



Norwegian University of  
Science and Technology

# Classification of logs using Machine Learning Technique

**Edwin Giancarlo Vasquez Villano**

Master of Science in Telematics - Communication Networks and Networked

Submission date: June 2018

Supervisor: Maria Bartnes, IIK

Norwegian University of Science and Technology

Department of Information Security and Communication Technology



---

*This work is dedicated to my parents Benito and Maria, who always encouraged me since my childhood to study a master's degree outside Peru. Last but not least, to my whole family, whom always being on my side.*

---

---

---

---

# Problem Description

Nowadays, we live in a digital world. The use of computers, laptops, mobile phones, tablets, etc. has spread in any kind of business or people. In addition, majority of them have permanent connection to Internet, or are interconnected with other networks. However, in the last years, the number of automatic cyber security attacks has sharply increased, impacting negatively in business and people. Perimeter defense is no longer effective in a new scenario of ever growing attacks and rapidly changing patterns attacks. Therefore, today more than ever, it is needed to protect the data information and network infrastructure, using other security devices, such as antivirus servers, antispam devices, IPS (intrusion prevention system), authentication servers, application firewalls, and so on, all of them generating a huge amount of logs per second to be handled.

The implementation of SOC (Security Operation Center) is also an increasing new trend of companies nowadays, the main reasons is that SOC offers a permanent monitoring of the logs and, in case of any anomalous traffic, to detect as well as to prevent any security incident. However, one of the main challenges in the SOC, is analyzing the enormous quantity of logs in real time, making almost impossible for security engineers to inspect all of them, causing some undetected attacks.

In this master thesis, the main purpose firstly is to investigate about decision tree algorithm, and its implementation in WEKA software. The goal is to evaluate if the algorithm can predict an attack or not, after a training phase. Secondly, I will research about the processes of correlation and normalization of logs. I will select the most appropriate approach for the use with machine learning. I am going to work with real logs obtained from internet, which contains anomalous traffic and some identified attacks.

---

# Abstract

Currently, the use of information technologies is growing very fast in private or public companies. This is a worldwide trend, it is becoming needed than computers, printers, servers, cameras, etc. being interconnected between them and to Internet, in order to make the processes of the companies more effective and productive. Furthermore, the new trend of Internet of Things (IoT) is increasing this interconnection very fast. However, this trend is exposed unfortunately to cyber-attacks, every time more sophisticated and developed. Even worse, the detection of these attacks analyzing the logs of the security devices, is even more complicated, due to the enormous amount of logs that are generated per minute. This is a challenging activity in the Security Operation Center (SOC).

In this work, the use of a new emerging machine learning technology has been analyzed, in order to find if it can be applied for helping to predict new cyber-attacks. For this work, two sources of public logs has been used, in order to test the software. Furthermore, a new framework for the normalization and correlation processes has been designed. This process is explained in detail, and some images of the software used are shown.

Finally, some simulations have been carried out in a software, using a set of data for training and testing separately.

---

# Preface

This master's thesis is submitted to the Norwegian University of Science and Technology (NTNU) as the final part of the Master of Science in Communication Technology program at the Department of Information Security and Communication Technology. I would like to thank my responsible professor Maria Bartnes and supervisor Roy Myhre for valuable comments and guidance throughout this work. I would also like to thank to R. V. for her contribution to the correction of the document in the last weeks. The work was carried out from Jan 15 - Jun 11, in the spring semester of 2018.

Trondheim, June 11, 2018



Edwin Giancarlo Vasquez Villano

# Table of Contents

<b>Abstract</b>	<b>i</b>
<b>Preface</b>	<b>ii</b>
<b>Table of Contents</b>	<b>iv</b>
<b>List of Tables</b>	<b>v</b>
<b>List of Tables</b>	<b>v</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Figures</b>	<b>viii</b>
<b>Abbreviations</b>	<b>ix</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Objectives . . . . .	2
1.2 Scope and Limitations . . . . .	2
1.3 Thesis Structure . . . . .	2
<b>2 Background</b>	<b>5</b>
2.1 Security Operation Center . . . . .	5
2.1.1 In- house vs Outsourced SOC . . . . .	5
2.2 Main Challenges in the SOC . . . . .	6
2.3 Log message normalization . . . . .	8
2.4 Standard IDMEF . . . . .	9
2.5 Correlation of events . . . . .	11
2.6 Machine Learning . . . . .	11
2.6.1 Supervised Machine Learning . . . . .	12
2.6.2 Unsupervised Machine Learning . . . . .	12
2.6.3 Data Representation . . . . .	12



---

2.7	Decision Tree Algorithm . . . . .	13
2.7.1	Selection of problem for decision tree . . . . .	14
2.8	Confusion Matrix . . . . .	15
2.9	Entropy (S) . . . . .	17
2.9.1	Information Gain (G) . . . . .	17
2.10	Cohens Kappa Statistics . . . . .	17
2.11	Cross Validation . . . . .	18
<b>3</b>	<b>Method</b>	<b>21</b>
3.1	Deductive Research approach . . . . .	21
3.2	Flow Diagram . . . . .	22
3.3	Data Collection . . . . .	22
3.4	Data Analysis . . . . .	23
3.4.1	Exports the logs into MS Excel . . . . .	23
3.4.2	Data Reduction . . . . .	25
3.4.3	Data Normalization . . . . .	25
3.4.4	Data Correlation . . . . .	29
3.4.5	Grouping . . . . .	29
3.5	Simulation . . . . .	30
3.5.1	Files ARFF . . . . .	31
3.6	Limitations and Challenges . . . . .	32
<b>4</b>	<b>Experiment Setup and Results</b>	<b>37</b>
4.1	Description of the data-set . . . . .	37
4.2	Reduction of logs . . . . .	39
4.3	Simulation . . . . .	40
4.3.1	Scenario 1 . . . . .	40
4.3.2	Scenario 2 . . . . .	41
4.3.3	Scenario 3 . . . . .	43
<b>5</b>	<b>Discussions</b>	<b>53</b>
5.1	Software User Experience . . . . .	54
5.2	Recommendations . . . . .	54
<b>6</b>	<b>Conclusions and Future Work</b>	<b>55</b>
	<b>Bibliography</b>	<b>57</b>

# List of Tables

2.1	Outsourced SOC . . . . .	7
2.2	In-house SOC . . . . .	8
3.1	Number of logs from [1] after removing the duplicates ones. . . . .	35
4.1	Logs in the subdirectory http . . . . .	38
4.2	Logs in the subdirectory iptables . . . . .	38
4.3	Logs in the subdirectory snort . . . . .	38
4.4	Logs in the subdirectory syslog . . . . .	38
4.5	Super-events for scenario 1 . . . . .	41
4.6	Manual calculations of metric for scenario 2 . . . . .	44
4.7	Description of new dataset [2] . . . . .	45
4.8	Information about Training and testing dataset . . . . .	46

---

# List of Figures

2.1	Attributes IDMEF Format [3]	10
2.2	Instances for playing tennis [4]	13
2.3	Decision Tree Graphic [4]	14
2.4	Confusion Matrix	15
2.5	Four fold Cross Validation [5]	18
3.1	Deductive Researching [6]	22
3.2	Workflow of the thesis	23
3.3	Flow of Data Analysis	24
3.4	Wizard text to columns	24
3.5	Formula for identifying the zone	25
3.6	Wizard to remove duplicate data in the file access_log.1	26
3.7	Function ISNUMBER and IF	30
3.8	Logs correlated with different Attributes	31
3.9	Wizard Consolidate	32
3.10	Super-events extracted from [1]	33
3.11	Decision Tree Simulation options	34
3.12	Example of ARFF File	34
4.1	Logs in their default structure (Iptables)	39
4.2	File ARFF for scenario 1	42
4.3	Results in the experiment 1	43
4.4	Decision Tree for scenario 1 (size: 3, number of leaves: 2)	43
4.5	Attributes of scenario 2	44
4.6	Results scenario 2	45
4.7	Tree generated for scenario 2 (Number of leaves: 10)	46
4.8	Training data set	47
4.9	Text mode of the tree generated in the Scenario N <sup>o</sup> 3	48
4.10	Output scenario N <sup>o</sup> 3	48
4.11	Confusion Matrix Scenario N <sup>o</sup> 3	49

---

4.12	Testing dataset with the missing value of class attack . . . . .	49
4.13	Opening the testing data file . . . . .	50
4.14	Option for reevaluate the model with the testing dataset . . . . .	50
4.15	Predicted values for the file data8-test.arff . . . . .	51
4.16	Real and predicted values for class attack . . . . .	51

---

# Abbreviations

<b>ARFF</b>	Attribute Relation File Format
<b>AI</b>	Artificial Intelligence
<b>CMMI</b>	Capability Maturity Model Integration
<b>DTA</b>	Decision Tree Algorithm
<b>DMZ</b>	Demilitarized Zone
<b>IDMEF</b>	Intrusion Detection Message Exchange Format
<b>HTTP</b>	HyperText Transfer Protocol
<b>IDWG</b>	Intrusion Detection Working Group
<b>ID3</b>	Iterative Dichotomiser 3
<b>IIS</b>	Internet Information Service
<b>IoT</b>	Internet of Things
<b>IPS</b>	Intrusion Prevention System
<b>LAN</b>	Local Area Network
<b>ML</b>	Machine Learning
<b>NSM</b>	Norwegian National Security Authority
<b>PCI DSS</b>	Payment Card Industry Data Security Standard
<b>RFC</b>	Request for Comments
<b>SLA</b>	Service Level Agreement
<b>SIEM</b>	Security Incident and Event Management
<b>SNMP</b>	Simple Network Management Protocol
<b>SOC</b>	Security Operation Center
<b>VPN</b>	Virtual Private Network
<b>WAN</b>	Wide Area Network
<b>WEKA</b>	Waikato Environment for Knowledge Analysis
<b>XML</b>	Extensible Markup Language

---

# Introduction

Nowadays, cyber attacks to companies (small, big, public or private) are increasing every day. For instance, Wannacry ransomware is one of the latest large-scale cyber-attacks which has had a global impact. It infected hundreds of thousand computers in more than 150 countries in May 2017, encrypting the internal files of computers, and requesting payment in order to get the decryption key [7]. The Norwegian National Security Authority (NSM) states that this it has been the largest cyber-attack they have seen in large scale [8]. Among the victims were Norwegian hotels, British health care services, the Russian government, and a French car manufacturer.

The digitalization has transformed organizations to innovate using new technology and IT infrastructure. However, organizations, individuals, authorities and operations are susceptible to cyber-attacks. These kind of attacks have a great negative economic impact on businesses; targets are chosen based on potential profit. Other incidents are motivated by political issues, for example the group Anonymous attacked government websites from United States and Syria [9]. Easy access to hacking tools is one of the drivers for the increased threat. For this reason, it became more common in companies to implement themselves or to hire outsourcing services of Security Operation Center (SOC), in order to detect or prevent potential threats in real time.

One of the main responsibilities of SOC is to monitor continuously the security devices (firewalls, anti-virus servers, anti-spam, email servers, routers, switches, access points, etc.) from companies. The attack can be external (i.e. Wannacry) or internal (when an employee connects an infected pen drive to the local network). SOC plays an important role in companies in this increasingly interconnected world. In the past, traditional SOCs deployed individual software management tools, meanwhile SOC 2.0, emphasizes the use of SIEM (Security Incident and Event Management) software, a centralized collector of logs from different internal security devices.

SIEM is a platform whose main task is collecting all logs, processing and analyzing them, and displaying them on a screen in a graphic form, making understandable the detection of any security incident. Different brands such as Cisco, Fortinet, CheckPoint, etc have their own SIEM platforms, suitable for their own products, to some extent, compat-



ible with other vendors too. However, one of the main disadvantages of the SOC is the enormous quantity of log they receive per second, making impossible any human analysis and detection of threats in real time.

This thesis covers the study of two main functions of SIEM: normalization and correlation of logs. Furthermore, the current treatment and features (protocols, architecture, framework, techniques) as well as how these functions can be enhanced using Machine Learning (ML) are described in this thesis work.

## 1.1 Objectives

The goal if this master thesis is to study to what degree Machine Learning, specifically using Decision Tree Algorithm (DTA), helps with the prediction of attacks and identification of new security incidents. This leads to the following research question:

Could the Decision Tree Algorithm be used for the prediction of security attacks, through the analysis of logs?

The main objectives of this thesis are:

- To evaluate the performance of Decision Tree Algorithm through different set of experiments.
- Analyze the results given by the software of machine learning.

Specific objectives that have to be met in order to achieve the main objective are:

- To set up a procedure for correlation and normalization of logs.
- To familiarize with the use of the software machine learning proposed.

## 1.2 Scope and Limitations

Machine Learning is a broad term, and currently there are different algorithms. Decision Tree Algorithm was selected for this master thesis, because it has several applications in the real life. In [10] there is a list of academic articles published between 1993 and 1995, with real application of data mining using DTA , in different areas such as agriculture, astronomy, financial system and so on. Furthermore, the scope of this study is limited to the analysis of security logs obtained from the Honeynet Project [1] and other public security log sharing site. Both are an international, non-profit research organization dedicated to improving the security of the Internet at no cost to the public. The logs are collected from real systems, and contain some evidence of compromise and other malicious activity.

## 1.3 Thesis Structure

This thesis is organized as follows:

- **Chapter 2: Background** - provides general understanding about the concepts, technologies and algorithms that are the basis of this thesis. A literature review is also presented.
- **Chapter 3: Method** - summarizes the research methodologies, followed by a description and analysis of the dataset used in the simulation.
- **Chapter 4: Experiment Setup and Result** - this chapter discusses the experimental setup, the different metrics and scenarios used for our evaluation, as well as shows the findings of the simulation in the software.
- **Chapter 5: Discussion** - a discussion and analysis of the result obtained will be discussed in this chapter.
- **Chapter 6: Conclusion** This chapter includes concluding remarks and recommendations from the thesis. Suggestion for future work is also presented.



# Background

The aim of this chapter is to provide basic information about SOC, to summarize the main challenges and general knowledge about Machine Learning, and to describe the algorithm that will be utilized in the experimentation phase.

The following section includes an introduction about Security Operation Center, and its current challenges, as well as the description of standard IDMEF (Intrusion Detection Message Exchange Format), utilized mostly in Europe in the normalization process of logs. Lastly, it provides mathematical formulas in order to verify the results expected from the software machine learning during the simulation.

## 2.1 Security Operation Center

SOC is an important facility nowadays in companies. The purpose of the SOC is to monitor the different events or security logs from devices, including border security devices such as firewalls or IPS (intrusion prevention system), or different servers such as databases, authentication, mail, anti-virus systems, domain servers, etc. Each device must be monitored through the logs or events that they generate internally in each transaction. Typically, these logs are analyzed through sensors (IPS, Snort). If a sensor detects an alert, it can be a false positive or a real attack. Therefore, security analysts must make an internal analysis immediately, to establish whether it is a routine maintenance, or a malicious activity [11]. There are two types of SOC.

### 2.1.1 In- house vs Outsourced SOC

Several years ago, when the security attacks through internet started arising, the companies solved the problem buying traditional security devices, such as firewalls or antivirus software. However, over the years, the attacks have evolved everyday, making more difficult to detect a new one. For that reason more precise and efficient security devices are needed/required, in the different parts of the internal network (LAN, DMZ, WAN, Data-

center, WIFI, Servers, etc.). Consequently, maintaining a SOC can be costly, due to the following reasons:

- Specialized personnel should monitor the logs 24/7/365.
- Different specialized areas are needed in the SOC, for example Tier 1 (for daily monitoring), Tier 2 (for solving complex security issues), Forensic Analysis (research consequences of any incident), and so on.
- Security devices (firewalls, IPS antispam, webfilters, etc.) are expensive.
- Permanent training of staff: international standards, latest security technologies, learning new security technologies.
- High salaries for Security Analysts.

Therefore, it is becoming common for small or medium companies to hire an external SOC (Outsourced SOC), instead of implementing one internally. Logs are sent in encrypted form through VPN to the external SOC for further analysis. On the other hand, an in-house SOC is an internal facility in the company that focuses only on security of this company, and it is customized to the business needs. Normally, larger businesses such as banks, insurances companies, service providers, etc. deploy an in-house SOC. Table 2.1 and 2.2 shows the pros and cons of them.

## 2.2 Main Challenges in the SOC

Nowadays, due to the high complexity of computer attacks and continuous improvement, the SOC have great challenges. In [12] there is a detailed description of each challenge:

1. **Selection of Specialists.** Currently, SOC managers are presenting difficulties in selecting good candidates and retaining staff who already knows the different procedures and technologies used in the SOC. For this reason, during the selection process, manager should search professionals with a desire to seek new knowledge, curious about new forms of cyber-attacks. It is highly advisable to select someone who already has a background in IT (Helpdesk, Network administrators, etc.).
2. **Retention and career progress.** A huge amount of logs are generated in the SOC normally every minute, therefore the analysts should be monitoring and investigating whether there is an attack or it is a normal traffic. These activities are usually repetitive, and cause analysts to become bored at work. Furthermore, normally the schedules in the SOC are rotating. This explains why, the rate of personnel rotation is high. In order to counteract this, Managers should always encourage a good working environment, and promote analysts Tier 1 to the most advanced security groups, such as Tier 2, Audit, Forensic, Response Teams, etc. Contacts with other IT departments in the company also should be established, (Network Monitoring, Server Administration, etc.) in order to exchange professionals for some periods, and ensure a more rewarding career.

<b>Advantages</b>	<b>Disadvantages</b>
Cheaper than in-house SOC. Avoid capital expenses for the Company.	Internal employees know more the company instead of outsourcing people.
Usually, outsourced SOC deploy their own hardware and software in customers.	SLA (service level agreement) is high, for example make a simple reconfiguration in the fire-wall can delay 24 hours.
Highly specialized staff.	Multiple customers.
More experience in security incidents, monitoring tools.	Risk of leakage of sensitive information.
Unbiased	Normally lack of dedicated analysts to a single client.
Normally works 24x7x365	Log stored in a third-party company.
Usually, there are several customers from the same industry segment, giving more experience to the SOC.	

**Table 2.1:** Outsourced SOC

3. **Process and Procedures.** It is tedious to develop processes and procedures in the SOC in detail, because there are other time-consuming activities which demand big effort from specialists, especially in monitoring and solving cyber-attacks. Therefore, the lack of documentation of process or internal procedures is a common weakness in the SOC. In addition, is very prestigious for the company and the SOC, meets certain international standards (i.e. ISO 27001) or standards needed in order to work with a specific technology. For example, if the company issues credit cards, it should meet the standard PCI DSS (Payment Card Industry Data Security Standard). All these standards carry a large number of documents, such as manuals, procedures, checklists, etc. These documents are very important especially for the new members, during the training phase. Ideally, senior members should train the new ones, contributing to a transfer of knowledge. However a constant high workload can make it difficult. CMMI (Capability Maturity Model Integration) is an approach to improving processes and it can be a reference guide in the SOC. For most organizations, CMMI Level 3 is an appropriate goal [12].

4. **Huge amount of logs or events.** Normally, in order to avoid computer attacks on

<b>Advantages</b>	<b>Disadvantages</b>
Dedicated staff, who knows better the infrastructure than third-party companies. The software or SIEM platforms are customized for the business need.	Infrastructure and maintenance are expensive.  In some cases, the analyst and attacker can be colluded for performing any hide attack.
In some cases, the response can be fast. SLA usually short. Any request can be solved fast. Integration with other departments from the company.	Selection of specialized personal is hard.  High salaries for the SOC analysts.
Less risk of leakage sensitive information. Logs are stored locally.	High pressure for fast responses.

**Table 2.2:** In-house SOC

a network, the companies install several security devices such as Firewalls, IPS, antivirus systems, etc, which are monitored by SOC. These devices generate lots of logs. Moreover, there must be added the logs from Internet browsing, mail system, database, user authentication, Wireless Access, etc. These logs contain valuable information that must be analyzed in real-time to avoid security incidents, policy violations, fraudulent activities and operational problems [13]. For example, EMC Corporation can generate up to 1.4 billion logs daily, around 1 TB a day [14]. Therefore, analyzing these data presents a challenge to the specialists in SOC.

## 2.3 Log message normalization

There are different structures in the logs from the networking devices, because they use different protocols or they are from different vendors (Cisco, Fortinet, SNMP, NetFlow, Linux or Windows platforms). It is also a global trend that companies select the best security products not necessarily from the same vendor, i.e. HPE TippingPoint as IPS, Cisco ASA as firewall and Kaspersky for antivirus software in hosts (computers, servers).

This heterogeneous selection makes the customer feel very confident in using the best products of each supplier, in some cases based on their own job experience, or following the recommendation of Gartner Quadrant. However, all these different devices generate logs that do not necessarily contain the same structure or design. Since each solution has its own dashboard or management software, it can be difficult for the SOC to detect attacks in real time or to perform a forensic analysis. Therefore, the logs need to be standardized in a manner that allows the software SIEM to reflect them on one single dashboard. This process is called normalization. In [15] a series of rules for normalizing logs is proposed.

Some of them are used in this thesis:

- Removing punctuation, e.g., . ; : ? ! = - [ ] — ; , +, including the content between parentheses.
- Removing definite and indefinite articles, e.g., a , an , the.
- Removing weak words and prepositions: e.g., be, is are, of, at, such, after, from, being, do, done.
- Replacing all numbers by the word NUMBER.
- Replacing domain specific identifiers by corresponding words such as REGISTER or DIRECTORY.
- Replacing all dates by DATE.
- Replacing the upper-case letters by the corresponding lower-case letters.
- Replacing present participles, past of verbs e.g. FAILING, FAILED by the main verb e.g. FAIL.
- Replacing the plural noun e.g. ERRORS by the singular form e.g. ERROR.
- Deleting the name of the day, for example Tuesday, or TUE. It is enough 20/Feb/2005, in order to reduce the length of the message.

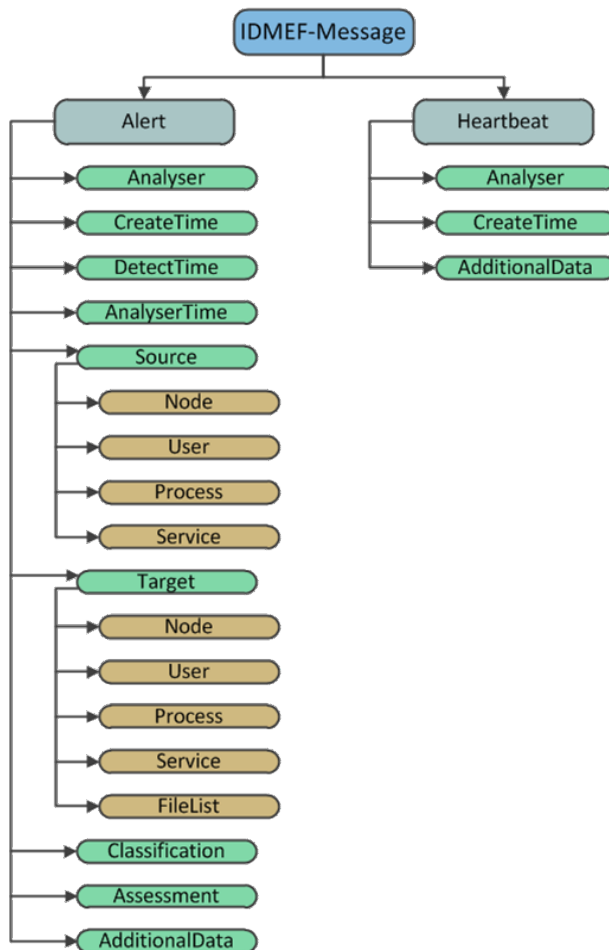
In addition, these previous rules may help to build a word dictionary, and to identify the main keywords of event descriptions. For the training of the ML, these keywords are very important, to identify an incident.

## 2.4 Standard IDMEF

Intrusion Detection Message Exchange Format (IDMEF) is a standard defined in the RFC 4765, published in 2007, which defines the data formats and exchange protocols for sharing information of interest to intrusion detection and response system and to the management systems that may need to interact with them. The details of the structure are described in [16]. This standard is used in information security for reporting or exchanging incidents with other parties, enabling inter-operability between commercial, open source and research systems. IDMEF has a good structured object-oriented format. It has 33 classes containing 108 fields. The implementation of IDMEF is in Extensible Markup Language (XML). This format was selected by the Intrusion Detection Working Group (IDWG) due to XML is being used for exchanging documents and data on Internet. Furthermore, XML-based applications are being used for different applications, such as electronic business cards, financial data exchange, web technology, weather observations, and so on. XML has no license fee.

In this project, this emerging standard was selected due to its inter-operability with different technologies. Furthermore, it has several attributes that will be used for the processes of correlation and normalization. Figure 2.1 shows the different attributes of the





**Figure 2.1:** Attributes IDMEF Format [3]

IDMEF format, for alerts and heartbeats. In this master thesis, all the logs belong to the alert category.

A few attributes were selected to use during the normalization process:

- CreateTime: The time when the alert was created.
- Source: The IP address of the origin of the traffic. It includes the sub-attributes Node, user, process and service.
- Target: The IP address destination of the attack, or device affected by the change described in the log. It includes the sub-attributes: Node, User, process, service and file list.
- Classification: This field describe which kind of attack is performing, for example:

ping of the death attack, tcp port scanning, etc.

However, more attributes were created in the Chapter three, for describing thoroughly the logs in this thesis.

## 2.5 Correlation of events

Correlation of events is to compare and to analyze the logs from different sources in one interval of time, to identify any common pattern or relationship. This practice helps IT staff to identify a security or failure incidents, reacting quickly to minimize negative business impacts and losses. For instance, it can detect an imminent attack if there are several login attempts to one device using one specific user, followed by scanning of ports in the LAN network using the same user. SIEM can mark these events as Curious and IT staff should check it. Performing this work manually would be tedious, therefore a SIEM and ML tools are needed. The advantages of correlation are [17]:

- Monitoring all security devices as a unit, and not in isolation. Full visibility of the entire system.
- Real time threats visibility, using a SIEM platform.
- Centralize all logs in a single console monitoring, improving visibility of trends, events and recurrences.
- Prioritizing all logs, maybe some logs may seem insignificant, but they may contain valuable information if it correlates with other logs, and can serve for preventing hacking attack.
- Speed in the detection of threats and attacks.
- Reduction of operational costs, because collection of logs is centralized, resulting in optimal number of specialists for monitoring different platforms.

Analyzing a single type of log, to generate an intrusion alerts, gives a high number of false positives and false negative. Therefore, it is evident that multiple types of log need to be considered. Correlating all different types of logs (from different sources) is more enriching, for detection of new attacks.

## 2.6 Machine Learning

Machine Learning (ML) is a part of Artificial Intelligence (AI), where the system is configured for learning and thinking like a human being. Initial training is given to the system, where the algorithm can learn the data and their classification, and the final purpose of the system is to make its own decisions (similar to a human being) in the future. ML works with certain degree of probability, based on the data that is analyzed and the decision that the system will adopt. The core of ML is predicting, and its ability is to predict future events based on past events. Currently, there are two types of ML:

## 2.6.1 Supervised Machine Learning

Supervised ML has more applications in the field of AI. This is one type of ML, which consists of working initially with a data set that includes both input and output. Supervised ML will build an algorithm that models this data set. While as large as the data set, the system will have more exact prediction for new values; it means the system performance will be more accurate. The goal is to make predictions for new, never seen before data, based on previous experiences. This type of ML needs a data set for training, and another data set for testing the model. Supervised ML includes two types of algorithms:

1. **Classification:** this type of algorithm should identify the major features or class level of each object, depending on the class to which it corresponds. It is widely studied in Psychology and Computer Science. Classification is sometimes separated into binary and multiclass. The former has only two outputs, for example if an email is a spam: yes or no, meanwhile the latter can have several outputs, for example identifying the language of one website (it can be any language i.e. Spanish, English, French, etc.).
2. **Regression:** these algorithms predict continuous-response values. For example, it can predict the price of a new house, in the range of 100, 000–200,000, based on the size, location, antiquity, and so on, or the salary of one person, based on education, job experience, city, etc.

Recognizing the difference between classification and regression is . simple. The former is a yes/no or positive/negative answer, or a specific language (English, Spanish, German, etc.). By contrast, the latter is a value, which has variation in the amount. For example, the ML can predict that the cost of the house is \$100,001 or \$100,002, but the real cost is \$100.005. If the algorithm starts deviating from expected results with new data, then the ML must return to the stage of learning algorithm, where the operator should provide more examples to the system.

## 2.6.2 Unsupervised Machine Learning

In this type of ML, one does not provide the system with a period of training with data set, just leaving the system to make their own decisions based on the configuration it has. This model is used for predicting unknown outputs. This kind of ML is still in phase of researching and testing.

## 2.6.3 Data Representation

In supervised ML, data set is shown as a table of instances. There are features (or attributes) along with a class. Figure 2.2, extracted from [4], shows an example of data set of fourteen instances, where each instance summarizes whether it is suitable or not play Tennis. The instances are D1, D2...D14. The attributes are Outlook, Temperature, Humidity and Wind. The Class is the column PlayTennis.

The idea of this data set is to predict if one specific day, for example Day N 15, is suitable for playing Tennis, based on different values of the attributes. The solution of this problem is in [4].

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

**Figure 2.2:** Instances for playing tennis [4]

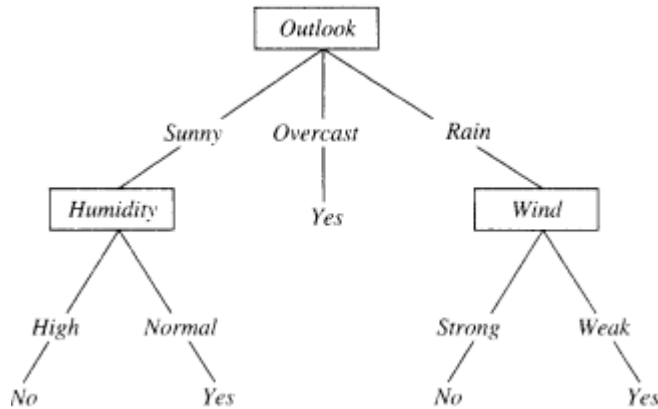
## 2.7 Decision Tree Algorithm

Decision Tree algorithm (DTA) classifies the instances by drawing the different attributes down the root. Figure 2.3 shows an example of DTA for the figure 2.2. Each branch represents a choice, possible value for this attribute; meanwhile each leaf represents a decision. The inference process starts at its root, and proceeds to a leaf. Each attribute is assigned to a node, and in the leaf there is a possible outcome (most probably state). It is used in supervised ML, both classification and regression.

The method for reaching a conclusion is through inductive inference, which means having a general conclusion from several examples. DTA concludes with a value for a dependent attribute (variable) or target function, based on different values of independent attributes (input). DTA are the most powerful approaches in knowledge discovery and data mining. It includes the technology of researching large and complex bulk of data in order to discover useful patterns. All theoreticians and specialist are continually searching for techniques to make the process more efficient, cost-effective and accurate.

Decision trees are highly effective tools in many areas such as data and text mining, information extraction, machine learning, and pattern recognition [18]. There are different algorithms for decision trees, such as ID3, ASSISTANT and C4.5. In this project, ID3 was chosen for this thesis because it can be simulated in the software WEKA. To identify which attribute was the root of the tree, it is needed to use a statistical formula, described in the next section.

ID3 algorithm is desirable for its simplicity, but it has several important disadvantages. ID3 chooses the best attribute and does not consider earlier choices. This can often lead to problems especially when the data set contains noisy data, for example, two records containing similar attribute values but different class labels. Then, in order to be more precise, the tree will grow too large trying to explain a simple noise in the data. This issue can be fixed in some way with pruning, where a whole sub tree is replaced by a leaf node



**Figure 2.3:** Decision Tree Graphic [4]

(nodes than does not have any child nodes), but the expected error will be greater than that of the single node [19].

### 2.7.1 Selection of problem for decision tree

According to [4], decision trees algorithm is best suited for problems which have the following characteristics:

- The problem is represented by attribute value-pairs. It has fixed attributes (I.e. origin of traffic: internal or external, or temperature: hot, mild or cool)
- The target function has discrete output values. In our model, the algorithm should predict if there is a security attack: yes or no. In the experimentation part, 1 means yes, and 0 means there is no attack.
- The training data may contain errors. Decision tree algorithm is robust to errors in classification of the training examples, and errors in the attribute values that describe the problem.
- The training data may contain missing attribute values. This algorithm can work even when there are some unknown values for the attribute. For instance, we cannot know exactly if the traffic is originated inside or outside of our LAN.

Many practical problems use this algorithm, for example for identifying faults in electronic devices, classifying medical patients by their diagnostic or issuing credit cards based on their likelihood to payments. More examples are in [10]. All these examples are categorizing as classification problems. Referent to this project, the goal is to identify an attack based on the security logs. Obviously, the output made by this algorithm, inevitably can include errors, increasing additional operation costs. These errors are categorized as follows [20]:

- The classifier gives an output attack when the traffic is normal (false positive).

- The classifier output is normal when the traffic is under attack (false negative).
- The inferred attack type is different from the actual attack (attack type misclassification).

Although all these errors should be minimized, it is more important to reduce the false positives, because they can trigger unnecessary activities, increasing the SOC budget and decreasing the reliability of the system.

However, it is also very important the false negatives, because the system is under attack. An attack can be easily inferred even with some false negatives. In the reference [21] there is the description of one attack, and it concludes that false positives should be take precedence, even though false negative have big impact too.

For this algorithm, it is important to select an effective attribute to be tested in each node of the tree. This goal is achieved using the Information-Gain criterion.

## 2.8 Confusion Matrix

A confusion matrix gives us information about actual and predicted values done by the machine learning algorithm. Performance of this algorithm is commonly calculated using the data in the matrix. Analyzing the confusion matrix give us a better idea if the classification is getting right and how many mistakes are present in the algorithm.

Given a classifier and an instance, there are four possible outcomes. If the instance is positive and it is classified as positive, it is counted as a true positive; if it is classified as negative, it is counted as false negative. If the instance is negative (not attack) and it is classified as negative, it is counted as true negative; if it is classified as positive, it is counted as false positive. The confusion matrix can be constructed representing the dispositions of the set of instances. This matrix forms the basis for many common metrics [22].

Figure 2.4 shows the confusion matrix, when the class is positive or negative.

		WEKA Confusion Matrix if is taken to be the positive class (Ex. There is an attack)	
		Predicted	
		Yes	No
Actual	Yes	True Positive TP	False Negative FN
	No	False Positive FP	True Negative TN

		WEKA Confusion Matrix if is taken to be the negative class (Ex. There is not an attack)	
		Predicted	
		Yes	No
Actual	Yes	True Negative TN	False Positive FP
	No	False Negative FN	True Positive TP

Figure 2.4: Confusion Matrix

Several standard terms have been defined from the confusion matrix [23]:

- The accuracy (AC) is the proportion of the total of number of predictions that were correct. It is determined by the equation:

$$AC = \frac{TP + TN}{TP + FP + TN + FN} \quad (2.1)$$

- The recall or true positive rate (TPR) is the proportion of positives cases that were correctly identified.

$$TPR = \frac{TP}{TP + FN} \quad (2.2)$$

- The false positive rate (FPR) is the proportion of negatives cases that were incorrectly classified as positive:

$$FPR = \frac{FP}{FP + TN} \quad (2.3)$$

- The true negative rate (TNR) is defined as the proportion of negatives cases that were classified correctly:

$$TNR = \frac{TN}{FP + TN} \quad (2.4)$$

- The false negative rate (FNR) is the proportion of positives cases that were incorrectly classified as negative.

$$FNR = \frac{FN}{TP + FN} \quad (2.5)$$

- F-measure (F) it a parameter that also shows WEKA in the output information. This metric combines precision and recall, and it is known as the harmonic mean.

$$F = 2 \cdot \frac{Precision \cdot Recall}{(Precision + Recall)} \quad (2.6)$$

Replacing the previous formulas:

$$F = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \quad (2.7)$$

- Finally, precision (P) is the proportion of the predicted positive cases that were correct:

$$P = \frac{TP}{TP + FP} \quad (2.8)$$

## 2.9 Entropy (S)

In information theory, Entropy is the measure of the uncertainty about a source of message. It gives us the degree of disorganization in our data. In one given set of data, it contains an arbitrary collection of examples. Then, it is separated the positive and negative samples. The formula is shown in (2.9).

$$Entropy(S) = -a \log_2 a - b \log_2 b \quad (2.9)$$

Where

$a$  = Proportion of positive examples.

$b$  = Proportion of negative examples.

For instance, if we have a data  $S$  of 16 samples, where 9 are positive ( $a$ ) and 7 negative ( $b$ ), the Entropy is calculated in (2.10).

$$Entropy(S) = -\frac{9}{16} \log_2 \frac{9}{16} - \frac{7}{16} \log_2 \frac{7}{16} = 0.909 \quad (2.10)$$

Noted that Entropy is 0 if all member of  $S$  belong to the same class or 1 if there are same amount of samples in each group. The function entropy varies in the range from 0 to 1.

### 2.9.1 Information Gain (G)

This concept measures the effectiveness of an attribute in classifying the training data, reducing the entropy. In order to reduce the decision tree graphic, the attribute with the high information gain is the best choice. Theoretically, the information gain of an attribute  $A$ , relative to a collection of examples  $S$  is defined in (2.11).

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|Sv|}{|S|} \times Entropy(S) \quad (2.11)$$

Where

$|S|$  = Number of elements in  $S$ .

$|Sv|$  = Number of elements in  $Sv$ .

$Values(A)$  = Set of all possible values of attribute  $A$ .

Noted that the first term of equation (2.11) is the entropy of the data set  $S$ . In the reference [24] there is one detailed example of how calculate these values.

## 2.10 Cohens Kappa Statistics

Sometimes in machine learning, in multi-class classification problems, the parameters such as accuracy, precision or recall do give total information about the performance of the algorithm. In this case, the parameter Kappa statistic is very useful and good measure



metric, which can handle multi-class and imbalanced class problems. This parameter always appears in the output of the software WEKA. The kappa coefficient (K) measures pairwise agreement among a set of coders making category judgments, correcting for expected chance agreement. It is defined in ( 2.12).

$$K = \frac{P(A) - P(E)}{1 - P(E)} \quad (2.12)$$

Where  $P(A)$  is the proportion of times that the coders agree and  $P(E)$  is the proportion of times that would be expect them to agree by chance, calculated along the lines of the intuitive argument presented above [25]. When there is no agreement, K is zero. When is total agreement, K is one. There is not standardized way to characterized values. However, [26] mention that the interval between 0-0.20 as slight, 0.21-0.40 as fair, 0.41-0.60 as moderate, 0.61-0.80 as substantial and 0.81-1 as almost perfect agreement (good reliability).

## 2.11 Cross Validation

In k-fold cross validation, the original dataset is subdivided randomly in k equal size sub-samples. Of the k sub-samples, a single one is retained as the validation data for testing the algorithm, and the other k-1 are used as training data. This process is repeated k times (the folds), where each of the k sub-samples is used only once as validation data. By default value of k in WEKA is 10. After, having done k-fold cross-validation with each sub-sample, and computed the evaluation results, the software invokes the learning algorithm a final (k+1) time on the entire dataset to obtain the model that it prints out in the summary section.

For example, if there is a dataset with 4-fold CV,  $\frac{3}{4}$  of the data would be used as training and  $\frac{1}{4}$  for testing, and it will be executed in total 4 times. The combinations are shown in figure 2.5 [5].

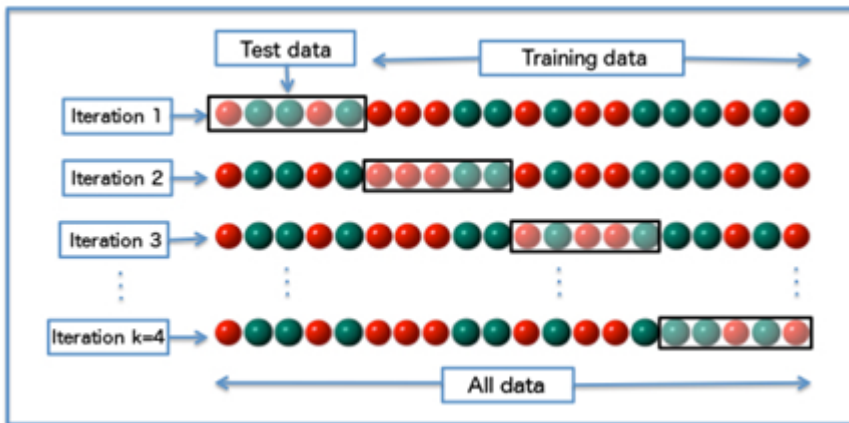


Figure 2.5: Four fold Cross Validation [5]

This generalization is usually better because it uses several validation data for testing the algorithm. In the software WEKA, this option is selected in the section Test Options.



# Chapter 3

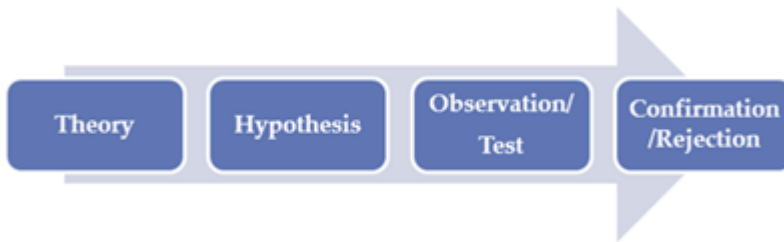
## Method

This chapter describes the methods used to answer the research question, and the reasons behind the choices made. Limitation and challenges regarding these methods are also described. This thesis is based on academic literature, industry whitepapers and various scientific publications. The method chosen to carry out this research is mainly literature review, followed by software experimentation, measurement and obtaining meaningful results. The goal of this thesis is to explore into the use of ML with security logs, therefore it is needed to use a machine learning software.

There are two types of approaches for research: Inductive and deductive. In a deductive research, a hypothesis is derived from existing theory and the empirical world is then explored, and data are collected, in order to test the truth or falsity of the hypothesis. An inductive approach is where the researcher begins with as few preconceptions as possible, allowing theory to emerge from the data [27]. Deduction begins with an expected pattern that is tested against observations, whereas induction begins with observations and seeks to find a pattern within them[28]. This thesis utilizes deductive approach, which is used in the traditional engineering instruction, beginning with general concepts and theories, to applying these concepts to the real applications.

### 3.1 Deductive Research approach

Deductive research approach explores a known theory or phenomenon and tests if that theory is valid in given circumstances. It has been noted that the deductive approach follows the path of logic most closely. The reasoning starts with a theory and leads to a new hypothesis. This hypothesis is put to the test by confronting it with observations that either lead to a confirmation or a rejection of the hypothesis [6]. Furthermore, deductive reasoning is thinking from the general to the particular, whereas inductive reasoning goes the opposite way. In the figure 3.1, demonstrates the direction of deductive research.



**Figure 3.1:** Deductive Researching [6]

## 3.2 Flow Diagram

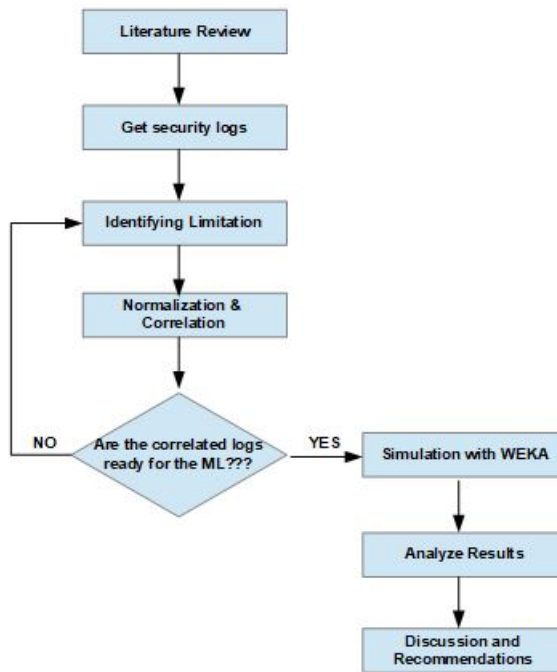
The flow diagram in figure 3.2 shows the overall workflow and the methodology used.

- Literature Review. This was the first step upon the selection of this topic. Different works related to this topic were reviewed and gaps were identified.
- Get security logs. Available for two public servers. Section 3.4 details this phase.
- Identifying limitation. When the logs are obtained from internet, it is important to categorize the logs and to select the utility of them. Furthermore, it was necessary to research the structure of logs, the sensor that generated them, and to choose an interval of time between them.
- Normalization Correlation. Section 2.5 describes this phase. The next chapter presents the results.
- Simulation. In this thesis, software WEKA was used to simulate machine learning (decision tree algorithm) either in training phase or testing phase. Section 3.3 provides further details.

## 3.3 Data Collection

Security logs are required for training the algorithm. There is one publicly available data source from the HoneyNet Project [1]. The HoneyNet Project is an international, non-profit research organization dedicated to improving the security of the Internet. Scan of the Month 34 (SoTM 34) logs have been widely used for testing IDS products. They contain extensive examples of attacks and background traffic. The information about the traffic and the attacks are provided in the following files:

- Apache web server log
- Snort Sensor log
- Linux syslog



**Figure 3.2:** Workflow of the thesis

According to [1], the LAN network of the Honeynet is 11.11.0.0/16 and the DMZ used is 12.12.0.0/16. The DNS servers of the network are in 22.22.22.\* and 23.23.23.\*.

On the other hand, the logs are needed for testing the model in the third simulation executed in the next chapter. Both datasets have almost the same structure, it means generated by the same sources. These were downloaded from [2].

## 3.4 Data Analysis

The analysis of the data was executed using the software Microsoft Excel Version 2010. The software assisted with normalization and correlation processes, creating rules and converting the log files into excel sheet, mainly making a separation using a space and colon (:). For working with the logs, the workflow corresponded the stages as shown in figure 3.3.

### 3.4.1 Exports the logs into MS Excel

The procedure was the same for both datasets. In total there were 42 files downloaded from [1]. All the files were opened with Notepad++, and then exported to Excel, separated by spaces and colons into columns, using the Wizard text to columns (Figure 3.4)). Then, the alerts were grouped by source IP address, port source, destination IP address, port

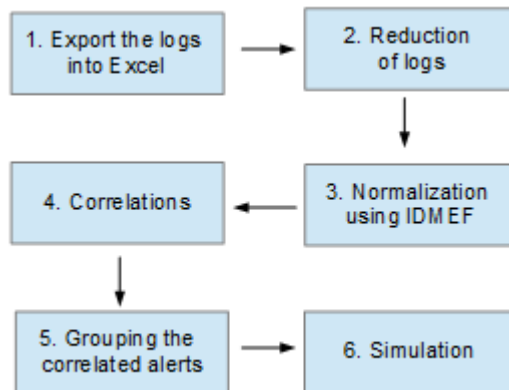


Figure 3.3: Flow of Data Analysis

destination, date, time and message. Further, it had to be identified if the Source and Destination IP address belong to the outside or inside zone.

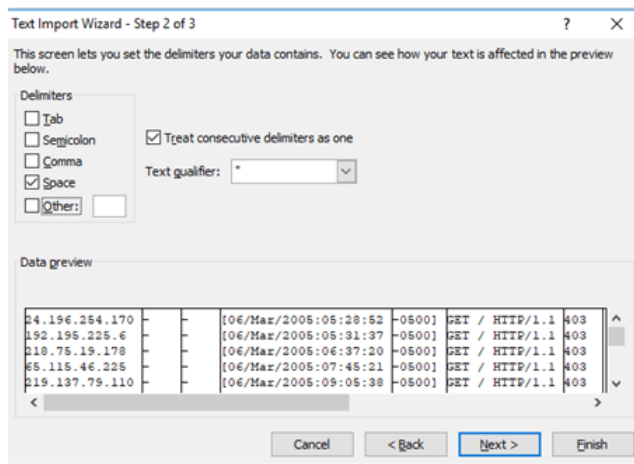


Figure 3.4: Wizard text to columns

To achieve this, the IP address columns were separated into four different columns (figure 3.5), separated by dots (using again the wizard text to column). In order to identify the zone, only the first two columns B2 and C2 (equivalent to the first two octets of the IP addresses section 3.3) had to be classified following the algorithm:

**Require:** B2, C2

**Output:** source\_zone

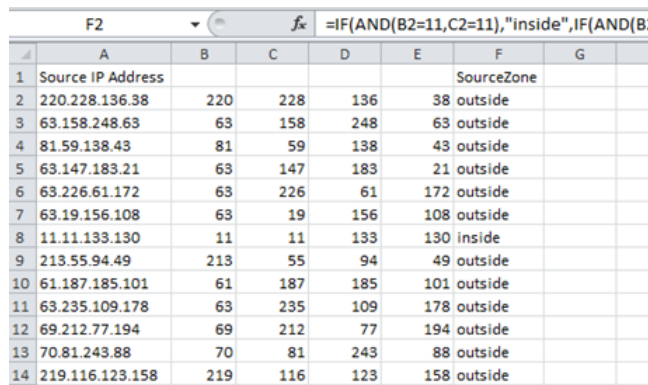
If (B2=11 and C2=11)

Then source\_zone= inside

*Else if (B2=12 and C2=12)*  
*Then source\_zone= outside*  
*Else*  
*Source\_zone=outside*  
*End if*

Therefore, the rule was created in excel file in the cell F2 (figure 3.5) and applied in the whole column:

`=IF(AND(B2=11,C2=11),"inside",IF(AND(B2=12,C2=12),"inside","outside"))`



	A	B	C	D	E	F	G
1	Source IP Address					SourceZone	
2	220.228.136.38	220	228	136	38	outside	
3	63.158.248.63	63	158	248	63	outside	
4	81.59.138.43	81	59	138	43	outside	
5	63.147.183.21	63	147	183	21	outside	
6	63.226.61.172	63	226	61	172	outside	
7	63.19.156.108	63	19	156	108	outside	
8	11.11.133.130	11	11	133	130	inside	
9	213.55.94.49	213	55	94	49	outside	
10	61.187.185.101	61	187	185	101	outside	
11	63.235.109.178	63	235	109	178	outside	
12	69.212.77.194	69	212	77	194	outside	
13	70.81.243.88	70	81	243	88	outside	
14	219.116.123.158	219	116	123	158	outside	

**Figure 3.5:** Formula for identifying the zone

Finally, in the column F from figure 3.5 the Source IP address is classified into outside or inside zone. This is important to note because in the chapter four, source\_zone serves as an attribute for the software. The same procedure is repeated for the destination address.

### 3.4.2 Data Reduction

The goal in this phase is to remove information that is not interesting, or alerts that are not appropriate for the simulation with WEKA software, like duplicate alerts or normal access to websites. In Microsoft Excel, identifying and eliminating duplicates logs was easy to perform by using the button Remove duplicates from Data tab (figure 3.6). Table 3.1 summarizes the logs from [1] that are left after removing the duplicate data

### 3.4.3 Data Normalization

In this phase, a linearized form of the standard IDMEF was used as template. Linearization provides better performance in terms of memory and processing compare to XML format described originally by the IDMEF format. Millions of logs are generated daily in security devices. Hence, in order to normalize the data, it was created in total nineteen attributes and one class attribute (which specify if it is an attack or not). Mostly, to calculate the



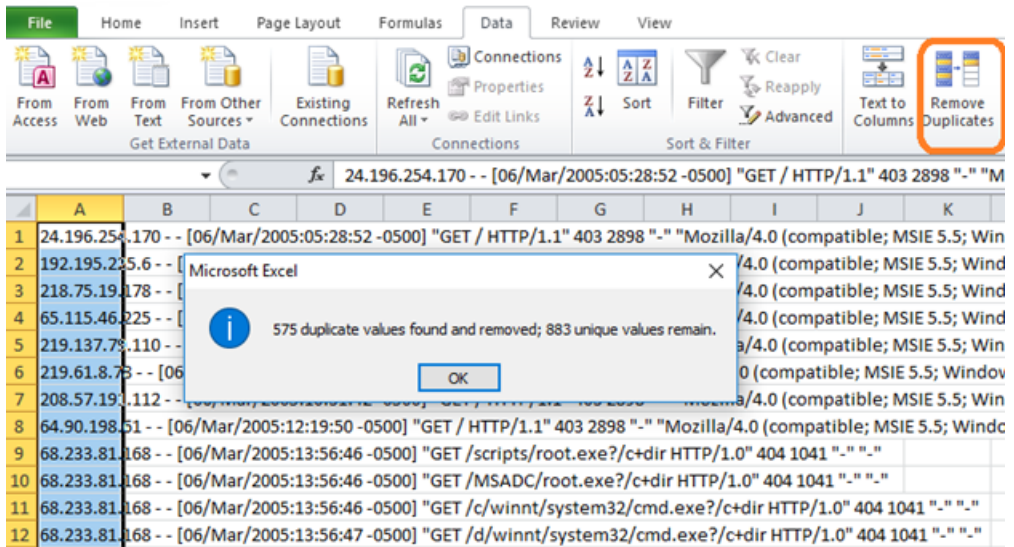


Figure 3.6: Wizard to remove duplicate data in the file access\_log.1

value of the attribute, the sum of these values of each logs involved in the super-event is calculated. Afterwards, the numeric value of the attribute in one super-event is the arithmetic sum of single logs values.

1. **Attribute ID Super Event.** Is the ID of every super-event. Super-event groups are several unitary logs extracted from the sources, with a common pattern. This attribute is only a reference; it serves for keeping an order in the logs, it is not used during the simulation with the software.
2. **Attribute source\_zone and dest\_zone.** They specify the source/destination of the traffic respectively. The values are inside, outside. However, in the experiment number 2, value 1 indicates outside, 0 indicates inside and NS is Not specified when the logs do not have any reference to this value. As an exception, the value DMZ was used for demilitarized zone in experimentation number one. These values helped to identify if the attack comes from Internet, or if it originated inside the red LAN. It should be emphasized that during the simulation, it was not important to specify the numerical value of IP address, only the zone where the traffic comes from. This is because the IP address varies every time, making it difficult to build a model. The goal is to be independent of these numerical values, since modern attacks uses normally fakes IP addresses.
3. **Attribute priority.** Each log extracted from the public servers has its own priority (1, 2 or 3), being the number 1 the most critical. If one log did not have a value, it is assigned a priority value of 3. Then, when the super-event is generated, the average of all these individual priorities is calculated, giving the priority of the super-event. This attribute is very important since a value of one means either a high

probability of attack or something anomalous is happening that is detected by the sensor. Therefore, SOC Engineers should react immediately.

4. **Attribute count\_access\_log.** Access logs are generated by the Webserver sensor. It records the web traffic connection outgoing and incoming. Therefore, this attribute is important, in order to detect if there is a connection or data download from any suspicious websites, trying to access to the LAN network. This attribute specifies the total number of logs generated, for every specific suspicious website or IP address.
5. **Attribute count\_error\_log.** These logs are generated also from the Webserver sensor. It detects if there is any error or anomalous traffic in the connection to any website, outgoing or incoming. Therefore, this attribute is crucial for detecting suspicious attacks. The attribute counts the number of error logs created in the access to one website. It can be generated several of them depending of the traffic.
6. **Attribute count\_snort\_log.** These logs are generated from the IPS sensor. It detects anomalous traffic in the content, not only in the source/destination IP addresses or protocol. Therefore, the quantity of logs generated for one single event is very important, because it can reflect that bad traffic is trying to access to the LAN network.
7. **Attribute interval\_time\_min.** This attribute specifies the range of time for the correlated logs for one super event. If there is an imminent attack, alerts are activated in different sensors, and in the majority of cases, they happen in a short period of time. For example, if there is an anomalous traffic at 1pm, and then other alerts appear after 12 hours in different sensors, they probably do not have the same cause. As long as the time interval is shorter, the probability of the logs being related to each other will be higher. The dimension is in minutes. It is used only in the scenario number one.
8. **Attribute count\_keyword\_attack.** This attribute specifies the number of keywords that appear in all the correlated logs in one super event. Keywords are words related to any kind of attack, for example: web-attack, awstats.pl (name of exploit), attack-response, information leak, worm propagation, bot scanning, etc. With these keywords, it is possible to create a dictionary attack. The more keywords the dictionary contains, the more likely it is to detect future attacks.
9. **Attribute awstat\_log.** Awstat.pl is a remote exploit that can execute commands remotely. This bug resides in the awstats.pl perl script. If the users send a command prefixed and postfixed with the character —, the command will be executed. During the examination of the logs downloaded from [1], it was found that this kind of exploit attacked permanently the system, generating lots of logs in different sensors. For calculating the value of this attribute, first it was necessary to count how many times the phrase awstats.pl appears in every individual log.
10. **Attribute cmd\_log.** it counts the number of request for the command *cmd.exe* that are in the logs. This command calls an internal process of the system, where the hacker can execute other commands, modifying files or even to shutdown the system. In summary, in some cases, these commands can harm the computer. Hence

it is important to have a register of these logs. For calculating the value of this attribute, it was counted how many times appear the phrase `get system32/cmd.exe` in every individual logs.

11. **Attribute root\_log.** it counts the number of request in the logs of the command `scripts/root.exe`. Some malware camouflages itself as `root.exe`, particularly when they are located in the `C:\Windows` or `C:\Windows\System32` folder, for example Trojan: Win32/Malat (detected by Microsoft), and TROJ\_DROPPER.NXE (detected by TrendMicro). For this reason, it is important to keep visible the number of request of this command in the logs. For calculating the value of this attribute, it was counted how many times the phrase `get scripts/root.exe` appears in every individual logs.
12. **Attribute worm\_count.** It counts the number of logs with the phrase *Worm propagation*. These logs mean that the sensor has detected an anomalous traffic. Therefore, it is needed to keep the register for each super event.
13. **Attribute Web\_attack\_count.** Similar to the previous attribute, it counts the number of logs with the phrase *Web application attack detected*. This phrase means that sensors have identified an anomalous traffic.
14. **Attribute backdoor\_count.** It counts the number of logs with the phrase *backdoor Trojan was detected*. This phrase means that sensors have identified an anomalous traffic which try to access to the system bypassing normal authentication.
15. **Attribute exploit\_count.** It counts the number of logs with the phrase *string exploit attempt*. This phrase means that sensors have identified an anomalous traffic which try to identify any vulnerability in the system.
16. **Attribute botnet\_count.** It counts the number of log which registered a botnet attack, trying to authenticate to the devices, using different usernames. Consecutive logs with phrases such as *attempt of login to 11.11.\*.\* as* or *failed password for* counts for this attribute.
17. **Attribute code\_count.** It counts the logs with the phrase *Classification Executable code was detected*. An executable file can contain a program that is capable of run a program in the computer. This program can be a virus, and it can harm the computer which makes it important to count these logs.
18. **Attribute illegal\_user\_count.** It counts the number of illegal authentication to the system; using different usernames are performed in short time.
19. **Attribute Information\_leak\_count.** The snort sensor can detect if there is an information leak to the outside. This attribute counts the number of log which contains the phrase *information leak detected*.
20. **Attack.** Finally this attribute specifies if the super event that appears in the table is an imminent attack, or not. In the context of this thesis, this attribute is the most important, because it has the classification that needs be predicted. The numeric value of the attribute in one super-event is the average of single logs values.

All these attributes are needed for creating the file \*.arff (Attribute-Relation File Format). These attributes were idealized after a detailed exhaustive analysis of the structure of logs, their messages and comparing with other similar online tools. It was possible to identify some phrases or texts which make reference to some kind of attack or anomalous traffic passing by the network.

### 3.4.4 Data Correlation

With the alerts reduced to a more specific data subset and normalized using the linearized IDMEF format, it is time for correlating the logs with other sources. For this process, all the files were unified in MS Excel, and all the logs together were added in only one sheet, for filtering all the remained logs and using as framework the attributes described in the previous section. Afterwards, the logs were consolidated, filtering by source IP address, destination IP Address, date and time. It means, the logs are correlated if they have the same source IP address, same creation date and the time of the events are less than 1 hour.

### 3.4.5 Grouping

This layer takes the alerts from the correlation stage, and grouped them into **super-events**. Super-events refer a single line in one row with all the attributes filled numerically (sum or average, depending on which attributes). These super-events are the input for the software in the next chapter. Its design fulfills the following requirements [29]:

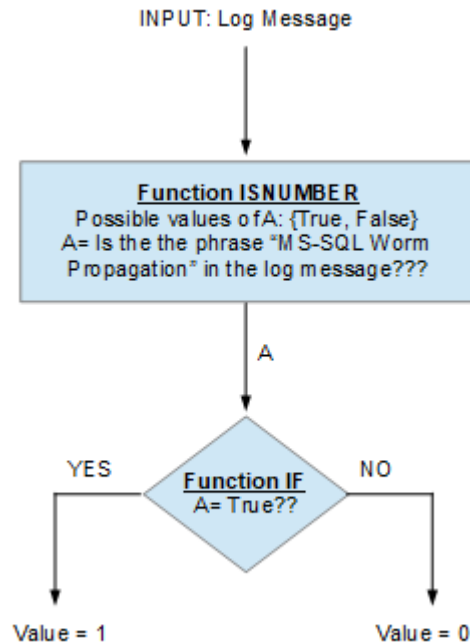
1. Efficiency: operate under the shortest subset of alerts attributes and limit searches to the super-events in time.
2. Intelligibility: The resulting super-events should be understandable to human being.
3. Coherence: clustering mechanism must adhere to the types of attacks supported by the model.

I decided to use a straightforward algorithm which grouped them based on the date and time of the event, and if they have the same source IP address, defined in the column source IP address. For instance, for calculating the attribute worm\_count (Column L in the figure 3.8), it was used a combination of IF and ISNUMBER function in the excel file, shown graphically in the figure 3.7. Then, the formula in the cell L2 is (applied to whole column) is:

```
=IF(ISNUMBER(SEARCH("MS-SQL Worm propagation",L2))=TRUE,1,0)
```

Once the attributes values of each individual log were ready, the calculation of the super-event value is using the wizard Consolidate (figure 3.9, located in the tab DATA. The option sum or average was selected in the function menu, depending on which attribute (section 3.4.3) it was being calculated.

At the beginning, all the individual logs obtained after the correlation (32958) were used for consolidate the data, obtaining in total 1985 super-events. In the figure 3.10 is shown an example of the output, after applying the wizard Consolidate. Noted that, the



**Figure 3.7:** Function ISNUMBER and IF

source IP addresses are now unique, not duplicated. This mean, that every row is a super-event. In order to train the machine learning, from the 1985 super-events created from the logs, 126 super-events were classified as attack (6.34%). This information was used in the chapter 4, in the scenario N<sup>o</sup> 2.

### 3.5 Simulation

The machine learning tool used in our evaluation is Waikato Environment for Knowledge Analysis (WEKA) [7]. WEKA is a suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. This software provides different ML algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. WEKA contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes. WEKA is open source software issued under the GNU General Public License [30].

The C.45 algorithm for building decision trees is implemented as classifier J48 (figure 3.11). The full name is `weka.classifier.trees.J48`. The classifier is shown in the text box next to the Choose button: It reads `J48 -C0.25 -M 2`. This text gives the default parameter settings for this classifier. C4.5 has several parameters, by the default visualization (when

O7												
								ATTRIBUTES				
A	B	C	D	E	F	G	H	I	J	K	L	M
Class	Source_IP_Address	Source_TCP	Dest_IP_Address	Dest_Port	date	Time	Message	Source_zone (1=outside, 0=inside, NS)	Dest_zone (1=outside, 0=inside, NS)	snort_count	MS-SQL Worm propagation	Priority
snort	02.99.177.20	4189	11.11.79.95	1434	25-Feb-05	5:14:00 PM	MS-SQL Worm propagation attempt [Classification Misc Attack] [Priority 2] [UDP]	1	0	1	1	2
snort	02.99.177.20	4189	11.11.79.95	1434	25-Feb-05	5:14:00 PM	MS-SQL Worm propagation attempt OUTBOUND [Classification Misc Attack] [Priority 2] [UDP]	1	0	1	1	2
snort	3.191.112.98	NS	11.11.79.67	NS	25-Feb-05	5:16:00 PM	ICMP PING NMAP [Classification Attempted Information Leak] [Priority 2] [ICMP]	1	0	1	0	2
snort	11.11.79.110	80	70.183.2.231	3563	25-Feb-05	5:16:00 PM	ATTACK-RESPONSES 403 Forbidden [Classification Attempted Information Leak] [Priority 2] [TCP]	0	1	1	0	2
snort	72.136.95.12	3129	11.11.79.85	1434	25-Feb-05	5:29:00 PM	MS-SQL Worm propagation attempt [Classification Misc Attack] [Priority 2] [UDP]	1	0	1	1	2
snort	72.136.95.12	3129	11.11.79.85	1434	25-Feb-05	5:29:00 PM	MS-SQL Worm propagation attempt OUTBOUND [Classification Misc Attack] [Priority 2] [UDP]	1	0	1	1	2
Access_log	21.234.48.24	NS	NS	NS	25-Feb-05	10:44:41	GET /scripts/root.exe?/c=dir HTTP/1.0	1	NS	0	0	3
Access_log	21.234.48.24	NS	NS	NS	25-Feb-05	10:44:42	GET /MSADC/root.exe?/c=dir HTTP/1.0	1	NS	0	0	3
Access_log	21.234.48.24	NS	NS	NS	25-Feb-05	10:44:44	GET /c/windows/system32/cmd.exe?/c=dir HTTP/1.0	1	NS	0	0	3

Figure 3.8: Logs correlated with different Attributes

you invoke the classifier) only shows: Confidence value (default 25%) - lower values incur heavier pruning and -M 2 Minimum number of instances in the two most popular branches (default value is 2) [31].

### 3.5.1 Files ARFF

An ARFF (Attribute-Relation File Format) file is an ASCII text file that describes a list of instances sharing a set of attributes. ARFF files were developed by the Machine Learning Project at the Department of Computer Science of The University of Waikato for use with the Weka software [32]. ARFF files has two section: Header and data. Fig 3.12 shows an example of ARFF files for the tennis example given in chapter 2.

#### Header Section

The ARFF Header section contains the name of the data, and attribute declarations. The relation name is defined in the first line of the file.

@relation *relation-name*

Noted that relation-name is a string, and it must be quoted if there are spaces in the name.

The attribute declarations are ordered in a sequence in the data section. It means that when the data is filled and separated by commas, each value between commas corresponds to one attribute, according to the declaration in the header section. Each attribute in the data set has its own @attribute statements. The order that the attributes are declared indicates the column position in the data set. The format for the attribute is:

@attribute *attribute-name*

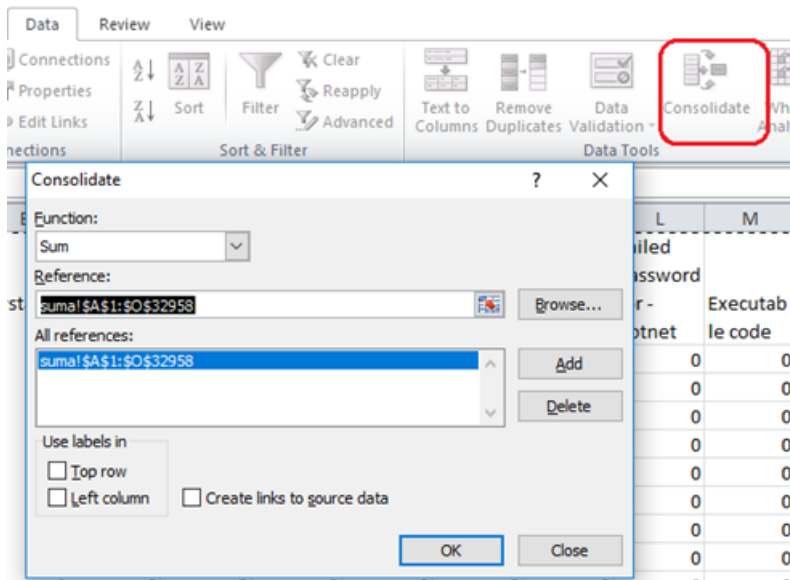


Figure 3.9: Wizard Consolidate

The datatype can be any of the four types:

- Numeric: can be real or integer numbers
- Nominal: it is specified the possible values, i.e. value1, value2, value3
- String: declaration of text values
- Date: specifies the date and time of the data set. The format follows the ISO8601 yy-MM-dd0 HH:mm:ss.

In this project were used numeric and nominal values.

## Data Section

In this part, each instance is represent in one single line. The values of each attribute are separated by commas, as it is shown in Figure 3.12. The order is very important, as it is declared in the header section. Data section starts with the declaration *@data*. In the reference [32] there is a detailed description about each section, as well as several examples.

## 3.6 Limitations and Challenges

The topic Machine Learning is huge. It involves different kind of software, algorithms, statistic models, probabilities, etc. Therefore, challenges related to writing a thesis in

	A	B	C	D	E	F	G	H	I
1			Attributes						
2	ID Super-Event	Source_Address	count_access_log	count_error_log	count_snort_log	awstats_log	cmd_log	root_log	worm count
3	1	5.199.231.23	1	2	3	2	0	0	0
4	2	198.54.202.4	1	2	3	2	0	0	0
5	3	02.155.10.13	1	1	1	0	0	1	0
6	4	03.112.195.15	16	13	0	0	14	1	0
7	5	09.120.201.17	16	13	38	0	14	1	0
8	6	09.201.25.17	23	23	55	0	23	0	0
9	7	10.118.169.2	387	410	0	0	0	0	0
10	8	10.51.12.23	69	98	0	69	0	0	0
11	9	1.200.172.16	16	13	38	0	14	1	0
12	10	1.191.134.23	0	0	12	0	0	0	12
13	11	3.159.48.144	0	0	1	0	0	0	0
14	12	3.26.231.138	0	0	2	0	0	0	0
15	13	19.197.76.95	0	0	2	0	0	0	2
16	14	52.33.100.15	0	0	2	0	0	0	2
17	15	13.93.149.22	0	0	2	0	0	0	2
18	16	53.17.18.143	0	0	1	0	0	0	0

Figure 3.10: Super-events extracted from [1]

twenty weeks may include time management and the restriction of the scope. Additionally, the topics considered in this project are quite new and relevant nowadays, so at the beginning it was difficult to limit the goals and scope of this thesis. Furthermore, part of the documentation or academic articles found in the internet, contains the point of view or comprehension of the authors, proposing their own framework for normalization or correlation, or different algorithms they used for testing. Difficulties may arise when performing the practical testing as well. Therefore, the results may be a bit biased. However, there is not a good or bad algorithm. Some algorithms can give better results with certain data sets, but worse with other, and vice versa. It depends on the size, quality and nature of the data. It depends on what we will do with the answer. It depends on the math behind the algorithm, and its translation into instructions for the machine learning software. Therefore, selection an algorithms depend on the experience, training and knowledge of the person responsible for the simulation/testing. After thoroughly research, DTA was selected due to its simplicity for understanding, the math behind the algorithm is understable and its usage in several real applications [10], and the software WEKA, due to its graphical configuration interface, simplicity and graphical outputs. In addition, there are several free online course in this tool, for different levels, supported by University of Waikato [33].

On the other hand, generating own attack logs include to set up properly isolated environment, to configure different virtual devices (firewalls, IPS, web filter, etc.) and to simulate random attacks. This would have exceeded the scope of this project. For that reason, working with public logs was the best option, which contained specific attacks. However, these logs are from the year 2005, and there may be some controversies about the age of the data set. In the approach used in this master thesis, the age of data set was not sensitive to the results.



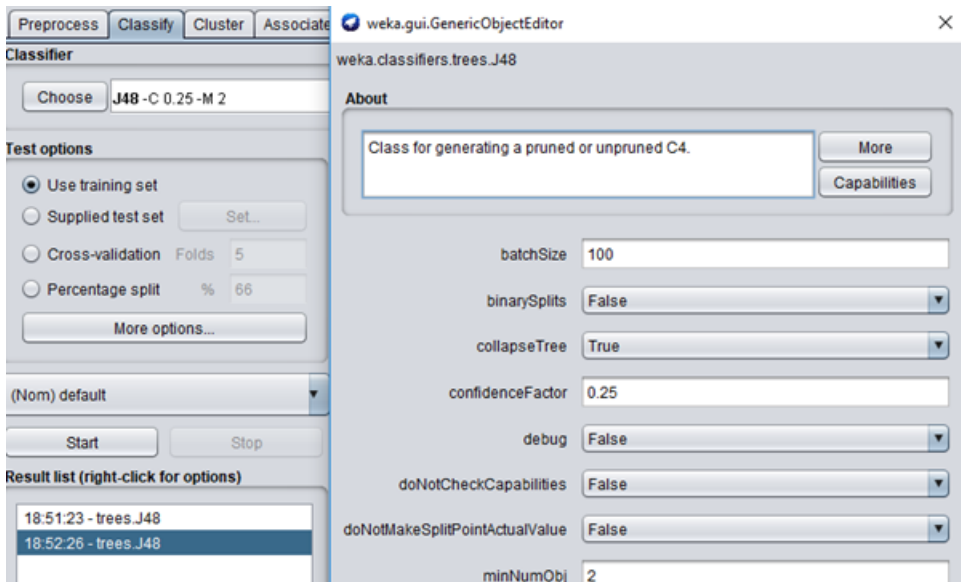


Figure 3.11: Decision Tree Simulation options

```

relation PlayTennis

@attribute day numeric
@attribute outlook {Sunny, Overcast, Rain}
@attribute temperature {Hot, Mild, Cool}
@attribute humidity {High, Normal}
@attribute wind {Weak, Strong}
@attribute playTennis {Yes, No}

@data
1,Sunny,Hot,High,Weak,No,?
2,Sunny,Hot,High,Strong,No,?
3,Overcast,Hot,High,Weak,Yes,?
4,Rain,Mild,High,Weak,Yes,?
5,Rain,Cool,Normal,Weak,Yes,?
6,Rain,Cool,Normal,Strong,No,?
7,Overcast,Cool,Normal,Strong,Yes,?
8,Sunny,Mild,High,Weak,No,?
    
```

Figure 3.12: Example of ARFF File

Folder	File	Number of logs	Duplicated logs	New Total
http	access_log.1	1457	575	882
	access_log.2	538	96	442
	access_log.3	26	31	229
	access_log.4	229	64	165
	access_log.5	406	66	340
	access_log.6	372	90	282
iptables	iptablesyslog	179752	73	179679
snortsyslog	snortsyslog	69039	1949	6790
syslog	maillog	105	0	105
	maillog.1	303	0	303
	maillog.2	203	0	203
	maillog.3	196	0	196
	maillog.4	81	0	81
	maillog.5	134	0	134
	maillog.6	150	0	150
	messages	150	0	150
	messages.1	195	1	194
	messages.2	127	0	127
	messages.3	127	0	127
	messages.4	180	0	180
	messages.5	256	5	251
	messages.6	189	0	189
	secure	166	0	166
	secure.1	234	0	234
	secure.2	256	0	256
	secure.3	149	0	149
	secure.4	211	0	211
	secure.5	96	0	96
secure.6	475	0	475	
			Total	255712

**Table 3.1:** Number of logs from [1] after removing the duplicates ones.



# Experiment Setup and Results

After the majority of main concepts have been presented in the previous chapters, the next step was to evaluate the data set in the software WEKA, and shows the results. Further analysis and discussion of the result will follow in the next chapter.

Furthermore, in this chapter is described in some detail the procedure of using the software. Some performance metrics such as accuracy, precision, false positive rates, etc. are calculated using the formulas from chapter 2. Two different dataset of logs extracted from internet were used in this chapter. The attributes has been defined in the previous chapter. A total of three scenarios with varying taxonomy were used for the training stage and one additional scenario for testing a new data. In this chapter, the main purpose is to model the data with DTA and to learn the use of software, in order to predict future attacks using common patterns. The processes of correlation and normalization are described at the beginning. It is identical in all experiments. Firstly, the reduction of logs is very important, since there are a considerable amount of logs in both dataset. They had to be reduced. Secondly, the linearized version of the IDMEF standard has been re-defined in a way to fit our specific context.

## 4.1 Description of the data-set

The first dataset was extracted from [1]. The task after download the logs were to examine inside of the zip folder. This could provide an idea about the data. The numbers of logs downloaded are shown in the tables 4.1 to 4.7

The number of log entries is really high. Therefore, it was needed to look at every folder inside, and correlated them for removing the duplicates ones.

It is noted from the tables 4.1 to 4.7, that the log files start with different date. I assumed that all the systems are synchronized. Therefore, a reasonable cross correlation of logs can start 25 February 2005 to 16 March 2005. However, the older entries of HTTP and syslog may contain useful information, but in order to simplify the analysis, this interval of time was considering for working with.

Files	Initial Date	Initial Time	Final Date	Final Time	Number of entries
Access_log	30/01/2005	4:34:59 AM	17/03/2005	11:38:27 AM	3554
Error_log	30/01/2005	4:33:18 AM	17/03/2005	11:38:27 AM	3692
Ssl_error_log	30/01/2005	4:33:18 AM	16/03/2005	01:01:27 AM	374
				<b>Total</b>	7620

**Table 4.1:** Logs in the subdirectory http

Files	Initial Date	Initial Time	Final Date	Final Time	Number of entries
iptables	25/02/2005	12:11:24 PM	31/03/2005	11:57:48 AM	179752
				<b>Total</b>	179752

**Table 4.2:** Logs in the subdirectory iptables

Files	Initial Date	Initial Time	Final Date	Final Time	Number of entries
snort	25/02/2005	12:21:33 PM	31/03/2005	11:49:38 PM	69039
				<b>Total</b>	69039

**Table 4.3:** Logs in the subdirectory snort

Files	Initial Date	Initial Time	Final Date	Final Time	Number of entries
Mail_log	30/01/2005	4:19:59 AM	17/03/2005	11:14:33 AM	1172
messages	30/01/2005	4:09:22 AM	17/03/2005	13:06:33 AM	1166
Secure	31/01/2005	6:16:51 AM	17/03/2005	12:59:00 AM	1587
				<b>Total</b>	3925

**Table 4.4:** Logs in the subdirectory syslog

## 4.2 Reduction of logs

The logs obtained from [1] are stored in the public server without any change in their structure. For instance, figure 4.1 shows logs from the firewall device (iptables).

```
Feb 26 18:55:24 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=219.146.168.94 DST=11.11.79.110 :
Feb 26 18:55:26 bridge kernel: INBOUND UDP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=202.99.159.2 DST=11.11.79.75 LEN
Feb 26 18:55:27 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=63.95.250.236 DST=11.11.79.73 LEI
Feb 26 18:55:27 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=63.95.250.236 DST=11.11.79.73 LEI
Feb 26 18:55:28 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=63.95.250.236 DST=11.11.79.73 LEI
Feb 26 18:55:55 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=218.206.148.200 DST=11.11.79.70 :
Feb 26 18:55:55 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=218.206.148.200 DST=11.11.79.70 :
Feb 26 18:56:04 bridge kernel: INBOUND TCP: IN=br0 PHYSIN=eth0 OUT=br0 PHYSOUT=eth1 SRC=218.206.148.200 DST=11.11.79.125
```

**Figure 4.1:** Logs in their default structure (Iptables)

These logs were normalized using linearized IDMEF structure. The reduction was done manually. After eliminating the duplicates logs, the following rules were applied [34] to the Microsoft Excel file:

1. Delete all the logs before to 25 February.
2. Delete all the logs that contain *default.ida error 404*. The request of this file is actually the result of the Code Red II worm, which have infected several public servers in Internet. This worm attacks the service IIS from Windows NT or Windows 2000. In this project, the web server is Apache. Therefore, I did not have to worry about it.
3. Delete all the logs with */NULL.printer scans with error code 404*. This is an IIS exploit for Windows servers. Not applicable in my project.
4. Delete logs which contain phrases */sumthin scans error code 404*. This scan was not considered, because it was assumed that there was not an identified target compromised. The method of this scan is to request a file from the server. Most default Apache configuration will send the version file within the error 404 messages body. Error 404 means that the client was able to connect with the server, but the server is not able to process the request.
5. Delete logs which contain *fp30reg.dll scan error 404*. This is an IIS windows exploit. Not applicable because the Webserver runs over Apache.
6. Delete logs which contain *nsiislog.dll scan error 404*. This is an IIS exploit. Not applicable in the Webserver because it runs Apache.
7. Delete logs *Get HTTP 1.0/1.1 error code 403*. These logs are ignored because examining the errors in the log files, I realized that the web server was configured not to serve as default page. Furthermore, error 403 means that the access to any specific page is strictly forbidden.
8. Delete normal logs of web surfing, connections to websites like yahoo, Microsoft, etc. Example:  
`220.170.88.36 25/Feb/2005:08:22:38 0500 "GET http://www.yahoo.com/HTTP/1.1" 403 2898 "" "Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)"`

9. Delete error logs which mentions file does not exist. These logs are not important for detecting any anomalous behavior in the traffic, because only they mention that one file is not found in the server. Example:  
*Tue Feb 22 11:04:40 2005 [client 211.59.0.40] File does not exist: /var/www/html/scripts*
10. Deleting logs which do not have any IP address (source or destination), because then they cannot be correlated. Example:  
*[Sun Feb 27 04:04:01 2005] [notice] Digest: generating secret for digest authentication*
11. Delete logs which record a simple web transaction, without any attack involved.  
*220.170.88.36 [25/Feb/2005:08:22:39 -0500] "GET http://www.yahoo.com/HTTP/1.1" 403 2898 "-" "Mozilla/4.0 (compatible; MSIE 4.01; Windows 98)"*

Once the logs has been reduced and normalized, they were exported to Microsoft Excel, in order to be correlated by the IP source address, date and time. In total, there were 32958 logs and 1985 super events.

## 4.3 Simulation

Now that the logs are correlated, it was time to execute three tests in the software proposed in order to know a bit more about DTA.

### 4.3.1 Scenario 1

In this first simulation, the goal was insert a few data to the software, and verify the result of WEKA. Furthermore, to learn a little more about the graphic that the software shows us and some parameters. After working with the logs, the table 4.8 was obtained with nine attributes and one additional (class attribute attack). In total there were 15 random super-events, extracted after the correlation process. This first small group of super-events were used in the software.

During the simulation, the source or destination IP address were used as attribute, because they might change frequently, and the firewall or web filter may block these IP address according to the blacklists configured or querying remote database. Furthermore, nowadays there are modern denials of service attacks which are launched from different infected IPs, or even attacks that are hidden or masked using other valid IPs, so is not valuable to make a thorough analysis based on IP address. Therefore, in this master work, it was focused the pattern of the attacks, the logs generated and the values of each attributes instead of the numerical IP addresses.

### Creation of ARFF file

For working with the simulation, the file prueba4.arff was created with the data from the table 4.5. The code is shown in figure 4.2

After running this script in the software, I got an accuracy of 80% (Figure 4.3), selecting the option of use training set in the test option. This option means that all the data input

Source zone	Dest zone	Priority	Count access log	Count error log	Count snort log	Interval time (min)	Count key-word	Attack
outside	inside	1	2	16	3	30	4	yes
outside	inside	1	13	164	0	30	4	yes
outside	inside	1	2	14	0	30	4	yes
outside	inside	1	219	201	0	30	6	no
outside	inside	2	21	48	0	28	3	no
inside	outside	3	10	20	5	56	1	no
outside	dmz	1	15	12	5	28	2	no
outside	inside	2	20	10	8	58	5	no
inside	dmz	3	9	9	4	36	3	yes
outside	inside	2	13	7	0	41	0	no
outside	inside	2	21	13	3	20	5	yes
outside	inside	1	9	9	0	19	2	no
outside	inside	3	8	8	4	31	8	yes
inside	dmz	3	15	2	0	24	6	no
outside	inside	3	12	5	6	51	5	no

**Table 4.5:** Super-events for scenario 1

is used for build the model and for generate the tree, and the learned rules. On the other side, there is another option which I could select for using 60% of the data for training, and the remaining 40% for testing, similar to cross validation method, but this was applied in the next scenario.

The tree generated by WEKA only took in consideration one attribute: count\_access\_log. This is because there were few data as input.

Since the attribute count\_access\_log is a continuous attribute, the algorithm discretized the range in two intervals: less or equal than 8 and greater than 8. From the graphic, it is understandable that if count\_access\_log attribute is higher or equal than this value, there is an attack, otherwise not.

### 4.3.2 Scenario 2

In this second scenario, all the attributes were used for analyzing in detail the confusion matrix and computing some metrics using the formulas from chapter two. The header section of the file data2.arff is shown in 4.5.

In the class attack, instead of using the words yes or not, numerical values were used. In this file data2.arff, there were 1985 instances in the dataset. All the super-events were used, in order to achieve better accuracy with the dataset, unlike the scenario 1, where only 15 instances were used.

Running the data into the software, were used the default values  $-C 0.25 -M 2$  (Confidence value = 25% and Minimum number of instances= 2). Furthermore, in this case, the test option cross validation was selected with default value of ten. This value is acceptable



```

1 @relation prueba4
2 @attribute zone_source {outside,inside,dmz}
3 @attribute zone_dest {outside,inside,dmz}
4 @attribute priority {1,2,3}
5 @attribute count_access_log numeric
6 @attribute count_error_log numeric
7 @attribute count_snort_log numeric
8 @attribute time_minutes numeric
9 @attribute count_keyword_Attack numeric
10 @attribute attack {yes,no}
11
12 @data
13 outside,inside,1,2,16,3,30,4,yes
14 outside,inside,1,13,164,0,30,4,yes
15 outside,inside,1,2,14,0,30,4,yes
16 outside,inside,1,219,201,0,30,6,no
17 outside,inside,2,21,48,0,28,3,no
18 inside,outside,3,10,20,5,56,1,no
19 outside,dmz,1,15,12,5,28,2,no
20 outside,inside,2,20,10,8,58,5,no
21 inside,dmz,3,9,9,4,36,3,yes
22 outside,inside,2,13,7,0,41,0,no
23 outside,inside,2,21,13,3,20,5,yes
24 outside,inside,1,9,9,0,19,2,no
25 outside,inside,3,8,8,4,31,8,yes
26 inside,dmz,3,15,2,0,24,6,no
27 outside,inside,3,12,5,6,51,5,no

```

**Figure 4.2:** File ARFF for scenario 1

for the number of super event were simulated.

Figure 4.6 and 4.7 shows the results after the simulation and the decision tree graphic respectively.

In order to verify the values of the results, table 4.6 shows the numerical calculations of the formulas given in the section 2.8 and the values in the confusion matrix.

All the values in the table 4.6 are equal to the figure 4.7. For the calculation of the kappa parameter, the total dataset contain 1985 instances. From the confusion matrix, there are 113 and 1853 true negative or true positives (the specific order is not important for this calculation). Thus, the proportionate agreement  $P(A)$  is:

$$P(A) = \frac{113 + 1853}{1985} = 0.9904$$

In order to calculate the probability of random agreement  $P(E)$ , from the confusion matrix:

- From the actual values, class one has 126 (113+13) instances. Thus, the proportion of class one is  $126/1985 = 0.0634$ .
- From the predicted values, class one has 119 (6+113) instances. Thus, the proportion of class one is  $119/1985 = 0.0599$ .

```

=== Summary ===

Correctly Classified Instances      12          80 %
Incorrectly Classified Instances    3           20 %
Kappa statistic                    0.5455
Total Number of Instances          15

=== Detailed Accuracy By Class ===

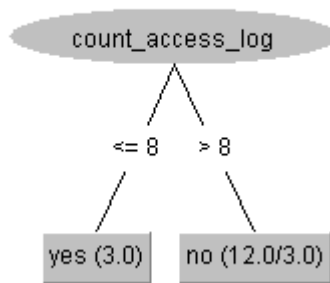
      TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
-----
0.500  0.000  1.000    0.500  0.667     0.612  0.750  0.700  yes
1.000  0.500  0.750    1.000  0.857     0.612  0.750  0.750  no
Weighted Avg.  0.800  0.300  0.850    0.800  0.781     0.612  0.750  0.730

=== Confusion Matrix ===

 a b  <-- classified as
 3 3 | a = yes
 0 9 | b = no

```

**Figure 4.3:** Results in the experiment 1



**Figure 4.4:** Decision Tree for scenario 1 (size: 3, number of leaves: 2)

Therefore, the probability of both would be class one randomly is  $0.0634 * 0.0599 = 0.0038$ , and the probability that both would be zero is  $(1-0.0634)*(1-0.0599) = 0.8805$

Overall random agreement probability is the probability that they agree on either class one or class zero. This result is similar from the software in the figure 4.7.

$$P(E) = 0.0038 + 0.8805 = 0.8843$$

Then, from the formula 2.12

$$K = \frac{0.9904 - 0.8843}{1 - 0.8843} = 0.9170$$

### 4.3.3 Scenario 3

In this section, new datasets for training and testing the algorithm are provided, extracted from other public log repository. Then, the percentage of accuracy, was calculated, com-

```

1 @relation data3
2
3 @attribute source_zone {outside,inside}
4 @attribute count_access_log numeric
5 @attribute count_error_log numeric
6 @attribute count_snort_log numeric
7 @attribute awstat_log numeric
8 @attribute cmd_log numeric
9 @attribute root_log numeric
10 @attribute worm_count numeric
11 @attribute web_attack_count numeric
12 @attribute backdoor_count numeric
13 @attribute exploit_count numeric
14 @attribute botnet_count numeric
15 @attribute code_count numeric
16 @attribute illegal_user_count numeric
17 @attribute information_leak_count numeric
18 @attribute priority numeric
19 @attribute 'attack' {1,0}
20
21 @data
22

```

**Figure 4.5:** Attributes of scenario 2

Parameter	When the class is positive: TP=113, TN=1853, FN=13	When the class is negative: TP=1853, TN=113, FP=13, FN=6
Accuracy	0.9904	0.9904
Recall or TPR	0.896	0.9967
False Positive Rate	0.0032	0.103
Precision	0.9495	0.993
F-Measure	0.922	0.994

**Table 4.6:** Manual calculations of metric for scenario 2

paring the prediction of attacks made by the machine learning and the real results. This public site contains various free shareable log samples from different systems, security and network devices, applications, etc. [2] (it has Linux logs, apache web server logs, error logs, snort logs and iptables firewall logs). Bundle N<sup>o</sup> 1 and Bundle N<sup>o</sup> 5 were selected, because their structure and sources are the same from the files in [1]. Therefore, all the previous steps could be repeated. Furthermore, in this section, the experimentation part was the main focus, and the analysis of results, instead of the correlation and normalization processes. However, a brief description of the files is shown in the table 4.7

In order to build the \*.arff file, the same attributes defined in the previous chapter were used. However, in the Bundle N<sup>o</sup>5, there are logs from different, so I filtered only the snort logs. In addition, for the whole data-set, I assumed that all the logs were generated in the same interval of time, because my goal for this part was to build a significant MS Excel table, with high numerical values in the attributes of each super-event. After the processes of normalization and correlation, I got in total 1537 super events, containing 507 attacks.

```

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances      1966           99.0428 %
Incorrectly Classified Instances     19           0.9572 %
Kappa statistic                     0.9174
Total Number of Instances          1985

=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                -----  -----  -
                0.897   0.003   0.950     0.897   0.922     0.918   0.969   0.873     1
                0.997   0.103   0.993     0.997   0.995     0.918   0.969   0.996     0
Weighted Avg.   0.990   0.097   0.990     0.990   0.990     0.918   0.969   0.988

=== Confusion Matrix ===

  a  b  <-- classified as
113 13 |  a = 1
 6 1853 |  b = 0

```

Figure 4.6: Results scenario 2

Bundle	Name of the File	Size(KB)	Number of logs
1	access_log.[1-31]	6430	43345
	error_log.[1-33]	4710	53813
	messages.[1-33]	2228	30128
	secure.[1-33]	1105	12474
5	2006log.2	100633	567871
	2006log.3	61767	357869
	2006log.4	46443	254242
	<b>Total</b>		1319742

Table 4.7: Description of new dataset [2]

Therefore, my strategy for this scenario was to divide the data set in two parts, according to the table 4.8

The simulation in the software for the training data-set was exactly the same as the procedure in the scenario 2. However, the graphic of the tree generated was too big (size: 141, number of leaves: 71). The file \*.arff created was named data8\_training. The attributes were the same from scenario N<sup>o</sup>2 (4.8).

However, it was still possible to see the decision tree in text mode in the window Classifier Output, under the section J48 pruned tree (Figure 4.9) shows an extract).

During the execution of the training dataset, there was an accuracy of 98.2% for the training test. The number of incorrect classified super-events was twenty two super-events. More details it is shown in the confusion matrix, detailed in the figure 4.11

The percentage of accuracy (AC) is:

$$AC = \frac{391 + 815}{391 + 11 + 11 + 815} = 0.9820$$

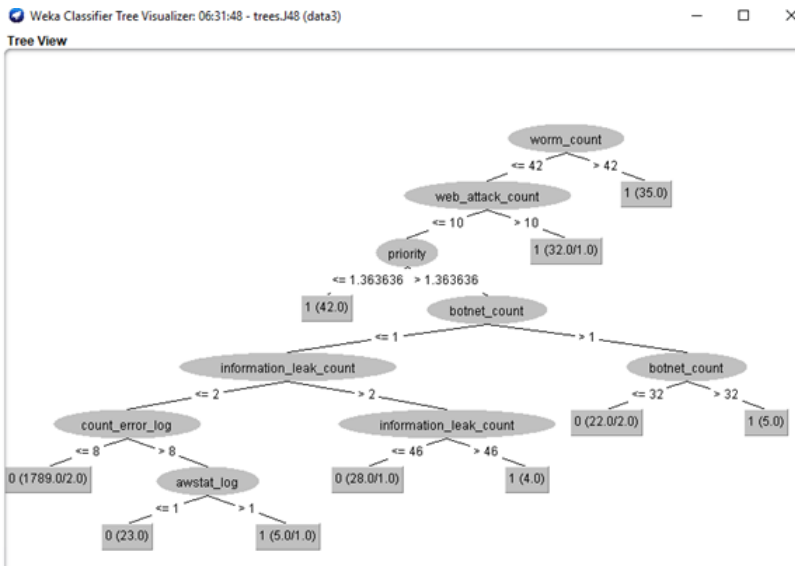


Figure 4.7: Tree generated for scenario 2 (Number of leaves: 10)

Type	Super Events (normal traffic)	Super events with attack	Total
Training data	826	402	1228
Testing data	204	105	309
		<b>Total</b>	<b>1537</b>

Table 4.8: Information about Training and testing dataset

The values of TP Rate, FP Rate, Precision and Recall can be calculated using the formulas from the section 2.8.

Once the model was learned from the training phase, it was time to run the testing data set. In this case, the data contained in total 309 super-events, with 105 identified attacks.

For testing the new dataset, the main requirement was that the file with the data to predict needs to have the same structure that the file used to learn the model. The difference is that the value of the class attribute is ? for all instances (question marks represent missing values in WEKA) [35]. Therefore, all the values of the class attack were replaced with closing question mark (?). Afterwards, once the values were predicted, they were compared with the real values, calculating the percentage of prediction of our model. Figure 4.5 shows the testing dataset. This file was called data8-test.arff.

Once the file was ready, in the tab Classify, in the test options window, the button supplied test set was selected, and the file data8test.arff loaded.

Finally, for re-running the model with the new data, select the option Re-evaluate model on current test set(Figure 4.14). The predicted results appear in the section Classifier Output window under the title Re-evaluation on test set, in the column predicted, after

```

1 @relation data@_training
2
3 @attribute source_zone {outside,inside}
4 @attribute count_access_log numeric
5 @attribute count_error_log numeric
6 @attribute count_snort_log numeric
7 @attribute awstat_log numeric
8 @attribute cmd_log numeric
9 @attribute root_log numeric
10 @attribute worm_count numeric
11 @attribute web_attack_count numeric
12 @attribute backdoor_count numeric
13 @attribute exploit_count numeric
14 @attribute botnet_count numeric
15 @attribute code_count numeric
16 @attribute illegal_user_count numeric
17 @attribute information_leak_count numeric
18 @attribute priority numeric
19 @attribute 'attack' {1,0}
20
21 @data
22
23 outside,1,2,3,2,0,0,0,3,0,1,0,0,0,0,2,0
24 outside,1,2,3,2,0,0,0,3,0,1,0,0,0,0,2,0
25 outside,1,1,1,0,0,1,0,1,0,0,0,0,0,0,2.333333333,0
26 outside,16,13,0,0,14,1,0,0,0,0,0,0,0,0,3,1
27 outside,16,13,38,0,14,1,0,16,0,0,0,0,0,22,2.194029851,1
28 outside,23,23,55,0,23,0,0,18,0,0,0,0,0,36,2.287128713,1
29 outside,69,98,0,69,0,12,8,5,14,15,1,5,2,13,3,1
30 outside,16,13,38,0,14,1,0,16,0,0,0,0,0,22,2.194029851,1
31 outside,13,135,29,13,0,0,0,27,0,13,0,0,0,2,2.683615819,1
32 outside,2,16,3,2,0,0,0,3,0,1,0,0,0,0,2.714285714,1
33 outside,23,23,0,0,23,0,0,0,0,0,0,0,0,0,3,1
34 outside,8,8,16,0,6,1,0,8,0,0,0,0,0,8,2.25,1

```

**Figure 4.8:** Training data set

the colon. The first column is the ID of each super-event of the testing file (Figure 4.15).

In the real data, the testing dataset contained 309 super-events, which 105 were classified as attack. On the other hand, the software predicted the value of the attribute attack (figure 4.16). Therefore, these values were exported to MS Excel File, and then I compared between them.

Figure 4.16 shows an extract of the table build with the two values: real and predicted (1= is an attack, 0= is not an attack). In summary, from 309 super-events, 266 super-events were classified correctly (yes). There were 43 mistakes. Therefore, the percentage of prediction (P) is 86.1

$$P = \frac{266}{309} = 86.1\%$$

This percentage is acceptable for the simulation. This value will increase as long as the model is feed with more data.

```

Classifier output

J48 pruned tree
-----
count_snort_log <= 8
| code_count <= 3
| | priority <= 2.2
| | | awstat_log <= 11
| | | | backdoor_count <= 0
| | | | | web_attack_count <= 0: 0 (349.0/3.0)
| | | | | web_attack_count > 0
| | | | | priority <= 1.8: 1 (6.0)
| | | | | priority > 1.8: 0 (17.0)
| | | | | backdoor_count > 0
| | | | | | root_log <= 0
| | | | | | | awstat_log <= 4: 1 (6.0)
| | | | | | | awstat_log > 4: 0 (2.0)
| | | | | | | root_log > 0
| | | | | | | | information_leak_count <= 8: 0 (59.0)
| | | | | | | | information_leak_count > 8
| | | | | | | | | exploit_count <= 14
| | | | | | | | | | illegal_user_count <= 13
| | | | | | | | | | | count_access_log <= 38: 0 (40.0)
| | | | | | | | | | | count_access_log > 38
| | | | | | | | | | | | botnet_count <= 6: 0 (5.0)

```

Figure 4.9: Text mode of the tree generated in the Scenario N° 3

```

=== Summary ===

Correctly Classified Instances      1206      98.2085 %
Incorrectly Classified Instances    22      1.7915 %
Kappa statistic                    0.9593
Mean absolute error                 0.0317
Root mean squared error             0.1258
Relative absolute error              7.189 %
Root relative squared error          26.8152 %
Total Number of Instances          1228

=== Detailed Accuracy By Class ===

          TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
          0.973   0.013   0.973     0.973   0.973     0.959   0.995    0.991    1
          0.987   0.027   0.987     0.987   0.987     0.959   0.995    0.996    0
Weighted Avg.   0.982   0.023   0.982     0.982   0.982     0.959   0.995    0.994

=== Confusion Matrix ===

  a  b  <-- classified as
391 11 | a = 1
 11 815 | b = 0

```

Figure 4.10: Output scenario N° 3

		<u>ACTUAL</u>	
		YES	No
<u>PREDICTED</u>	Yes	TP = 391	FP= 11
	No	FN= 11	TN= 815

Figure 4.11: Confusion Matrix Scenario N° 3

```

1 |relation data8_test_weka
2 |
3 | @attribute source_zone {outside,inside}
4 | @attribute count_access_log numeric
5 | @attribute count_error_log numeric
6 | @attribute count_snort_log numeric
7 | @attribute awstat_log numeric
8 | @attribute cmd_log numeric
9 | @attribute root_log numeric
10 | @attribute worm_count numeric
11 | @attribute web_attack_count numeric
12 | @attribute backdoor_count numeric
13 | @attribute exploit_count numeric
14 | @attribute botnet_count numeric
15 | @attribute code_count numeric
16 | @attribute illegal_user_count numeric
17 | @attribute information_leak_count numeric
18 | @attribute priority numeric
19 | @attribute 'attack' {1,0}
20 |
21 | @data
22 |
23 |
24 | outside,47,0,3,0,0,0,0,1,0,0,0,0,0,0,2,?
25 | outside,2,0,2,3,10,10,4,12,9,1,13,2,3,15,2,?
26 | outside,24,0,2,15,3,9,2,8,0,4,15,6,13,9,2,?
27 | outside,5,0,2,9,11,10,15,11,4,13,9,11,7,1,2,?
28 | outside,49,0,2,7,7,6,3,13,13,8,3,13,15,3,2,?
29 | outside,47,0,2,8,10,6,0,15,13,8,10,10,5,7,2,?
30 | outside,20,0,2,5,15,7,8,7,14,0,11,7,9,0,2,?
31 | outside,49,0,4,0,0,0,0,0,0,0,0,1,0,0,2,?
32 | outside,32,0,4,0,0,0,4,0,0,0,0,0,0,0,2,?

```

Figure 4.12: Testing dataset with the missing value of class attack



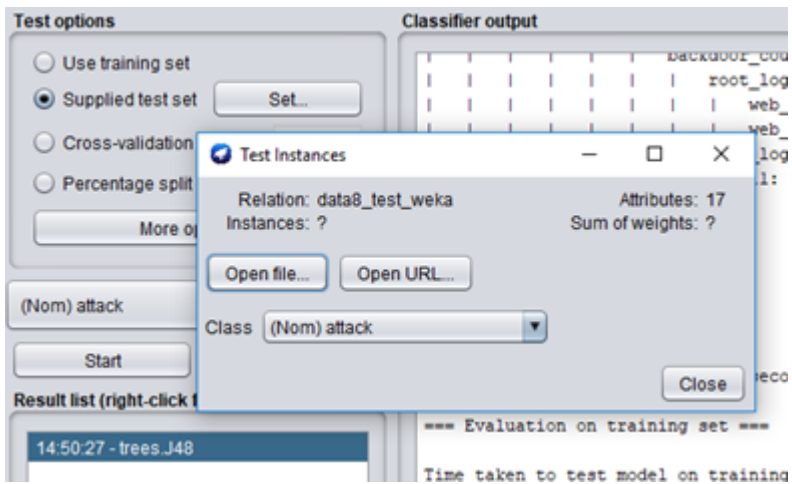


Figure 4.13: Opening the testing data file

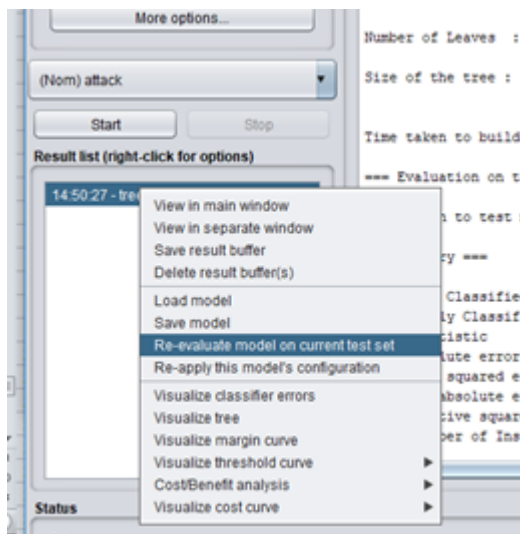


Figure 4.14: Option for reevaluate the model with the testing dataset

```

=== Re-evaluation on test set ===

User supplied test set
Relation:      data8_test_weka
Instances:     unknown (yet). Reading incremen
Attributes:    17

=== Predictions on user test set ===

inst#  actual  predicted error predicti
  1     1:??  2:0     1     1
  2     1:??  2:0     1     1
  3     1:??  2:0     1     1
  4     1:??  2:0     1     1
  5     1:??  1:1     1     1
  6     1:??  1:1     1     1
  7     1:??  1:1     0.833
  8     1:??  2:0     0.991
  9     1:??  2:0     0.991
 10     1:??  1:1     1
 11     1:??  2:0     1
 12     1:??  1:1     0.969
 13     1:??  2:0     0.991

```

Figure 4.15: Predicted values for the file data8-test.arff

	A	B	C	D
	ID Super-event	Real Value	Predicted value	Equal??
1	1	0	0	yes
2	2	0	0	yes
3	3	0	0	yes
4	4	1	0	no
5	5	1	1	yes
6	6	1	1	yes
7	7	0	1	no
8	8	0	0	yes
9	9	0	0	yes
10	10	1	1	yes
11	11	0	0	yes
12	12	1	1	yes
13	13	0	0	yes
14	14	0	1	no

Figure 4.16: Real and predicted values for class attack



## Discussions

This chapter discusses the findings from chapter 4 and establishes the link with the research question. This thesis builds on utilizing two sources of external logs, establishing a framework for the correlation and normalization processes, and defining the attributes that would help to fit the data in the software. The simulations were run with the algorithm (Decision tree), due to its practical application in the real world. However, nowadays there are other algorithms such as Bayesian network, linear regression, learning vector quantization and so on, which may have advantages or disadvantages depending on the type of data we are working with and the expected results. Selecting the right algorithm can be challenging even for experienced users in machine learning.

Setting up the experimental test bed has been the biggest achievement of this thesis. Designing the framework for correlation and normalization processes was the first challenge, because similar software available on internet neither shows in detail the treatment of data nor permits to modify selected attributes for experimental testing. In addition, the majority of such software have license fee; therefore the free version has few features available. The main focus in the three scenarios of the experimental setup was the performance of the software with different datasets. This study confirms the importance of researching the connection between security incidents and machine learning. In a real environment, thousands of logs are generated and then visualized in SIEM, making it impossible for human beings to analyze all of them, and to predict future cyber-attacks. Therefore, SOC are mainly reactive, and not proactive.

In the first simulation, there was accuracy of 80%, which seemed a reasonable response. However, it was noted that only one attribute was used for modeling. This is because the input data to WEKA was small (only fifteen super-events). Therefore, WEKA could not perform more calculation. Furthermore, the decision tree graphic was simple and small due to the few input data. This graphic can increase its complexity as the input data increases.

In the second simulation, the processes of correlation and normalization were explained in detail. These processes were tedious at the beginning, due to the huge amount of logs. It was needed a Windows computer with 16GB RAM to execute the filters in

Microsoft Excel. In some cases, these processes took more than 1 hour. At then end, a considerable group of super events was obtained. High percentage of accuracy in the algorithm was obtained, more than 95%. During the simulation, all the available data was used for training the model, so it helped using the option ten cross validation, where the same software split up the data set in training and testing data. After 10 rounds, an accuracy of almost 99% was achieved, and 9 attributes out 19 were used for creating the tree. This result was unexpected but it helped validating the simulation, as it indicated that the algorithm found a certain pattern in the input super events. Furthermore, the graphic generated was understandable and visible, showing the values calculated by the software and the behaviour in the classification of super events.

For the scenario three, the time processing was really high, taking more than 6 hours to execute the logs. The training phase in the simulation three resulted in an accuracy of 98.2%, acceptable with the test of 1302 super-events. Then, in the testing phase, after uploading the test dataset, the percentage of prediction decreased slightly, to 86.1%, but the model guessed a total of 86 attacks out of 105 attacks correctly.

Once the results were obtained from both simulations, it was possible to positively answer the research question. The results of the experiments confirmed that it is possible use machine learning techniques for predicting cyber-attacks. Conclusively, it is evident that some types of cyber-attacks follow a certain pattern of behavior when they are detected by the security devices, and the respective logs are generated. The results are more precise when the data input is bigger.

## 5.1 Software User Experience

The graphical interface of the software creates a friendly environment for the user. The options for selecting different algorithms are helpful. However, it was notorious during the simulation that the options in the software are not intuitive. There is no correlation between the name of the option and its meaning. When one invokes the classifier the first time, not all the parameters are shown on the main interface, which can confuse new users.

## 5.2 Recommendations

To achieve better results and to expand the scope of this thesis, the following improvements could be considered:

Firstly, implementing automatic monitoring and detection mechanism in the LAN network, and sending all the events generated directly to the ML software. This would save time in the correlation and normalization processes. Definitely, it would be great if the ML tool also has a SIEM module embedded into it.

Secondly, educating SOC operators in the management of the SIEM and the additional modules of machine learning. Frequently, SOC's manager do not use these additional tools already included in the software, because there is no specialized staff, even tough the cost of licence is high.

## Conclusions and Future Work

In this thesis, after several simulations with WEKA, it has been proved that it is possible to use a machine learning for detecting new attacks, because the DTA found certain patterns in the super events feed into the software. However, the thesis focused only on few exploits and force brute authentication attacks. Nowadays, there are lots of different cyber-attacks, excluding their variations. Providing more data to the algorithm, in the training phase, would provide greater precision for future modeling. Furthermore, creating an own environment, with different simulated attacks, and variations on these anomalous traffic, may increase the accuracy of prediction in machine learning.

Information security remains an unsolved challenge for organizations, because every day brings new and sophisticated cyber-attacks. The traditional approach to security that attempted to counteract this situation, is to install and deploy several security box. However, this perimeter defense proves not effective in a scenario where the attacks constantly change their pattern.

As the threats to the security increase, there is a need for a flexible and dynamic approach, that can react faster to security incidents. It is necessary to reduce the time for detection and analysis, and to tackle effective countermeasure to any possible attack. Machine learning can provide techniques that can help effectively solving these challenges.

In turn, training machine learning is a continuous learning process. SOC engineers should be responsible to feed this database constantly, in order to reach ration prediction higher. As part of future work, the following alternatives are suggested:

- Using a management software database as Oracle or MS-SQL for the treatment and filtering of logs, instead of MS Excel. This could generate a greater speed when filtering the data, especially when the data is large. Furthermore, filtering or deleting can be executed using simple queries. For instance, [35] provides an example of some commands used in the iptables database, previously exported in SQL.
- For the simulation of ML, use other software like Matlab or Python. This software has additional modules that allow simulating different algorithms. It would be great to compare the results of them with the results of WEKA, in order to identify if the

results are independent of the simulation platform.

- Creating an isolated environment, with various security devices of different brands, where different controlled attacks and their variation can be generated, and feed this data to the software, in order to get better accuracy.

# Bibliography

- [1] Anton Chuvakin. Scan34, <http://old.honeynet.org/scans/scan34/>. Last Accessed: 2018-11-03.
- [2] Anton Chuvakin. Public security log sharing site, <http://log-sharing.dreamhosters.com/>. Last Accessed: 2018-19-04.
- [3] Wikipedia Online. Idmef format, [https://en.wikipedia.org/wiki/Intrusion\\_Detection\\_Message\\_Exchange\\_Format](https://en.wikipedia.org/wiki/Intrusion_Detection_Message_Exchange_Format). Last Accessed: 2018-18-04.
- [4] Tom M. Mitchell. *Machine Learning*. McGraw Hill, 1997.
- [5] Wikipedia Online. Cross-validation (statistics), [https://en.wikipedia.org/wiki/Cross-validation\\_\(statistics\)](https://en.wikipedia.org/wiki/Cross-validation_(statistics)). Last Accessed: 2018-12-04.
- [6] Ren Pellissier. *Business Research Made Easy*. Juta Academic (September 5, 2008), ISBN-13: 978-0702177033.
- [7] Geoffrey K T Holmes, Andrew J Donkin, and Ian H. Witten. *Weka: A machine learning workbench*. 1996.
- [8] Camilla Wernersen Ellen Omland. Nsm: strste dataangrepet verden har sett), [https://www.nrk.no/norge/nsm\\_-\\_storste-dataangrepet-verden-har-sett-1.13515221](https://www.nrk.no/norge/nsm_-_storste-dataangrepet-verden-har-sett-1.13515221). (In norwegian) Last Accessed: 2018-11-06.
- [9] Kavita Iyer. 8 most amazing and daring hack attacks carried out by anonymous), <https://www.techworm.net/2016/01/8-amazing-daring-hack-attacks-carried-anonymous.html>. Last Accessed: 2018-11-05.
- [10] Kolluru Venkata Sreerama Murthy. *On growing better decision trees from data*, 1995. University of Maryland Institute for Advanced Computer Studies, Master Thesis.



## BIBLIOGRAPHY

---

- [11] S. Bhatt, P. K. Manadhata, and L. Zomlot. The operational role of security information and event management systems. *IEEE Security Privacy*, 12(5):35–41, Sept 2014.
- [12] Hewlett Packard Enterprise. Security operation center - building a successful soc. *Business White Paper*, 2015.
- [13] S. Getachew T. and D. Ejigu D. Layer based log analysis for enhancing security of enterprise datacenter. *International Journal of Computer Science and Information Security*, 14(7), 2016.
- [14] Ting-Fang Yen, Alina Oprea, Kaan Onarlioglu, Todd Leetham, William Robertson, Ari Juels, and Engin Kirda. Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks. In *Proceedings of the 29th Annual Computer Security Applications Conference, ACSAC '13*, pages 199–208, New York, NY, USA, 2013. ACM.
- [15] S. Bhatt, P. K. Manadhata, and L. Zomlot. The operational role of security information and event management systems. *IEEE Security Privacy*, 12(5):35–41, Sept 2014.
- [16] D. Curry H. Debar and B. Feinstein. The Intrusion Detection Message Exchange Format (IDMEF). RFC 4765, RFC Editor, March 2007.
- [17] S. Omid A. and S. Shiry Ghidary. Logs correlation: Current approaches, promising directions, and future policies. *Journal of Basic and Applied Scientific Research*, 2(5):4413–4422, 2012.
- [18] Ritu Bhargava N. Bhargava and M. Mathuria. Decision tree analysis on j48 algorithm for data mining. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(6):1114–1122, Jun 2013.
- [19] Kaan Ataman, George Kulick, and Thaddeus Sim. Teaching decision tree classification using microsoft excel. *INFORMS Transactions on Education*, 11(3):123–131, 2011.
- [20] K. Ryosuke O. Satoru and K. Kiyoshi. Minimizing false positives of a decision tree classifier for intrusion detection on the internet. *Journal of Network and Systems Management*, 16(4):399–419, 12 2008.
- [21] David Moore, Colleen Shannon, Douglas J. Brown, Geoffrey M. Voelker, and Stefan Savage. Inferring internet denial-of-service activity. *ACM Trans. Comput. Syst.*, 24(2):115–139, May 2006.
- [22] Tom Fawcett. An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861 – 874, 2006. ROC Analysis in Pattern Recognition.
- [23] Department of Computer Science Course: Computer Science 831 Knowledge Discovery in Databases University of Regina, USA. Confusion matrix, [http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion\\_matrix/confusion\\_matrix.html](http://www2.cs.uregina.ca/~dbd/cs831/notes/confusion_matrix/confusion_matrix.html). Last Accessed: 2018-19-05.

- [24] Ramandeep Kaur. An introduction to machine learning with decision trees, <https://dzone.com/articles/machine-learning-with-decision-trees>. Last Accessed: 2018-15-04.
- [25] Jean Carletta. Assessing agreement on classification tasks: The kappa statistic. *Comput. Linguist.*, 22(2):249–254, June 1996.
- [26] J. Richard Landis and Gary G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174, 1977.
- [27] Martin G. Forsey. Book review: Mike crang and ian cook, doing ethnographies. london: Sage, 2007. 244pp. isbn 9780761944461 (pbk) 24.99 isbn 9780761944454 (hb) isbn 9781848607477 e-book karen o reilly, key concepts in ethnography. london: Sage, 2009. 234pp. isbn 9781412928656 (pbk) 19.99 isbn 9781412928649 (hb) 73.00. *Qualitative Research*, 10(4):504–507, 2010.
- [28] Earl R. Babbie. *The Practice of Social Research*. Cengage Learning, 2015.
- [29] Kleber Stroeh, Edmundo Roberto Mauro Madeira, and Siome Klein Goldenstein. An approach to the correlation of security events based on machine learning techniques. *Journal of Internet Services and Applications*, 4(1):7, Mar 2013.
- [30] New Zeland University of Waikato. Weka 3: Data mining software in java, <https://www.cs.waikato.ac.nz/ml/weka/>. Last Accessed: 2018-14-04.
- [31] Department of Linguistics Uppsala University, Sweden and Machine Learning for Language Technology (2016) Lab 02: Decision Trees J48 Philology. [http://stp.lingfil.uu.se/~santinim/ml/2016/Lect\\_03/Lab02\\_DecisionTrees.pdf](http://stp.lingfil.uu.se/~santinim/ml/2016/Lect_03/Lab02_DecisionTrees.pdf). Last Accessed: 2018-14-05.
- [32] New Zeland University of Waikato. Attribute-relation file format (arff), <https://www.cs.waikato.ac.nz/ml/weka/arff.html>. Last Accessed: 2018-17-05.
- [33] University of Waikato. Data mining with weka, <https://www.futurelearn.com/courses/data-mining-with-weka>. Last Accessed: 2018-11-06.
- [34] Christine Kronberg. Analysis of the logfiles given in sotm34, <http://old.honeynet.org/scans/scan34/sols/2/proc.pdf>. Last Accessed: 2018-18-04.
- [35] Spain Daniel Rodriguez, University of Alcala. Making predictions on new data using weka, <http://www.cc.uah.es/drg/courses/datamining/ClassifyingNewDataWeka.pdf>. Last Accessed: 2018-19-05.