otherwise the encryption would give an attacker a chance to hide its communications.

The analysis of network communication is extremely dependent from the company infrastructure and policies, but once the defence system is able to analyze every communication, the analyst team could highlight on both headers or content of the messages.

By looking at headers, the analyst team could identify suspicious communications by looking at:

- **Non-standard communication:** communication of unexpected or unknown protocols could indicate a malware activity. For instance, file sharing or torrent protocols could be unexpected or UDP communication on non standard ports.
- **Periodically visited websites:** basing on the company policies, a periodic visit of an unexpected website may be an indication of a malicious activity.
- **Host reputation:** contacts to hosts with poor reputation can be a suspicious activity.

Further detection could analyze the communications content for the presence of:

- Document uploading
- Inconsistent content with protocol header
- Unexpected encrypted content inside the communication

4.4 HAZARD: Hacking Approach for Zealot Advanced Response and Detection

`WASTE` isn't a complete solution against a targeted attacks. It strictly relies on existing detection systems, and their deploy inside the company. An expert attacker could completely bypass the defence system, neutralizing the effectiveness of `WASTE`. Or, even worse, it could know the exact detection methods and bypass them.

To provide a more comprehensive defence strategy, the organization should offer a full-time service against targeted attack. I propose an approach called `HAZARD` (Hacking Approach for Zealot Advanced Response and

Detection), aimed to provide security assessment inside a company. `HAZARD` uses `WASTE` together with existing defence systems, to provide:

- **Defence against opportunistic attacks:** the work includes detection, evaluation and response plan against opportunistic attacks.
- **Defence against targeted attacks:** an team should actively analyze events and warnings for clues of a targeted attack.
- **Vulnerability Assessment:** as part of the security service, the vulnerability assessment is offered for attack path detections and vulnerability notification.
- **Targeted attacks as a service:** this service may offers advanced advices on the weak points of a company against targeted attacks, together with refresher courses from an attacker point of view.

`HAZARD` is a business process with various groups and roles (described in section 4.4.1) and subprocesses. In order to detect targeted attacks, the `HAZARD` process must promote self-evolution and collaboration between groups. Everything that is learned by a single role must become part of the organization global knowledge, to gain the necessary flexibility when dealing with targeted attacks. The subprocesses are:

1. **Incident analysis:** analysis and mitigation of incidents detected by conventional defence systems (section 4.4.2).
2. `WASTE` **warning analysis:** analysis, correlation and possible mitigation of warnings generated by `WASTE` (section 4.4.2).
3. `WASTE` **issue management:** analysis, fix and feedbacks for issues on `WASTE` (section 4.4.4).
4. **Vulnerability assessment:** vulnerability management applied to the company (section 4.4.5).
5. **Targeted attack evaluation:** analysis of weak points and attack path from an attacker point of view (section 4.4.6).
6. **Targeted attack test:** penetration testing using targeted attacks techniques (section 4.4.7).

4.4.1 HAZARD Roles

The process identifies three groups:

- **Company IT:** the company IT is a role that must be present inside the client company. It's the point of contact with their system administrators.

- **Detection Den (D-Den):** the D-Den is the group assigned to the attack detection. It consist in two roles:
    1. **Analysis Team (A-Team):** the A-Team is responsible of analyzing the defence systems events and `WASTE` warnings to detect, evaluate and report attacks. The A-Team can be the *Analyst Team* actor of `WASTE` (see figure 4.1).
    2. **Detection Team (D-Team):** the D-Team is responsible of managing `WASTE`. In particular they have to correct, produce and improve the warning detectors. The D-Team can be the *Detection Team* actor of `WASTE` (see figure 4.1).
- **Hacking Den (H-Den):** the H-Den is assigned to perform vulnerability assessment and targeted attacks. The H-Den is also composed of two roles:
    1. **Vulnerability Team (V-Team):** the V-Team is responsible of performing vulnerability assessment, evaluation and fix plan for the Company IT.
    2. **Hacking Team (H-Team):** the H-Team performs evaluation from an attack perspective, and organize refresher courses for security experts and personnel.

4.4.2 Incident Analysis

Name	**Description**
Incident detection	Incident elaborated from any defence system. It's very likely to be an opportunistic attack
Mitigation Plan	The A-Team sent to the IT administrators the Mitigation Plan artifact
Mitigation Result	Once the mitigation plan is terminated, the IT administrators should send the A-Team the mitigation result.
Incident overview	Overview of the incident after the analysis
Attack Feedback	Knowledge on attacks methodologies gained by analyzing the incident
Vulnerability Feedback	Knowledge on vulnerabilities gained by analyzing the incident

Table 4.3: Incident analysis events

The incident analysis process involves all roles (see figure 4.11). It represents the mitigation action against a standard incident detected by the

Name	Description
Internal Incident Report	Report for internal use, generated after the incident analysis. This report includes technical and exhaustive details.
Mitigation Plan	Work plan for the threat mitigation. It contains summary description, risk evaluation and mitigation instruction for the company IT administrators.
Internal Attack Analysis Report	Report for internal use, generated from the incident analysis. It includes details on how the attack was performed. The goal of this document is to learn new attack techniques.
Known vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company.
Internal Vulnerability Analysis Report	Report for internal use, generated from the incident analysis. It includes details on which vulnerabilities the attack exploited. The goal of this document is to rank the opened fix plans and to learn new attack techniques.
Fix plan	Similar to the mitigation plan, it contains description, risk evaluation and fix plan for the company IT administrators.

Table 4.4: Incident analysis artifacts

defence systems. Table 4.3 shows the process events description, table 4.4 shows the process artifacts description.

The process starts when the A-Team receives an *Incident detection* message. The incident is analyzed and processed to produce an exhaustive report of the attack and a mitigation plan. While the mitigation plan is sent to the company IT administrators, the report is sent to the H-Team and V-Team, who will respectively generate an attack report and a vulnerability report.

If the mitigation isn't successful, both the mitigation plan or the incident analysis could be erroneous. The A-Team is responsible to reiterate the analysis process to understand what went wrong.

4.4.3 WASTE warning analysis

As the incident analysis, the warning analysis process also involves the participation of all roles (see figure 4.12). It represents the analysis and correlation of the `WASTE` warnings to obtain indication of a targeted attack. If a T.A. is detected, then mitigation actions can be performed. Table 4.5 shows the process events description, table 4.6 shows the process artifacts

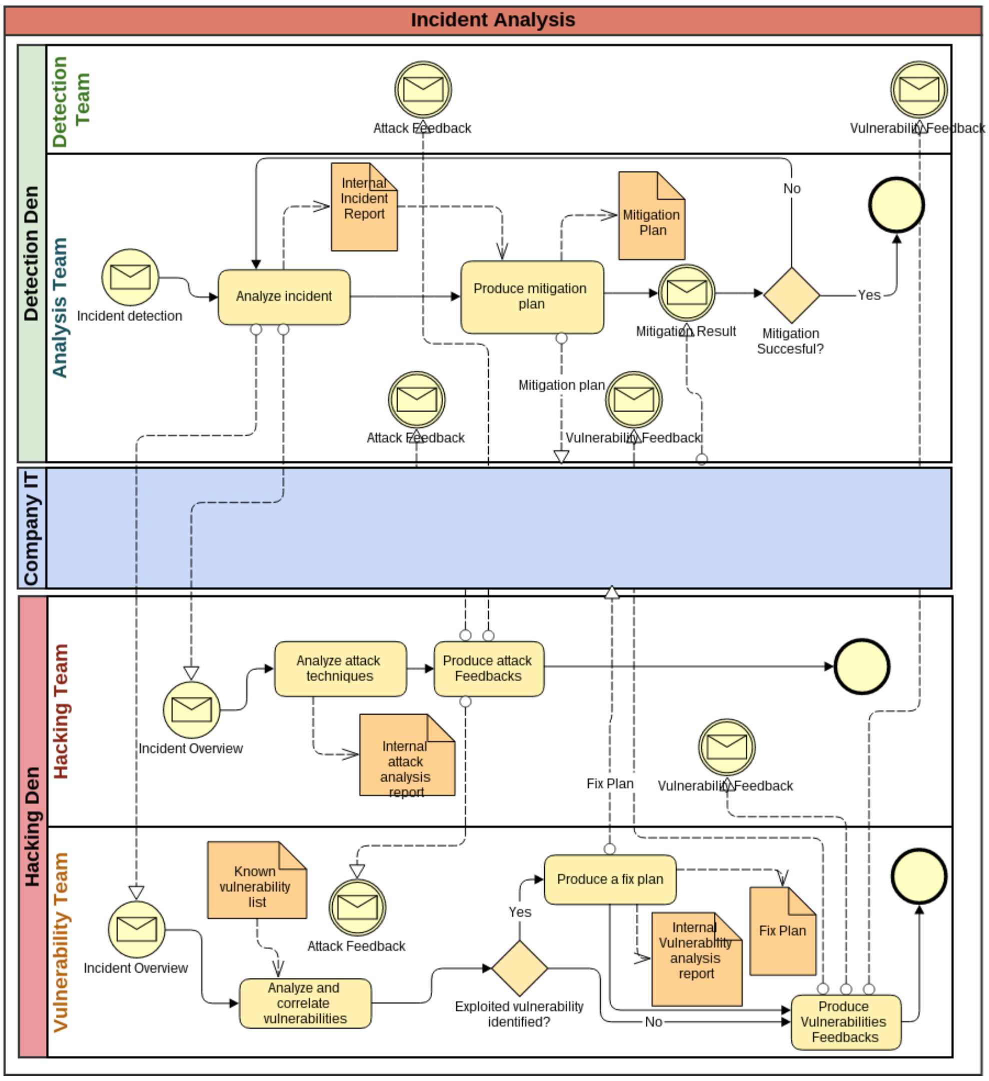


Figure 4.11: HAZARD: incident analysis process

description.

The process starts when the A-Team receives a `WASTE` *warning* message. The warning is analyzed to detect indicators of a targeted attacks. Then the A-Team tries to correlate existing indicators to confirm a targeted attack. At the end of the process, the D-Team receives a feedback on the warning, to improve or fix `WASTE` detectors.

If a targeted attack is detected, the A-Team produces a report for the company, the V-Team and the H-Team. The company has to remove the threat, evaluate the attack damage and pursue legal actions, while the H-Team and V-Team will respectively generate a a report about attack techniques and about exploited vulnerabilities.

Name	Description
WASTE Warning	Warning generated by the `WASTE` system.
Authorization	Further analysis may have to access to restricted or sensitive information. The A-Team must ask for permission.
Permission Granted	In reply to authorization message, it authorizes the access to the required data.
Permission Denied	The access to the required information is denied. The A-Team must proceed the analysis basing on whatever data they have extrapolated.
Mitigation Plan	Work plan for the threat mitigation. It contains summary description, risk evaluation and mitigation instruction for the company IT administrators.
T.A. Company Report	Detailed report based on forensic evaluation of the targeted attack. The company may need it to pursue legal actions or to evaluate the damage.
T.A. Report	Detailed report of the Targeted attack sent to the H-Team and V-Team for further analysis.
Warning Feedback	Feedback on how the warning has been analyzed and correlated with other indicators.
T.A. Feedback	Knowledge on targeted attacks methodologies gained by analyzing the targeted attack
Vulnerability Feedback	Knowledge on vulnerabilities gained by analyzing the targeted attack

Table 4.5: Warning analysis events

Name	Description
Analyzed Warning List	Opened and uncorrelated warnings already analyzed by the A-Team.
T.A. Indicator List	List of indicators extrapolated by the warnings.
Internal T.A. Report	Report for internal use, generated from the analysis of warnings and indicators. It includes details on how the targeted attack was performed. The goal of this document is to expand the organization knowledge.
Internal Warning Report	Report for the D-Team, generated from the warning analysis. It includes details on how the warning was used, which extra information was required and so on. The objective of this document is to stimulate the improvement of the `WASTE` system.
Mitigation Plan	Work plan for the threat mitigation. It contains summary description, risk evaluation and mitigation instruction for the company IT administrators.
T.A. Company Report	Technical details and forensic results obtained from the analysis of the targeted attack. The report is meant to be used for legal purpose and damage evaluation.
Internal T.A. Analysis Report	Report for internal use, generated from the targeted attack analysis. It includes details on the attack process. The goal of this document is to learn new attack approaches and to understand the company weak points.
Social Engineering Report	Report for internal use, generated from the targeted attack analysis. It includes more details on the social engineering approaches used from the targeted attacks. The goal of this document is to provide policy improvement in the company.
Known vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company.
Internal Vulnerability Analysis Report	Report for internal use, generated from the targeted attack analysis. It includes details on which vulnerabilities the attack exploited. The goal of this document is to rank the opened fix plans and to learn new attack techniques.
Fix plan	Similar to the mitigation plan, it contains description, risk evaluation and fix plan for the company IT administrators.

Table 4.6: Warning analysis artifacts

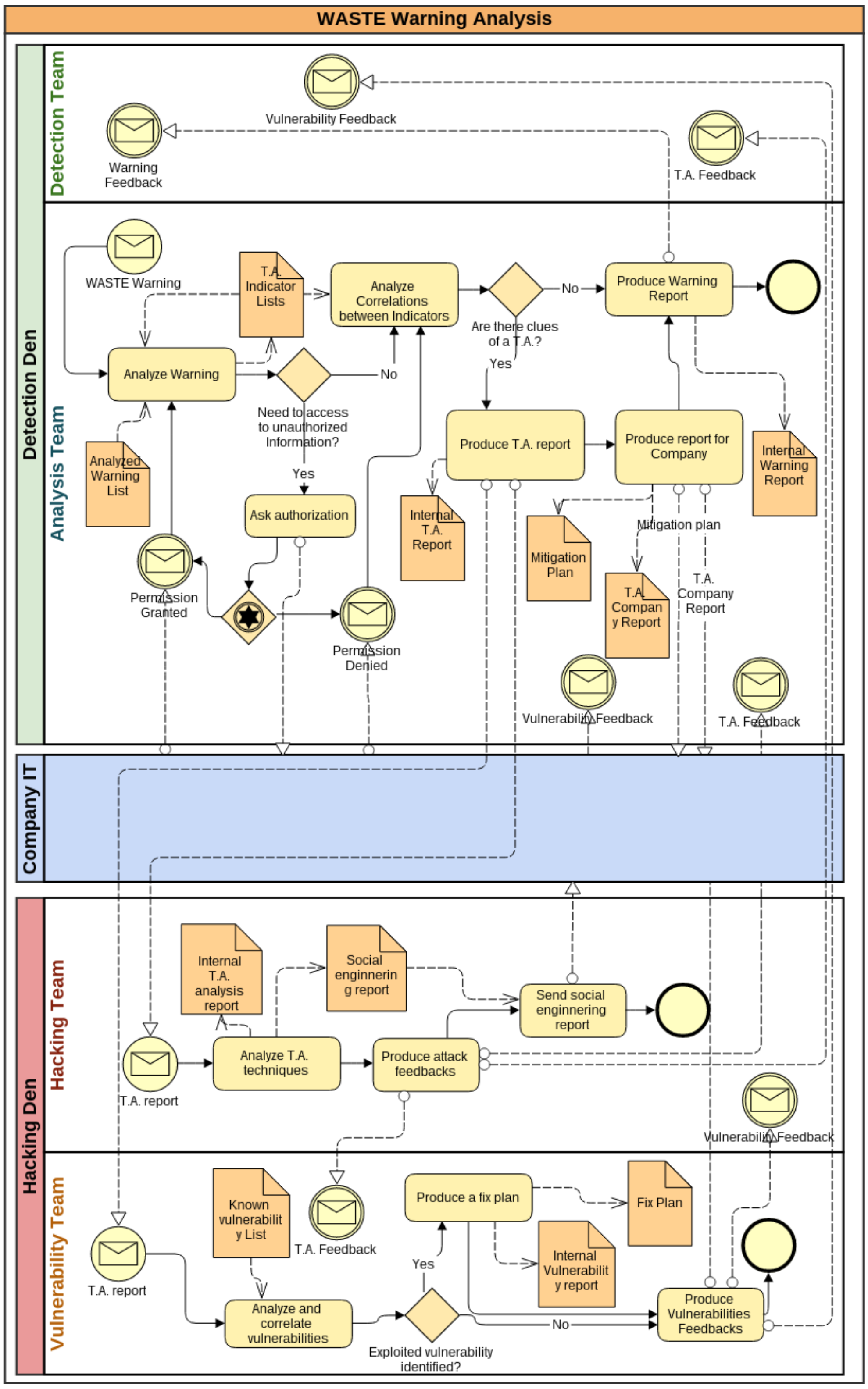


Figure 4.12: HAZARD: warning analysis process

4.4.4 WASTE issue management

Name	Description
WASTE issue detected	Event generated by the monitoring of the WASTE system and caught by the D-Team.
Overview and related warning of the issue	The D-Team send to the A-Team an overview of the issue and the warnings that may be effected.
Issue Feedback	The A-Team replies to the D-Team a feedback on the issue.

Table 4.7: WASTE issue management events

Name	Description
Analyzed Warning List	Opened and uncorrelated warnings already analyzed by the A-Team.
T.A. Indicator List	List of indicators extrapolated by the warnings.
Internal T.A. Report List	List of analyzed targeted attacks.
Issue Report	Report on the issue, containing an exhaustive description of the issue, its fix and the feedbacks from the A-Team.

Table 4.8: WASTE issue management artifacts

The WASTE issue management process is part of the continuous improvement of the detection system. When the D-Team identifies an issue or in WASTE detectors, it must fix it, send a feedback to the A-Team which will analyze the affected warnings, indicators and reports. This operation generates a feedback to be sent back to the D-Team. If the issue is solved, a report is compiled and the issue is solved. Otherwise the issue must reiterate the fixing process.

The process is shown in figure 4.13, while table 4.7 shows the process events description and table 4.8 shows the process artifacts description.

4.4.5 Vulnerability Assessment

As part of the complete security service, the D-Team performs periodic scans of the internal and external systems, in order to find known vulnerabilities and evaluate them. Once the scan results are ready, the D-Team determinates new vulnerabilities, producing a report and a fix plan for each of them. If some vulnerabilities are already known, it may happen that further knowledge had made necessary to reevaluate and change the risk level.

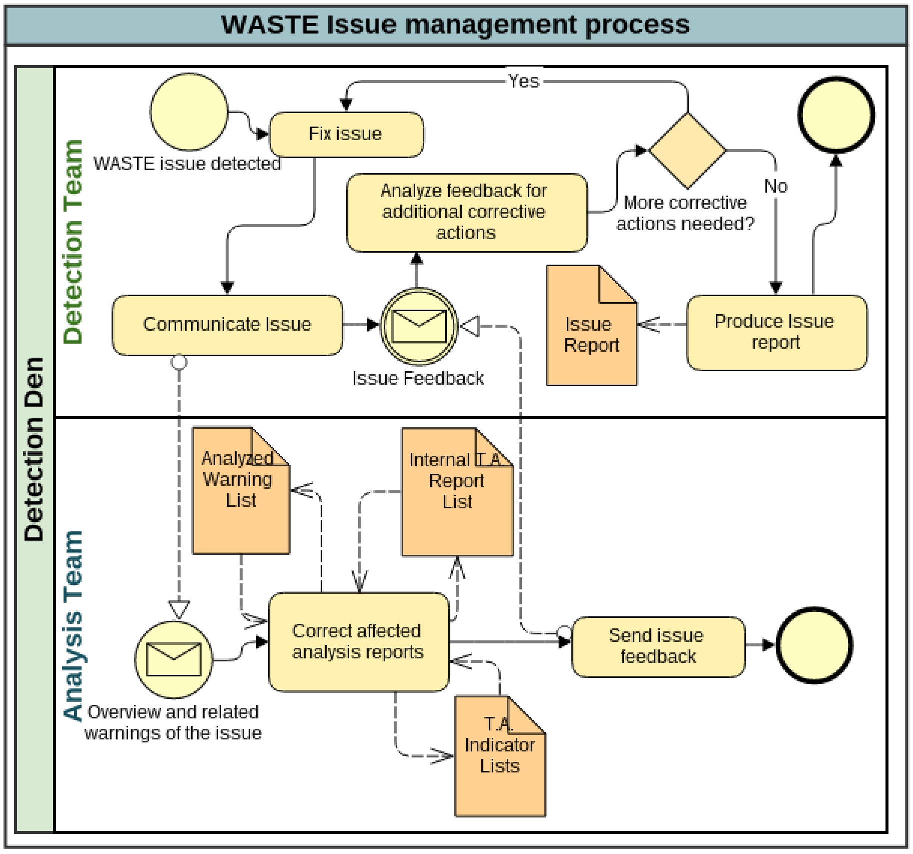

WASTE Issue management process
Detection Den
Detection Team
Analysis Team
WASTE issue detected
Fix issue
Yes
More corrective actions needed?
No
Analyze feedback for additional corrective actions
Communicate Issue
Issue Feedback
Issue Report
Produce Issue report
Analyzed Warning List
Internal T.A. Report List
Correct affected analysis reports
Send issue feedback
Overview and related warnings of the issue
T.A. Indicator Lists

Figure 4.13: HAZARD: WASTE issue management process

Name	**Description**
Periodic Scan	Periodic event that determinates when a scan should be launched.
Scan results	Obtained by the vulnerability scanners. Contains a list of vulnerability present in the system.
Vulnerability Feedback	The D-Team sends to the organization a feedback on the vulnerabilities, in order to increase the organization knowledge.

Table 4.9: Vulnerability assessment events

In such case, a new report and fix plan must be compiled and sent to the Company IT administrators.

The process is shown in figure 4.14, while table 4.9 shows the process events description and table 4.10 shows the process artifacts description.

Name	Description
Known vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company.
Internal Vulnerability Analysis Report	Report for internal use, generated from the incident analysis. It includes details on which vulnerabilities the attack exploited. The goal of this document is to rank the opened fix plans and to learn new attack techniques.
Fix plan	Similar to the mitigation plan, it contains description, risk evaluation and fix plan for the company IT administrators.

Table 4.10: Vulnerability assessment artifacts

4.4.6 Targeted attack evaluation

Name	Description
Planned Evaluation	Periodic event for the planned attack evaluation.
Attack Evaluation Report	Contains the description of possible exploitable vulnerability, attack path and spreading techniques seen from an attacker point of view. Each element or group of elements has an associated risk. The report is needed by the Company IT administrator for technical and administrative evaluation.
Attack Evaluation Feedback	The H-Team sends to the organization a feedback on the evaluation of the targeted attack, in order to provides new point of view for each activity.

Table 4.11: Targeted attack evaluation events

The targeted attack evaluation is one of the two hacking services performed by the H-Team. Periodically, the H-Team evaluates possible attack paths, by starting with an information gathering phase. Then, the H-Team supposes to be inside the company and analyzes internal vulnerabilities to find possible spreading approaches and APT techniques. Finally, for each scenario, the attack risk is evaluated and the H-Team send report to the Company IT administrators. As always, the H-Team sends to the other teams feedbacks on the attack.

The process is shown in figure 4.15, while table 4.11 shows the process events description and table 4.12 shows the process artifacts description.

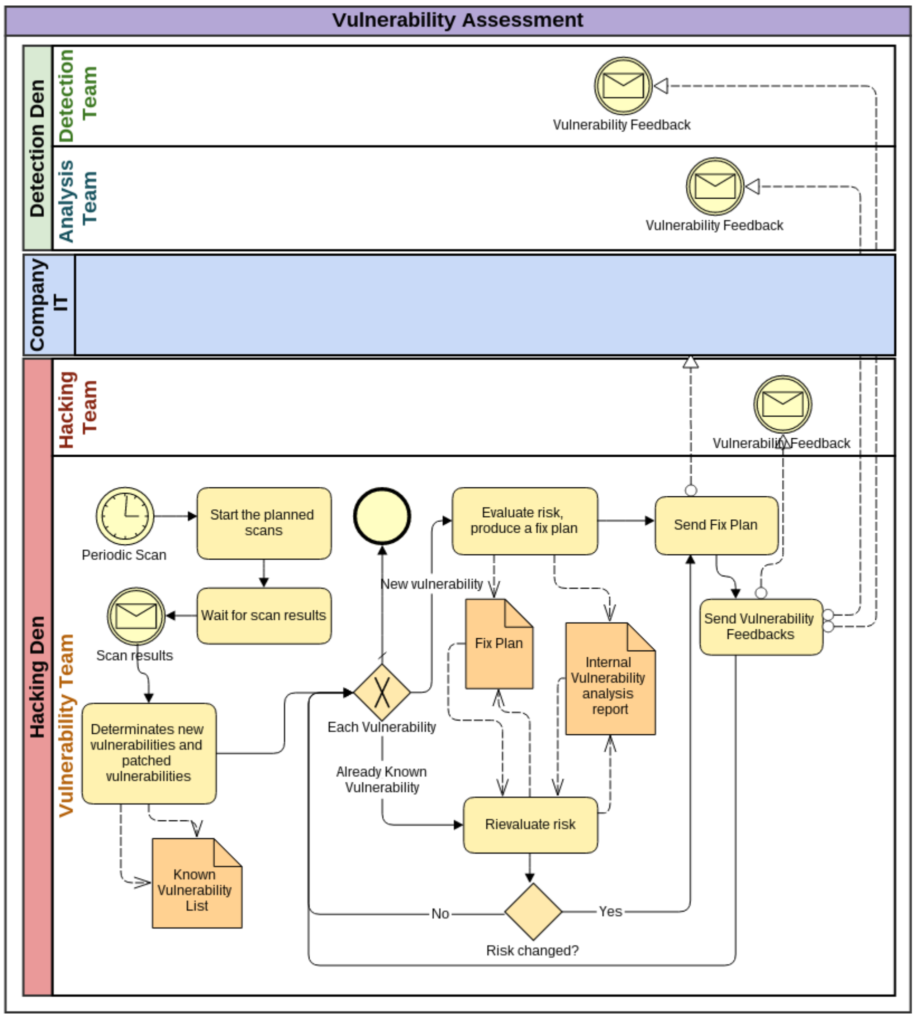


Figure 4.14: HAZARD: vulnerability assessment process

4.4.7 Targeted attack test

The targeted attack test is the second hacking services performed by the organization. With a longer period than attack evaluation, the H-Team try to perform a real targeted attack against the company. The H-Team may use any knowledge gathered during the attack evaluation process to design possible attack paths, malware or exploits.

The targeted attack can succeed or even fail, meaning that the H-Team could not bypass the defence measures. In either case the H-Team comiles an exhaustive report for internal use and a report for the company. The latter must also include a comparison with the other attacks to understand what changed and what not.

The process is shown in figure 4.15, while table 4.11 shows the process

Name	Description
Known external vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company external systems.
Known internal vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company internal systems.
External Knowledge	Knowledge gathered using information gathering techniques from the outside.
Social engineering approach list	Possible approaches that can be developed basing on the external knowledge.
Attack approach list	Possible approaches that can be developed basing on the external knowledge.
Spreading and APT approach list	Possible approaches that can be developed by analyzing the internal vulnerability and the attack path. The spreading approaches include ways for the attack to infect more valuable hosts, while the APT approaches include methods for advanced, persistent attack.
Attack evaluation report	Contains an evaluation of possible complete attack process with associated risks for data and services disclosure, disruptor and so on. This artifact is meant to be read and evaluated by advanced company IT administrators.

Table 4.12: Targeted attack evaluation artifacts

Name	Description
Planned targeted attack test	Periodic event for the planned targeted attack test.
Attack test report	The H-Team send to the Company IT administrator the attack test report.
Attack test feedback	The H-Team sends to the organization a feedback on the targeted attack, in order to provides new point of view for each activity.

Table 4.13: Targeted attack test events

events description and table 4.12 shows the process artifacts description.

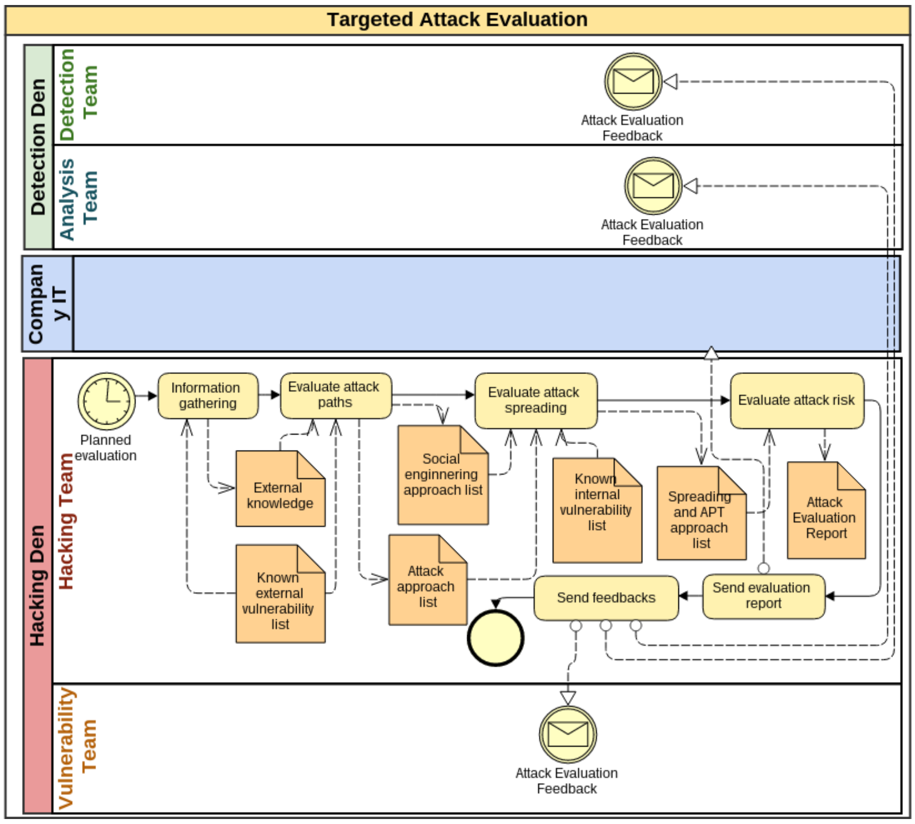


Figure 4.15: HAZARD: targeted attack evaluation process

Name	Description
Known external vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company external systems.
Known internal vulnerability list	Artifact that contains all the opened and unfixed vulnerabilities of the company internal systems.
External Knowledge	Knowledge gathered using information gathering techniques from the outside.
Social engineering approach list	Possible approaches developed during the T.A. evaluation process.
Attack approach list	Possible approaches developed during the T.A. evaluation process.
Spreading and APT approach list	Possible approaches developed during the T.A. evaluation process.
Delivery techniques	Evaluated using all the available knowledge, they include all possible attack vectors and delivery techniques for all or part of the weapons.
Weapons	Malware or software to be used during the attack. Basing on the attack vector, the weapon will be packed inside a deliverable, such as a zip or PDF document.
Attack test internal report	Contains details on the targeted attacks. How it failed or succeeded, any problems and results.
Attack test internal report list	Previous report of targeted attacks.
Attack test report	Contains the description of the attack, how it succeeded or how it failed. In case of successful attack, the report include how deep the infection went and what a real attacker could had achieved. The report contains also comparison with previous attacks.
Attack evaluation report	Contains an evaluation of possible complete attack process with associated risks for data and services disclosure, disruptor and so on. This artifact is meant to be read and evaluated by advanced company IT administrators.

Table 4.14: Targeted attack test artifacts

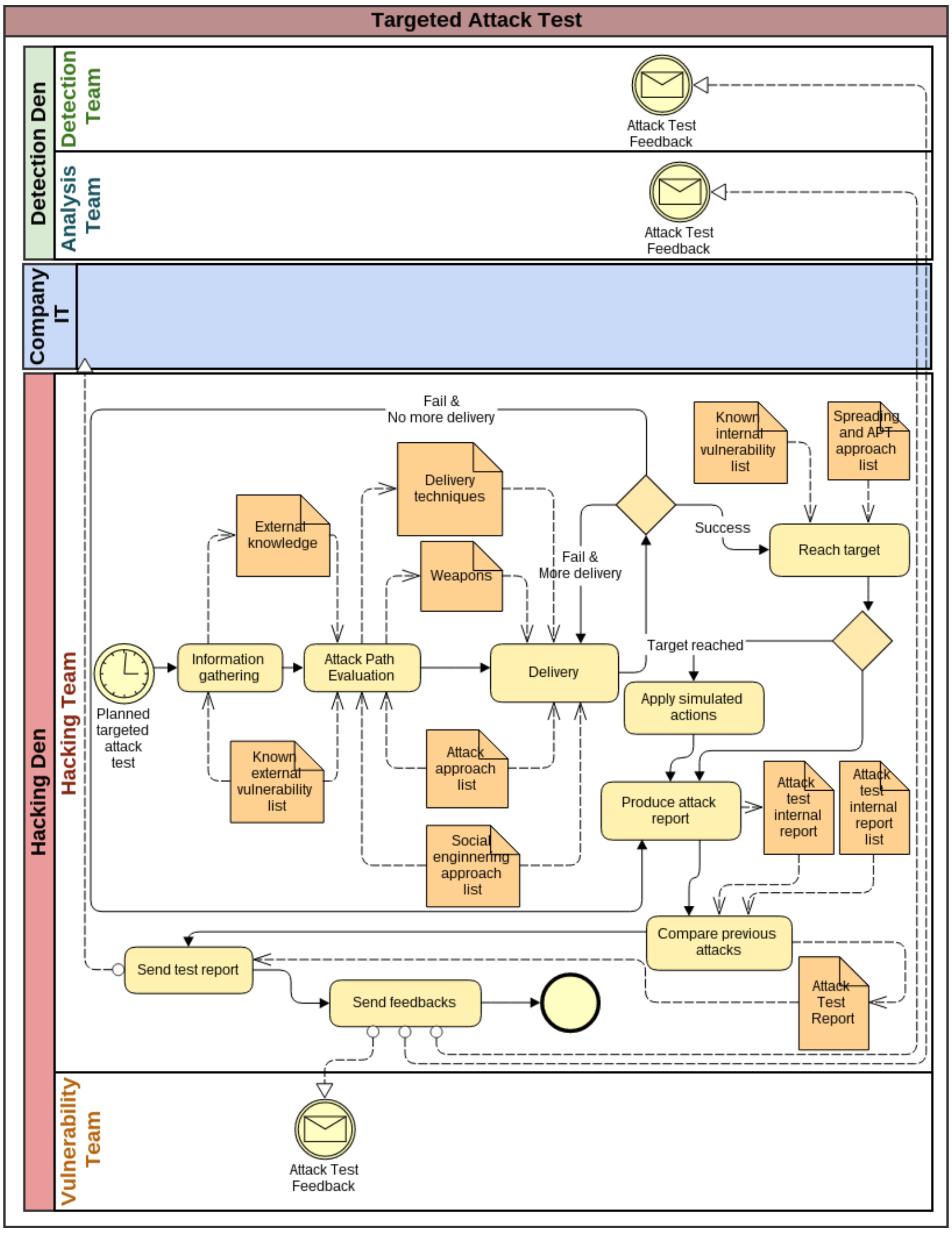


Figure 4.16: HAZARD: targeted attack test process

Chapter 5

Defence strategy application

After performing the targeted attack described in section 4.1, my work moved to analyze and defend the `ACME Corporation` infrastructure.

The next sections report the application description, used tools and results of the `HAZARD` process phases. In section 5.1 I describe the defence process against opportunistic attacks and their mitigation, in section 5.1.3 the defence against targeted attacks, while in section 5.2 I report the approach to vulnerability assessment.

I didn't perform the `HAZARD` phases targeted attack evaluation and test, because they would required more technical evaluation and lot of paperwork.

5.1 Defence against and mitigation of cyber attacks

The defence against attack was based on events created by automatic defence systems. `ACME Corporation` used an installation of `FortiGate`, which is an application firewall (see section 3.2.2) to provide policy enforcement and an installation of `Lastline` to monitor the network.

I created also some `WASTE` detectors, to narrow the search of suspicious communications and malicious downloads.

5.1.1 Lastline

`Lastline` is a network intrusion detection system (see section 3.3), aimed to analyze both the objects that enter a network and the traffic being generated by the internal hosts[25]. `Lastline` is composed by three components:

Sensor: soft appliance that collects incoming artifacts (documents and executable downloaded or received as email attachments), identifies malicious behavior in network traffic, collects statistic about network traffics and optionally blocks malicious network activities[25].

Manager: soft appliance that collects artifacts and information from sensors and presents them to the user. The Manager also sends to the Engine the received artifact for analysis. It's also responsible of categorize and prioritize the events using an incident-centered approach[25].

Engine: soft appliance responsible for the analysis of artifacts. It uses a proprietary sandbox based on Anubis[25].

5.1.2 Incident monitoring, evaluation and mitigation

The incident monitoring had two objectives:

1. **Remove existing malwares -** Some hosts had been infected before the installation of `Lastline`, or they were compromised by other means. For instance mobile computer can be infected before being plugged to the network. Figure 5.1 shows an example of malicious events being detected by `Lastline`.

2. **Detect new malware downloads -** New downloaded malware must be rapidly detected and removed from the host before they can spread or communicate with the C&C. Figure 5.2 shows an example of malicious downloads analyzed and reported by `Lastline`.

3. **Detect suspicious downloads -** Some downloaded malware are reported by `Lastline` as non-malicious, despite being harmful. These samples can be either new malware variants or even targeted malware.

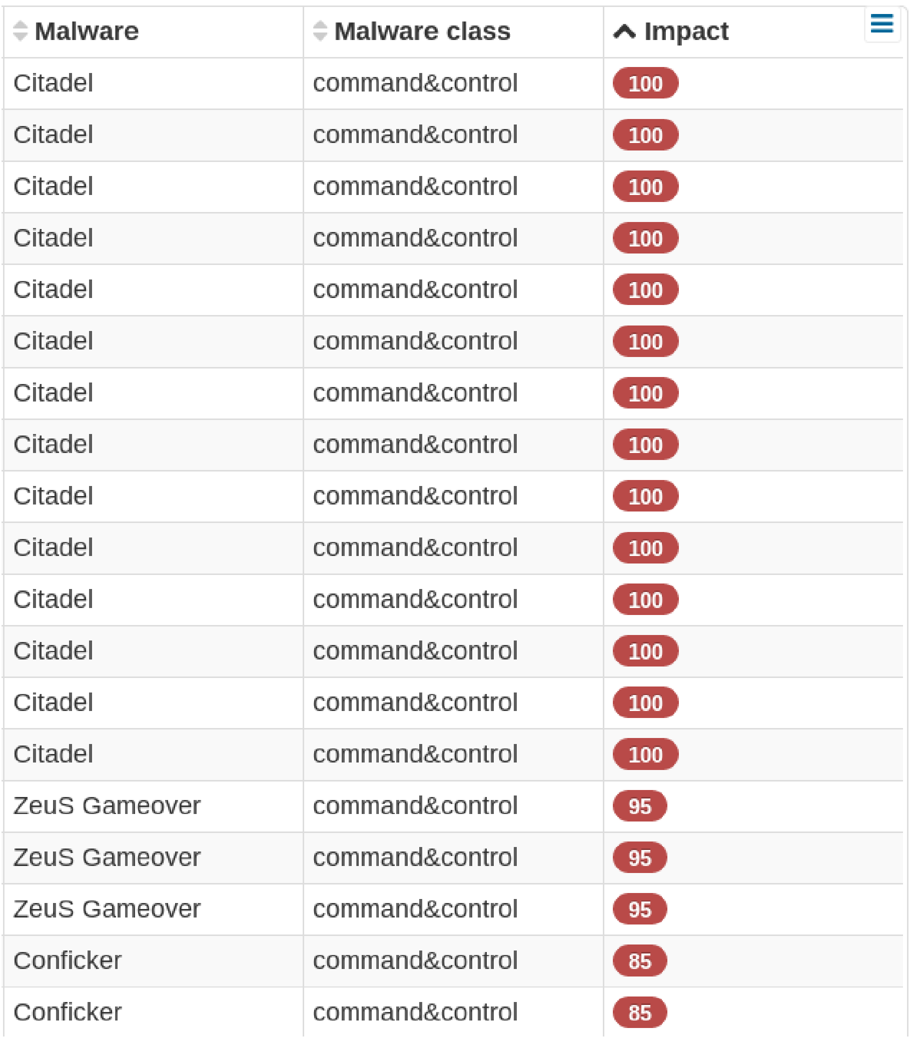
4. **Detect suspicious communications -** As for downloads, `Lastline` could underestimate the score of some communication. However, since `Lastline` registers only suspicious communications, further analysis requires a particular attention. For instance a moving laptop or the expiration of the DHCP leasing time can break some automatic analysis.

For each incident or malware download, I had to produce a mitigation plan to be sent to the *company IT administrators*. Each plan has a score, computed basing on the following parameters:

- **Host owner and role:** C-team hosts and production server, for instance, should have a higher priority.

- **`Lastline` score:** score automatically assigned to the incident by `Lastline`.

- **Machine type (desktop, laptop or server):** an infected laptop can spread the infection also by physical relocation.

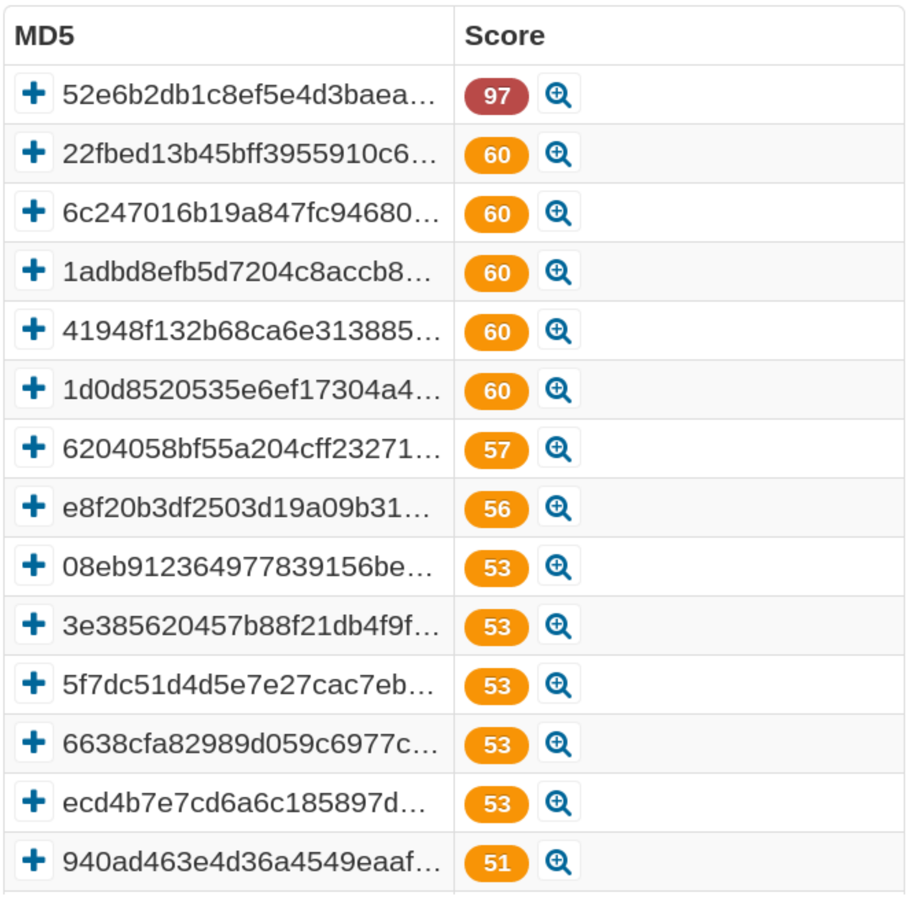


Malware	Malware class	Impact
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
Citadel	command&control	100
ZeuS Gameover	command&control	95
ZeuS Gameover	command&control	95
ZeuS Gameover	command&control	95
Conficker	command&control	85
Conficker	command&control	85

Figure 5.1: Malicious events detected by `Lastline`

- **Incident status:** gives an indication of the effective threat. It can be *confirmed* or *blocked*, meaning that the malware can communicate with the C&C or if their communications are blocked by the peripheral systems. The incident status should be related to the machine type. A laptop with a blocked malware, once outside the company network, could communicate with the C&C.

When dealing with malware downloads, however, the score plan is different. First of all, a download reported as malicious could be overrated, thus a manual check of the dynamic analysis is always required. Second, a download doesn't imply the machine is infected, only that the sample has been downloaded. Incidents relying only on malicious download need to be care-



MD5	Score
52e6b2db1c8ef5e4d3baea…	97
22fbed13b45bff3955910c6…	60
6c247016b19a847fc94680…	60
1adbd8efb5d7204c8accb8…	60
41948f132b68ca6e313885…	60
1d0d8520535e6ef17304a4…	60
6204058bf55a204cff23271…	57
e8f20b3df2503d19a09b31…	56
08eb912364977839156be…	53
3e385620457b88f21db4f9f…	53
5f7dc51d4d5e7e27cac7eb…	53
6638cfa82989d059c6977c…	53
ecd4b7e7cd6a6c185897d…	53
940ad463e4d36a4549eaaf…	51

Figure 5.2: Malicious downloads analyzed by `Lastline`

fully analyzed.

Figure 5.3 shows the number of infection events per week since the installation of `Lastline`. I immediately started analyzing events and infection, and deployed the first mitigation reports at the end of the week. The depression of the end of December is related to the Christmas holiday period. Back from vacation, the mitigation continued, keeping the infection under control. The peak of the first week of February was caused by a new malware, which spread across the network, causing a large number of events and incident. However, it was quickly contained and mitigated.

Although the number of events was reduced, it was never brought to zero, because of some installation and administration problems that both prevented me from identifying some infected hosts and prevented the `ACME Corporation` IT administrators from cleaning others.

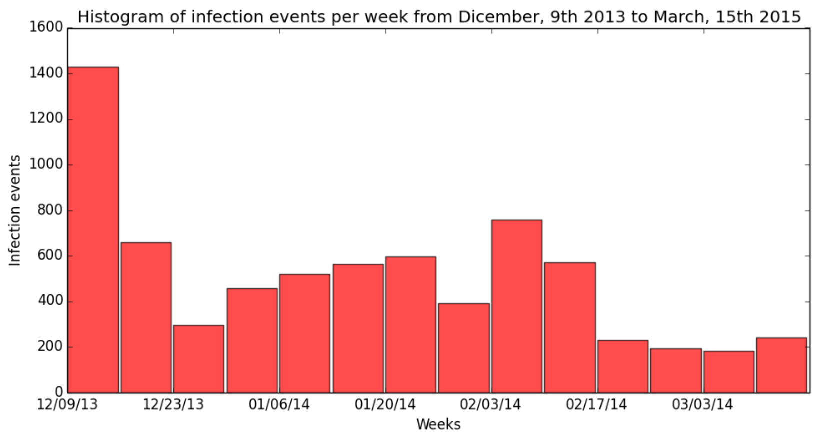


Figure 5.3: Number of infection events per time

5.1.3 Defence against targeted attacks

Most of my work was dedicated to clean the existing threats, but I wrote some `WASTE` detector to pinpoint suspicious downloads. I couldn't write detectors for emails because the `Lastline` mail analyzer was not activated. Also, I didn't perform analysis of network communication because of the same problem mentioned before: some network traffic sources were hidden behind some proxies and most of the hosts where managed by DHCP. Automatic detection based on `Lastline` events was hard to implement.

When checking for downloads, I focused on those identified by `Lastline` as being in order to find elusive malware. At first, I had to analyze the download statistic of `ACME Corporation` to understand which the habits were and how to reduce the samples to analyze. On average, `ACME Corporation` downloads 900 executables, where:

- 20% are updates from Microsoft and Google.
- 1% are downloads from known web sites (e.g. Skype).
- 50% are downloads from the internal `ACME Corporation` network.
- 15% are from the external `ACME Corporation` network.
- $10 - 12\%$ are downloads from external clients sites.
- $2 - 4\%$ are downloads from other websites.

Most of the downloaded samples came from trusted or internal network, which can be used in case of a targeted or insider attacks. Thus I separated `WASTE` detectors in three modalities:

- **Other websites:** `WASTE` fires warnings very often. These samples are in low number and came from a totally untrusted network.
- **Client external websites:** they can be a good entry point for targeted attacks; however their number is too high for a deeper analysis. Thus, `WASTE` raises warnings basing on narrow criteria.
- **Company websites:** the majority of downloads comes from here. Most of the samples are internal applications or libraries, which have similar characteristics and behavior. I programmed `WASTE` detectors to detect any anomaly, thus excluding most of the similar samples.

During my brief work as analyst, I didn't detect trace of targeted malware, however, `WASTE` detectors allowed me to identify some malware programs that weren't recognized by `Lastline`, or by antivirus programs.

5.2 Vulnerability Assessment

The vulnerability assessment was a relevant aspect of the work. By using `OpenVAS`, we detected known vulnerabilities or systems misconfiguration and performed a penetration testing.

5.2.1 OpenVAS

The Open Vulnerability Assessment System is a framework offering a vulnerability management solution[6]. The system is composed by clients (the `OpenVAS` user interfaces), services and data. There are three services:

- **Scanner:** the scanner executes a set of *Network Vulnerability Tests* (NVTs), which are tester for a single vulnerability.
- **Manager:** the manager manages the scan plans, organizes results and configurations and communicates with the clients.
- **Administrator:** the administrator permits the user and feed management.

Chapter 6

Conclusion and Future Research

Targeted attacks are an actual threat, for big and small companies. My work, along with other reports, demonstrates their potential risks: without the need of expensive founding an attacker can compromise the targeted company, by leaking sensible data or disrupting their services. The economic loss for the target can be very high.

Targeted attacks are capable of eluding conventional defence approaches, which rely on signature detection and follow a vulnerability-based taxonomy, without taking into account the entire attack process. Both academic and professional researches are studying the concept of targeted attack to provide new defence solutions. However, the real effectiveness of the proposed methods is hard to demonstrate.

My thesis defined an approach that can be used prior to the definition of a detection method. `HAZARD` and `WASTE` allow an immediate deploy inside a company, because they rely on existing defence systems, thus maintaining the same infrastructure. Once the process starts, the continuous testing and information exchange between the security team permits a company to enhance its cyber security by detecting threats and preventing some attacks, thanks to the proactive discovery of vulnerabilities and entry paths. Also, the process allows the definition of tests and monitoring to secure any infrastructure. For instance the security teams can easily deal with legacy server as well as SCADA systems and mobile devices.

`HAZARD` encourages the exchange of knowledge to improve the work of every team, so that any security flaw can be analyzed and fixed. Such knowledge can also be capitalized by an additional research and develop team, which can analyze the problem from a higher-level prospective to provide new systems and methods for defending against targeted attacks. In addition, the company could use the attack and analysis reports to prepare training program for the personnel.

I'm aware of the possibility of administrative limitations, for instance about privacy and classified information, or about any other requirement or problematic. However, `HAZARD` can be easily extended and adapted to handle these kinds of scenarios. Another problem is the excessive freedom given by `WASTE`. Nothing prevents the D-Team to write fully signature-based warnings, especially as a way to reduce false-positives. This modus operandi is counter-productive and can mine the analysis work. `HAZARD` tries to prevent this behavior with the H-Team work: if they can bypass the detection methods, the D-Team will be encouraged to improve or rewrite the detectors.

6.1 Future research

Future researches related to my work regards `WASTE` architecture, `HAZARD` feasibility, `HAZARD` effectiveness and new approaches based on `HAZARD` results.

`WASTE` doesn't have a proper architecture; it's only a conceptual framework. However, by defining suitable requirements followed by a problem analysis, `WASTE` could become a generic warning provider, capable of interfacing with any defence system and operate on any infrastructure.

`HAZARD` feasibility regards administration and organizational problems. The process still need to be fully deployed inside a company, with multiple people arranged in separated teams. Only then `HAZARD` could be redefined to increase its flexibility and adaptability.

As for every defence product, the effectiveness must be verified. `HAZARD` must be properly tested against a targeted attack. However, how to tell if a security system is secure against a targeted attack? A related research should answer this question, by defining external testing process capable of verifying the defence approaches.

Finally, the ultimate goal of `HAZARD` is to promote the study and the development of advanced strategies against targeted attacks. If security teams are allowed to create a common shared pool composed by attacks information, `WASTE` detectors and methods born from `HAZARD`, it can be used by the security researchers to better analyze the targeted attack problem and provide new methodologies to counter the threat.

Bibliography

[1] John Aycock. *Computer Viruses and Malware.* Springer Science+Busines Media, LLC, first edition, 2006.

[2] Matt Bishop and Matt Bishop. A taxonomy of unix system and network vulnerabilities. Technical report, CERT/CC, 1995.

[3] Bill Blunden. *The Rootkit Arsenal: Escape and Evasion in the Dark Corners of the System.* Jones and Bartlett Publishers, Inc., USA, 2009.

[4] Susan Cima. Sans - vulnerability assessment. `http://www.sans.org/reading-room/whitepapers/basics/vulnerability-assessment-421`, July 2001.

[5] Michael P. Collins, Carrie Gates, and Gaurav Kataria. A model for opportunistic network exploits: The case of p2p worms. In *WEIS*, 2006.

[6] OpenVAS Community. Openvas website. http://www.openvas.org/.

[7] Hervé Debar, Marc Dacier, and Andreas Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, 1999.

[8] Dorothy E. Denning. An intrusion-detection model. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, 13(2):222–232, 1987.

[9] Nitesh Dhanhani, Billy Rios, and Brett Hardin. *Hacking, The Next Generation.* O'Reilly Media, first edition, 2009.

[10] Manuel Egele, Theodoor Scholte, Engin Kirda, and Christopher Kruegel. A survey on automated dynamic malware-analysis techniques and tools. *ACM Comput. Surv.*, 44(2):6:1–6:42, March 2008.

[11] Verizon Enterprise. 2013 data breach investigations report. Technical report, Verizon Enterprise, 2009.

[12] FireEye. Advanced targeted attacks. `http://www2.fireeye.com/advanced-targeted-attacks-white-paper.html?x=FE_WEB_IC`, October 2013.

[13] Bingrui Foo, Matthew W. Glause, Gaspar M. Howard, Yu sung Wu, Saurabh Bagchi, Eugene H, and Contact Saurabh Bagchi. Intrusion response systems: A survey.

[14] Ernie Hayden. Assumption of breach: How a new mindset can help protect critical data. `http://searchsecurity.techtarget.com/tip/Assumption-of-breach-How-a-new-mindset-can-help-protect-critical-data`, March 2013.

[15] Kelly Jackson Higgins. How lockheed martin's 'kill chain' stopped securid attack, February 2013. `http://www.darkreading.com/attacks-breaches/how-lockheed-martins-kill-chain-stopped/240148399`.

[16] John D. Howard and Pascal Meunier. *Using a "Common Language" for Computer Security Incident Information*, chapter 3. John Wiley & Sons, 4 edition, April 2002.

[17] Galen Hunt and Doug Brubacher. Detours: Binary interception of win32 functions. In *Proceedings of the 3rd Conference on USENIX Windows NT Symposium - Volume 3*, WINSYM'99, pages 14–14, Berkeley, CA, USA, 1999. USENIX Association.

[18] Eric M. Hutchins, Michael J. Cloppert, and Rohan M. PH.D. Amin. Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. In *Proc. 6th Int'l Conf. Information Warfare and Security (ICIW 11)*, pages 113–125. Academic Conferences Ltd., 2010. `http://www.lockheedmartin.com/content/dam/lockheed/data/corporate/documents/LM-White-Paper-Intel-Driven-Defense.pdf`.

[19] K. Ingham and S. Forrest. A History and Survey of Network Firewalls. Technical report, University of New Mexico, 2002.

[20] Jay Jacobs. Ask the data: On opportunistic attacks, part 1. `http://www.verizonenterprise.com/security/blog/?postid=1587`, August 2012.

[21] Jay Jacobs. Ask the data: On opportunistic attacks, part 2. `http://www.verizonenterprise.com/security/blog/?postid=1589`, September 2012.

[22] Jay Jacobs. Ask the data: On opportunistic attacks, part 3. `http://www.verizonenterprise.com/security/blog/?postid=1593`, September 2012.

[23] Lt. Col. Lionel D. Alford Jr. Cyber warfare: A new doctrine and taxonomy. *The Journal of Defense Software Engineering*, 2001.

[24] Amit Klein. Multi-stage exploit attacks for more effective malware delivery. `http://www.trusteer.com/blog/multi-stage-exploit-attacks-for-more-effective-malware-delivery`, May 2013.

[25] Inc. Lastline. Lastline enterprise. `http://www.lastline.com/images/ds/Lastline_Enterprise.pdf`.

[26] Mike Lennon. Blackhole exploit kit author “paunch” arrested. `http://www.securityweek.com/blackhole-exploit-kit-author-paunch-arrested-reports`, October, 8th 2013.

[27] F. Li, A. Lai, and D. Ddl. Evidence of advanced persistent threat: A case study of malware for political espionage. In *Malicious and Unwanted Software (MALWARE), 2011 6th International Conference on*, pages 102–109, 2011.

[28] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16 – 24, 2013.

[29] Peter A. Loscocco, Stephen D. Smalley, Patrick A. Muckelbauer, Ruth C. Taylor, S. Jeff Turner, and John F. Farrell. The inevitability of failure: The flawed assumption of security in modern computing environments. In *In Proceedings of the 21st National Information Systems Security Conference*, pages 303–314, 1998.

[30] Kevin Mandia, Chris Prosise, and Matt Pepe. *Incident Response & Computer Forensics*. McGraw-Hill/Osborne, second edition, 2003.

[31] A. Moser, C. Kruegel, and E. Kirda. Exploring multiple execution paths for malware analysis. In *Security and Privacy, 2007. SP ’07. IEEE Symposium on*, pages 231–245, May 2007.

[32] Ben Nahorney. Connecting the dots: Downadup/conficker variants. `http://www.symantec.com/connect/blogs/connecting-dots-downadupconficker-variants`, April 2009.

[33] Tom Palmaers. Sans - implementing a vulnerability management process. `https://www.sans.org/reading-room/whitepapers/threats/implementing-vulnerability-management-process-34180`, June 2012.

[34] K. Podins, J. Stinissen, M. Maybaum (eds, and Angelos Stavrou). Towards a cyber conflict taxonomy. In *5th International Conference on Cyber Conflict*. NATO CCD COE Publications, Tallinn, 2013.

[35] M. Ramilli and M. Bishop. Multi-stage delivery of malware. In *Malicious and Unwanted Software (MALWARE), 2010 5th International Conference on*, pages 91–97, Oct 2010.

[36] Ravi Sandhu, David Ferraiolo, and Richard Kuhn. The nist model for role-based access control: Towards a unified standard. In *Proceedings of the Fifth ACM Workshop on Role-based Access Control*, RBAC '00, pages 47–63, New York, NY, USA, 2000. ACM.

[37] R.S. Sandhu and P. Samarati. Access control: principle and practice. *Communications Magazine, IEEE*, 32(9):40–48, September 1994.

[38] Karen Scarfone and Peter Mell. Guide to intrusion detection and prevention systems (idps). Technical report, National Institute of Standards and Technology, February 2007.

[39] R. Shirey. Internet security glossary, version 2. `http://tools.ietf.org/html/rfc4949`, August 2007.

[40] Chris Simmons, Charles Ellis, Sajjan Shiva, Dipankar Dasgupta, and Qishi Wu. Avoidit: A cyber attack taxonomy. 2005.

[41] Abhishek Singh and Yasir Khalid. Don't click the left mouse button: introducing trojan upclicker. `http://www.fireeye.com/blog/technical/malware-research/2012/12/dont-click-the-left-mouse-button-trojan-upclicker.html`, December 2012.

[42] Eugene H. Spafford. The internet worm program: An analysis. *SIGCOMM Comput. Commun. Rev.*, 19(1):17–57, January 1989.

[43] Natalia Stakhanova, Samik Basu, and Johnny Wong. A taxonomy of intrusion response systems. *Int. J. Inf. Comput. Secur.*, 1(1/2):169–184, January 2007.

[44] Olivier Thonnard, Leyla Bilge, Gavin O'Gorman, Seán Kiernan, and Martin Lee. Industrial espionage and targeted attacks: Understanding the characteristics of an escalating threat. In Davide Balzarotti, SalvatoreJ. Stolfo, and Marco Cova, editors, *Research in Attacks, Intrusions, and Defenses*, volume 7462 of *Lecture Notes in Computer Science*, pages 64–85. Springer Berlin Heidelberg, 2012.

List of Figures

List of Tables

Ringraziamenti

Questa tesi è potuta nascere solamente grazie al `Cesena Security and Network Applications`, nel quale ho potuto coltivare la mia passione per la sicurezza. Per il `CeSeNA` ringrazio Marco per averlo creato e averci seguito agli inizi e Luca per aver dato vita alla seconda generazione del `CeSeNA`. Infine ringrazio il professore Franco Callegati, per il suo supporto, la disponibilità e le sue lezioni sia accademiche che di vita (che spesso tendono a confondersi).

Insieme al `CeSeNA`, ringrazio anche il temuto `ANeSeC` e tutti i suoi componenti `alem0lars`, `esreverse`, `LMorl`, `Rez`, `stk` e `ZeroOFFset`. Grazie per il divertimento, il nerding e le serate trascorse a studiare e distruggere CTF.

L'ultimo ringraziamento security-related va alla `Room404`. Grazie ancora a Marco, Luca e Zoff e grazie a Stefano e Jake, ho imparato tanto a lavorare con voi.

Passando al mondo extra-security, ringrazio Pizza per il nerding, le serate al Mastro Birraio (che ringrazio vivamente) e le idee brutali che ne conseguivano. Ora è giunto il tempo di darle forma.

Ovviamente l'università non sarebbe la stessa senza il gaming spudorato e tutto ciò che ne consegue. In particolare grazie a Farne, Teo, Pizza e Robi per la compagnia e per le infinite serate passate a giocare a `D&D`.

Un enorme grazie al Babbo e alla Mamma, meglio nota come Dottoressa Agnese Treggiari, perché per 5+ anni mi hanno supportato, consigliato e creduto in me. Per questo ho dato il massimo. Grazie per tutto quello che avete fatto e quello che farete, con la speranza un futuro di poter ricambiare.

Ringrazio Marco e Claudio per le discussioni, il nerding, lo svago. È sempre divertente confrontarsi con voi.

Grazie al nonno Giuliano, di cui ho sempre ammirato la passione nel costruire e la dedizione per le nuove tecnologie. E grazie alla nonna Gilberta, per la sua cultura e voglia di scoprire e viaggiare. Vi ho sempre ammirato e mi avete continuamente ispirato a fare di meglio.

Un grazie allo Zio Adolfo (che ho finalmente raggiunto!), alla Zia Angela, lo zio Walter e alla Chiara.

Grazie anche a Massimo e Domenica, per il vostro sostegno e l'ospitalità. Grazie per non dimenticarvi mai di me. Grazie a Francesco e a tutto il tempo passato insieme, dalle battute ambigue ai giri sulla Buick.

Come nella tesi triennale, lascio il ringraziamento finale a Serena. Siamo ancora qui, abbiamo affrontato insieme Ingegneria e l'abbiamo sconfitta! Grazie per tutto quello che abbiamo potuto fare insieme. Studiare, guardare serie TV, giocare, uscire, divertirci. Grazie di tutto.