

Aalto University
School of Science
Degree Programme of Computer Science and Engineering

Alexander Zahariev

Graphical User Interface for Intrusion Detection in Telecommunications Networks

Master's Thesis

Espoo, March 28, 2011

Supervisor: Professor Tuomas Aura

Instructors: Ph.D. Kimmo Hätönen
M.Sc. Perttu Halonen

Aalto University School of Science Degree programme of Computer Science and Engineering		ABSTRACT OF THE MASTER'S THESIS	
Author: Alexander Zahariev			
Title: Graphical User Interface for Intrusion Detection in Telecommunications Networks			
Number of pages: 7+69	Date: 2011-03-28	Language: English	
Professorship: Data Communications Software		Code: T-110	
Supervisor: Professor Tuomas Aura			
Instructors: Ph.D. Kimmo Hätönen M.Sc. Perttu Halonen			
<p>Telecommunications networks increasingly depend on the Internet and computer networks. This exposes the telecommunications systems to intrusions, data theft, and service interruptions. Protecting against the intrusions is especially challenging because of the complex interdependencies inside the networks and between different networks. Moreover, a trend towards massive attacks against the network infrastructure is already evident.</p> <p>One solution to the security concerns is monitoring. Monitoring of large networks has become an active field both in practice and research. Through monitoring systems, malicious activities can be identified and analyzed, and knowledge is gained for better protecting the networks in the future. The work of network administrators can be aided by visualizing the monitoring data and results of analysis tools. The current security analysis and visualization tools have been designed for monitoring enterprise networks and do not adequately support the monitoring of telecommunication networks. One reason is that, in telecommunication networks, the volume of produced alarms and reports is far bigger than in enterprise networks and this increases the workload of network administrators. It is also necessary to understand the specific information, data sources and visualization methods suitable for telecommunications systems.</p> <p>This thesis focuses on solving the above problems in network-based intrusion detection systems (NIDS) that are based on anomaly detection. It presents a graphical user interface (GUI) concept for the analysis of anomalies in a telecommunications network environment. The goal of this GUI is to enable efficient exploration of suspicious events within the monitored network. In this concept, various visualization methods are used in order to enable a quick visual insight into communications patterns. Two use cases with synthetic data are used to demonstrate how the GUI facilitates the network administrator's work in judging the relevance of alerts and analyzing service usage within a network.</p>			
Keywords: network security monitoring, network-based intrusion detection, visualization for network security, large-scale network analysis			

ACKNOWLEDGMENTS

I would like to express my appreciations to all the people who helped me with this thesis. First of all I thank Professor Tuomas Aura for agreeing to supervise this thesis without any previous knowledge about me. I also thank both of my instructors Kimmo Hätönen and Perttu Halonen for their permanent support, flexible schedules and patience through all the phases of this project. I thank my colleagues Martin Peylo and Mika Wendelin who gave me the opportunity to be part of the world of Nokia Siemens Networks. Also worthy of mention is ICT SHOK Future Internet programme which greatly enabled this research. I also would like to express my appreciations to Gabriel Waller. Without his approval this project would not be possible. I would like to acknowledge Aliko, Liia and Konstantin for showing me their support in this thesis. Special thanks to Antti Niemelä who was with me all the way through this project, Boyan Tabakov for his invaluable technical advice, and Adam Herd for helping me express my thoughts in English.

Last but not least, I am deeply thankful to my parents and my entire family who has given me unconditional support from the very beginning of my entire Master's education.

Espoo, March 28th 2011

Alexander Zahariev

TABLE OF CONTENTS

1	INTRODUCTION	1
2	INTRUSION DETECTION AND SECURITY MONITORING	3
2.1	Intrusion Detection	3
2.1.1	Intrusion Detection Systems	3
2.1.2	Defense-In-Depth Strategy	5
2.2	Techniques of Intrusion Detection	6
2.2.1	Misuse Detection.....	6
2.2.2	Anomaly Detection.....	7
2.3	Importance of security anomaly visualization	12
2.3.1	Enterprise Networks vs. Telecommunication Environment	12
2.3.2	Main Problem with Security Visualization	13
2.4	Prior art	17
3	DATA AND VISUALIZATIONS IN SECURITY	20
3.1	Security Data.....	20
3.1.1	Data types	21
3.1.2	Data sources	21
3.1.3	Common Problems	25
3.2	Visualization methods	26
3.3	Information visualization process	30
4	COMPARISON OF MONITORING USER INTERFACES	33
4.1	Description of the tools.....	33
4.2	Comparison and analysis	40
5	SECURITY VISUALIZATION DEMONSTRATION	43
5.1	GUI concept	43
5.1.1	Components of UI	44
5.1.2	GUI development requirements	48
5.2	Use cases.....	49
6	DISCUSSION.....	57
7	CONCLUSIONS.....	61
	References	62

LIST OF FIGURES

Figure 2.1 General Network IDS Architecture. Figure is based on [10]	4
Figure 2.2 Defense-In-Depth Strategy	6
Figure 2.3 Architecture of a typical misuse detection-based system [2]	7
Figure 2.4 Architecture of a typical anomaly detection-based system [2].....	8
Figure 2.5 False positives, false negatives, true positives and true negatives.....	10
Figure 2.6 Architecture of a contemporary telecommunications network [45]	13
Figure 2.7 High-level overview of a telecommunications network with deployed IDSes	15
Figure 2.8 Screen shot of an alarm log file from Snort	16
Figure 2.9 Visual representation of the research problem	17
Figure 3.1 Data sources.....	21
Figure 3.2 Examples of legible (left) and illegible (right) pie charts	27
Figure 3.3 Examples of proper (left) and misleading (right) bar charts	27
Figure 3.4 Example of a line chart.....	28
Figure 3.5 Example of a link graph	28
Figure 3.6 Example of a scatter plot which visualizes a port scan attack	29
Figure 3.7 Example of a color usage in a report from IDS	29
Figure 3.8 The six steps of information visualization process [12].....	32
Figure 4.1 GUI of Sourcefire IPS [44].....	34
Figure 4.2 GUI of Cisco IDS event viewer [41].....	35
Figure 4.3 Some of the visualization methods used in Stonesoft's Management Center [42].....	36
Figure 4.4 Sample overview of configuration in Stonesoft's Management Center [42]	36
Figure 4.5 GUI of Splunk.....	37
Figure 4.6 GUI of Nessus' client part.....	38
Figure 4.7 GUI of Cacti	40
Figure 4.8 Screenshot of a report from Sourcefire system [44]	41
Figure 5.1 NIDS with a modular architecture	43
Figure 5.2 GUI concept.....	44
Figure 5.3 Buttonbar	45
Figure 5.4 Filterbar	45
Figure 5.5 Searchbar	45
Figure 5.6 Timeline.....	46
Figure 5.7 Events panel.....	47
Figure 5.8 Modules panel.....	47
Figure 5.9 Entries.....	48
Figure 5.10 Selection from the "Timeline" – step 1	49
Figure 5.11 Selection from the "Timeline" – step 2.....	50
Figure 5.12 Selection from the "Timeline" – step 3.....	50
Figure 5.13 Selection from the "Timeline" – step 4.....	50

Figure 5.14 Module bar – “Detail Descr” tab. Picture is based on [40]	51
Figure 5.15 Detailed description of a network element	52
Figure 5.16 Vulnerability report tab. Picture is based on [40]	52
Figure 5.17 Module bar – Detailed description tab. Picture is based on [40]	53
Figure 5.18 Module for various functionalities	54
Figure 5.19 Recent connections tab	55
Figure 5.20 Login statistics tab	55
Figure 6.1 Visualizing multiple events with same timestamp	59
Figure 6.2 Filtering events on the timeline	60

GLOSSARY

AD	Anomaly Detection
AS	Autonomous System
CEE	Common Event Expression
CPU	Central Processing Unit
DoS	Denial of Service
GUI	Graphical User Interface
HIDS	Host-based Intrusion Detection System
ID	Intrusion Detection
IDS	Intrusion Detection System
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPS	Intrusion Prevention System
LAN	Local Area Network
NE	Network Element
NIDS	Network-based Intrusion Detection System
NTP	Network Time Protocol
NVT	Network Vulnerability Tool
OS	Operating System
PNA	Passive Network Analysis
SNMP	Simple Network Management Protocol
SQL	Structured Query Language

1 INTRODUCTION

In the past decade, network management and security have become a key part in protecting the digital world. Network infrastructure is becoming more and more complex and its maintenance requires enormous effort from network administrators. Many working hours are spent on everyday incidents, which must be examined, categorized, decided how to act upon and preferably prevented in the future. Exhaustion becomes a major problem as network administrators have to crawl through endless logs, thousands of daily e-mails and alerts from various appliances. In addition, the increase of network attacks in terms of coverage, intensity and aggressiveness adds workload to network administrators. In particular, we are interested in attacks on corporate and government networks with the aim of stealing confidential information or blackmailing those whose business model relies on the availability of their services. An example of this is mobile operators which provide attackers with new attack surfaces. This additional attack surface emerged, because of recent convergence between traditional and mobile communications networks. To date, most of the network administrators rely on open-source tools, custom scripts and manual examination of logs.

One possible solution is to add a Network-based Intrusion Detection System (NIDS) as an additional layer of protection. NIDSes monitor network traffic and try to identify malicious activities from normal traffic flow. Such activities could be, e.g., Denial of Service (DoS) attacks, port scanning, or even attempts to gain illegal access to a machine. NIDSes in general perform detection and try to facilitate the work of network administrators. Although they reduce the amount of alerts and reports, NIDSes still tend to generate tremendous amounts of these [48]. In addition, there are no tailored solutions for telecommunications networks.

These facts point out the need of a software solution, which is able to cope successfully with, and efficiently analyze the network traffic, both in real-time and for audit purposes. Audit trail analysis can be especially useful for discovering compromised nodes in the network. Hence, a practical and useful graphical user interface for such a system is needed. This is not an easy task, because the proper information needs to be extracted and visualized in such a way that points out clearly what has caused the problem, so that the network administrators can perform appropriate actions in time.

This thesis is primarily focused on intrusion detection systems in telecommunications networks, and how to visualize anomalies and attacks in them. Nevertheless, in order for one to better understand the subject, other relevant topics will be briefly described and discussed as well. The final goal for this thesis is a graphical user interface concept for anomaly-based network intrusion detection. The operational environment for such a monitoring system is a mobile operator network.

The rest of this thesis is organized as follows. In Chapter 2, a high level introduction to intrusion detection will be provided. In addition, the application environment, as well as the main research problem is presented. The chapter ends with a literature overview of what has been done in this area so far.

Chapter 3 is dedicated to data and visualizations in security. In it, various data types and sources, as well as problems related to them are discussed. This chapter ends with the introduction of diverse visualization methods.

Chapter 4 provides presentation and comparison of several graphical user interfaces for various monitoring and reporting tools on the market. This chapter ends with a brief analysis and discussion about these tools.

Chapter 5 proposes a graphical user interface concept. This concept will face some of the problems discussed in Chapter 2. Chapter 5 is finalized with a description of two use-cases. Their goal is to show how the user interface can benefit the work of a system administrator.

Chapter 6 is dedicated to analysis and discussion of the achieved results.

Chapter 7 summarizes this thesis and conclusions will be offered to the reader as well.

2 INTRUSION DETECTION AND SECURITY MONITORING

This chapter gives a high level overview of the intrusion and security anomaly detection field. In order for the reader to better understand the current state of development, requirements, needs and problems in this area, one must have a basic knowledge about the following expressions and also be acquainted with the terminology related to it:

- intrusion detection
- misuse detection
- anomaly detection

In addition to the definitions and descriptions of the above mentioned terms, a description of the application environment and discussion about the research problem will be provided as well. Finally, the chapter ends with a review of prior art in the area of security visualization.

2.1 Intrusion Detection

The term *intrusion* is used to explain the process of gaining unauthorized access to a system. According to SANS Institute intrusion detection (ID) is “... the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource.” [1] In other words, intrusion detection is the field of research, which deals with intrusions and malicious activities in computers, networks and network elements.

An example of the above could be an intruder who uses software and hardware equipment in combination with computer networks to obtain customers’ credit card information and passwords from a bank server. An intrusion may also involve the spreading of malicious software to control infected computers. Intrusion detection is the method of finding, investigating and reporting unauthorized activities that may break the confidentiality, integrity or availability of data, services or systems.

2.1.1 Intrusion Detection Systems

Intrusion detection systems (IDS) can be software or hardware, which are deployed on network or a user computer, attempting to detect malicious activities. They primarily focus on identifying possible incidents, storing relevant information about these

incidents in a log, as well as providing reports to the security personnel. There are different kinds of IDSes including:

- intrusion prevention systems
- network-based intrusion detection systems
- host-based intrusion detection systems

Intrusion Prevention Systems

Intrusion prevention systems (IPS) are similar to IDSes, in that they try to detect intrusion and raise an alarm, but in addition, they also try to perform countermeasures to prevent intrusions. Such countermeasures include modifying firewall rules and closing established connection paths [1].

Network-based Intrusion Detection Systems

Network-based intrusion detection systems (NIDS) specifically monitor network traffic for intrusions and anomalous behavior. Usually NIDSes are software or hardware appliances which are deployed in specific places where they can monitor all traffic within a network. Packets are captured with no interferes to the connection and are then examined to find unusual traffic. If found, the system is set to raise an alarm, log the event and in some cases change a firewall rule or even reset the malevolent connection. By acting passively, NIDSes cannot easily be detected by an intruder. In addition they do not place significant overhead on the network [1]. However, they are not able to analyze encrypted or high-volume traffic [4, page 375]. Figure 2.1 depicts a general NIDS architecture.

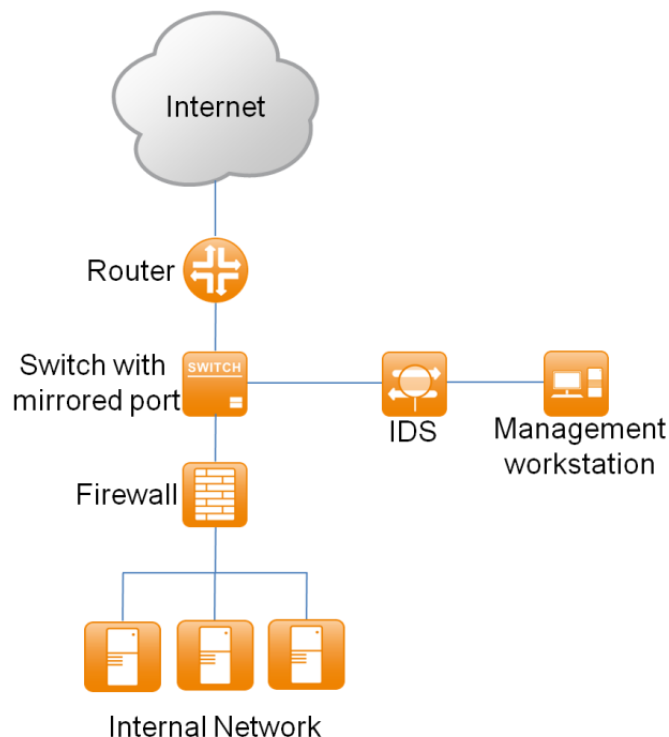


Figure 2.1 General Network IDS Architecture. Figure is based on [10]

Host-based Intrusion Detection Systems

Host-based intrusion detection systems (HIDS) are usually dedicated agents on a host machine. Their main purpose is to protect the state of that machine by monitoring different state indicators. These state indicators are typically: system vital files, currently running processes, network activity, application logs, currently logged in users, system calls, file system modifications (password files, binaries, databases), or any other user activity. Usually these systems are resource exhaustive and need to be installed on every host needing supervision [1].

2.1.2 Defense-In-Depth Strategy

Enterprise infrastructures, like banks or telecommunications networks, are valuable entities and from a financial point of view are highly profitable as well. It is obvious that people with enough motivation will attack these infrastructures. Such motivations may include fame, glory, entertainment and ruthless money making. Therefore, it is logical that the owners of such infrastructures will want to protect them. Hence, protecting the infrastructure becomes a vital issue for the economy of a company as it needs to deal with a large amount of information while maintaining integrity, availability and confidentiality.

In a computerized system which implements security policies, a well organized incident response system is crucial. Besides firewalls, anti-virus systems and other security solutions, IDSes play an important role because they [5]:

- examine the payload of packets and recognize malevolent traffic
- can raise an alarm in real-time
- provide information on scans that the network has been subjected to
- sense attacks initiated by malicious insiders from the internal network
- detect the presence of worms
- allow monitoring of the behavior of users within the network
- allow analysis of defensive measures to be taken into account for future attacks
- indicate the flaws in the system configurations or holes in the network
- show if another security solution did not work in a proper way
- are difficult to detect when they act passively

The broad functionalities described above led some security managers to believe that IDSes are able to cope with all security issues. Thinking that IDSes ensure total security of a system can be misleading and could create a false sense of security. Although they can provide decent security, an IDS acting alone will not be sufficient for various reasons, some of which are [5, 6]:

- they cannot replace other types of security solutions

- they may drop packets when the amount of traffic increases beyond their capabilities
- they are unable to analyze encrypted streams
- their efficiency is reduced as the number of rules increases
- they cannot replace security personnel
- they need constant rules updates in order to be efficient

IDSes should be used in addition to other security solutions in order to strengthen the protected system. It is important to realize that IDSes adds an extra layer of protection to a system. Figure 2.2 depicts the Defense-In-Depth strategy and shows how IDS contributes to it.

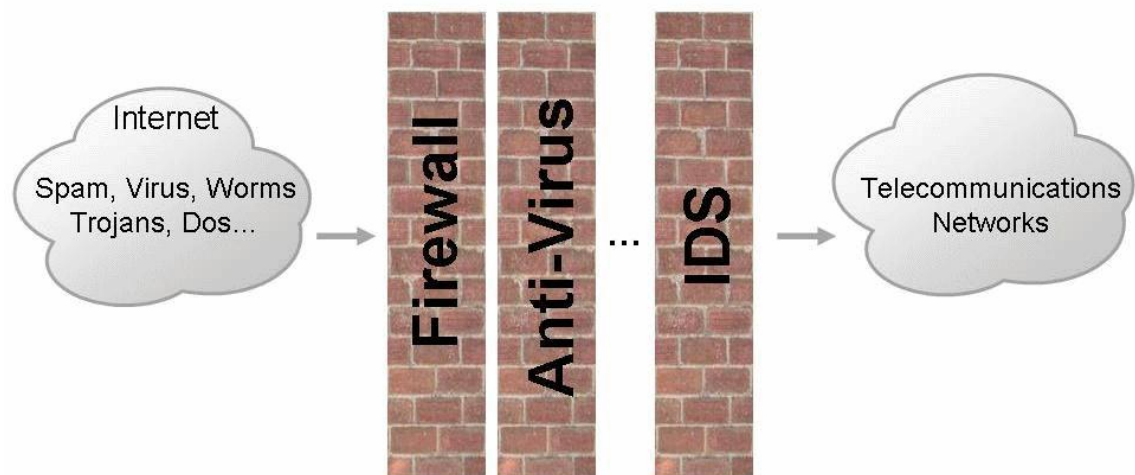


Figure 2.2 Defense-In-Depth Strategy

2.2 Techniques of Intrusion Detection

This section will present different methods used in ID. There is no perfect intrusion detection method. All of the available methods have advantages and disadvantages and are best employed depending on the case. Usually an IDS utilizes either one of the two major ID techniques [2, page 3], these are *misuse detection* and *anomaly detection*. Current trends show that a modern IDS takes advantage of a combination of these two major methods. In the following two sections, a high level overview of these techniques is discussed.

2.2.1 Misuse Detection

Misuse detection, also known as signature-based or pattern-based technique, is the most commonly used method. The concept behind it, is that there is a possibility to represent

attacks and intrusions or variations of them in the form of a pattern or a signature. These can refer to any combination of measurable or detectable characteristics in systems which are used to identify malicious activity. These characteristics can be the attempt to connect to a specific port, to start a process with a certain name, or to detect an incoming network packets which is known to trigger software vulnerability. Usually an IDS which employs *misuse detection* has a huge signature database and all events are checked against it. If a signature in the database matches the current activity, an alarm is raised to inform the security personnel. Such systems try to deal with *known* attack patterns.

The main concern in *misuse detection*-based systems is how to express an attack pattern in a signature that is able to cover all possible variations of this attack, and simultaneously write signatures which avoid matching legitimate activity. In addition, these signature databases need to be constantly updated, in order to cope with contemporary intrusion activities. Figure 2.3 depicts the architecture of a typical *misuse detection*-based system.

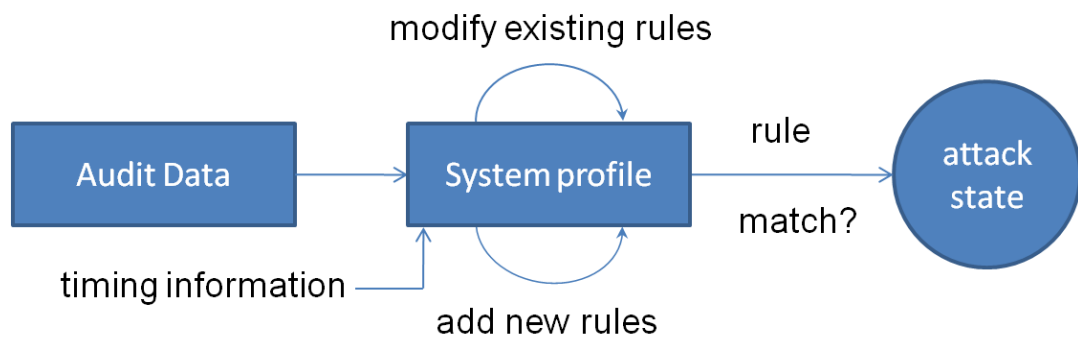


Figure 2.3 Architecture of a typical misuse detection-based system [2]

2.2.2 Anomaly Detection

Where the *misuse detection* technique, deals mainly with *known* attack types and malicious activities, *anomaly detection* deals with *known* and *unknown* attack types as well as intrusions. In IDSes which are using *anomaly detection* method, typically there are two major assumptions made. The first assumption is that network traffic has clearly distinguishable characteristics in “normal” condition. Based on these characteristics, a model or profile for normal activity or behavior can be created. The second assumption is that any deviation from this profile is not anticipated and may be a result of a malicious activity [2, 20].

Anomaly Detection as a Process

The concept behind *anomaly detection* is based on defining a profile for the normal operation of a system, and then monitoring for deviations from that profile. This is done

in two phases. The first phase is known as a calibration phase or training phase. In this phase the system is taught what is assumed to be “normal” data or activity and a model for this is created. In the second phase, once trained, the system monitors for deviations from this model. This means that once deviation from the model caused by an activity of a process is detected, an alarm or notification is raised. [2].

For a NIDS the first phase should consist of training the system with network traffic, clean of intrusions, in order to create a “normal” profile. In addition to this, the traffic should contain all possible allowed variations or fluctuations related to the environment which needs to be monitored [2]. Generating such traffic is not a simple task for the following four reasons. First, the scale of traffic simulated or generated in a laboratory environment is nowhere near to the size of the traffic in reality. Second, the variety of used protocols and their combined distribution cannot be simulated in the training traffic. Third, all traffic fluctuations or network element failures are difficult to simulate in laboratory environment. Fourth, if real traffic is used, there are no guarantees that it is clear of malicious activity [21].

When the model of normal network traffic is created and thresholds are defined, the second phase begins, and the NIDS monitors the network for deviations from that model. Once deviation from the profile occurs by crossing a threshold, further investigation and analysis is required in order to understand if this deviation is intrusive or not. Typically in practice, this analysis is done by a security expert. Figure 2.4 depicts the architecture of a typical anomaly detection-based system.

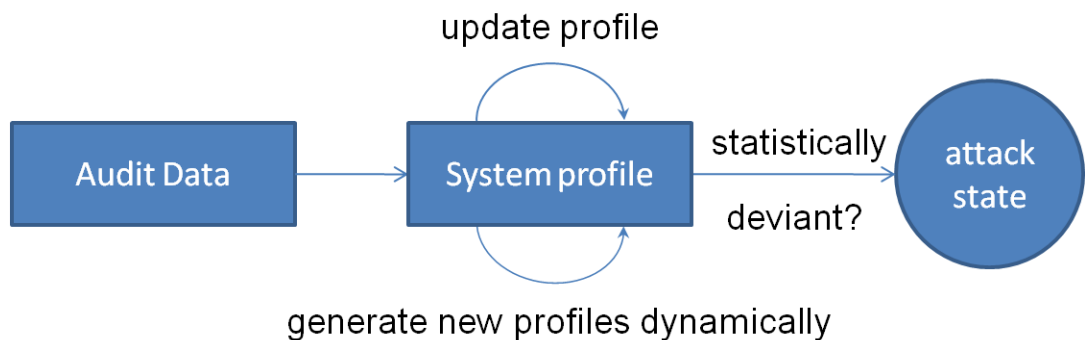


Figure 2.4 Architecture of a typical anomaly detection-based system [2]

Accuracy of classification

One of the main concerns with anomaly detection-based systems lies in its origins. It is important to point out that anomaly detection assumes that all intrusive activities are anomalous [2, page 3]. But this statement doesn't exclude that in reality there are also anomalies which are non-intrusive or violations that are not anomalous. For example, a non-intrusive anomaly in an NIDS can be an unexpected lack of network connectivity due to a malfunction in a network device.

Even though they are non-intrusive, these kinds of anomalies also generate alarms, known as false alarms or false positives. Once generated, an alarm needs to be analyzed by a human security expert. This is not a problem when there are only a few generated alarms, but in reality the amount of alarms is huge and it is difficult to track all of them [48]. In order to avoid generating false alarms, a proper selection of the threshold levels for the normal profile in the training phase is required, and again this is not a straightforward task. First, this is an expensive process because it requires time and constant updating of the profile metrics. Second, there are also possibilities to set the threshold levels in such a way, so that the NIDS will not raise an alarm if the intrusive activity is not anomalous. This is also known as a false negative. From an NIDS developer's point of view, these are the two most important and undesired instances. In order to evaluate the performance of an NIDS in general, it should generate as few as possible false positives and false negatives [2].

In addition to the false positives and false negatives, there are also two more occurrences. These are true positives and true negatives. True positives are occurrences of intrusions, which are correctly marked as anomalous. When observing the network traffic, an administrator wants legitimate traffic to not be classified as anomalous. These instances are also known as true negatives.

From a security researcher's point of view, true positives deserve attention because they tell about ongoing malicious activity. In addition to this, false positives and false negatives also deserve attention. This is because the security researcher has to investigate why they occurred, and possibly provide a solution for these false occurrences or at least reduce their number.

From a security administrator's point of view almost every occurrence is risky, except true negatives. True positives deserve the administrators' attention, in order for them to investigate and solve the problem. False positives will be brought to their attention for no reason, since these are legitimate activities being wrongly detected. Also this might lead to a scenario in which customers do not have access to their services. This is known as a denial of service (DoS). Finally, false negatives may lead him to a false sense of security.

All four types of occurrences are depicted in Figure 2.5. The vertical axis represents how activity is detected by an NIDS and the horizontal axis shows what the activity actually is.

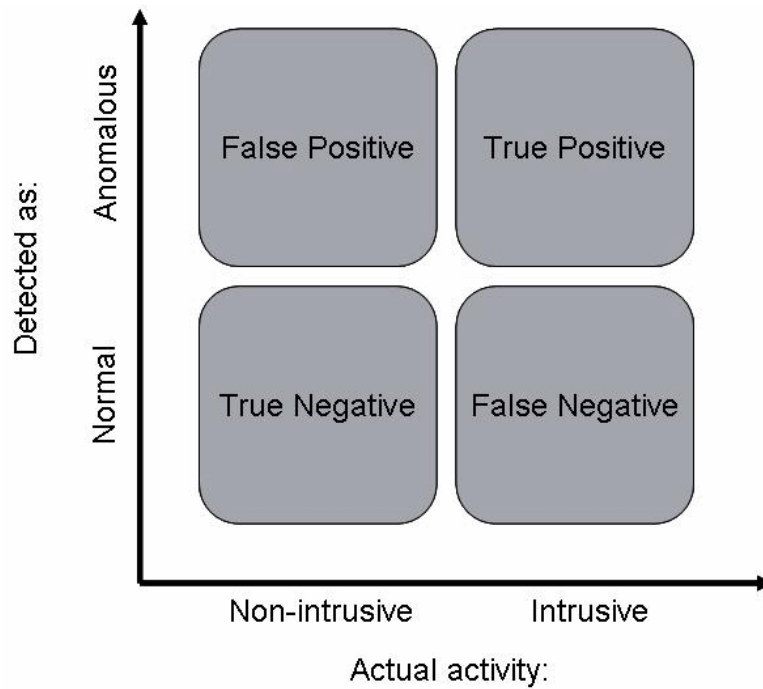


Figure 2.5 False positives, false negatives, true positives and true negatives

Anomaly Detection Methods

There are various methods for anomaly detection. Some of these are [2, 22, 23, 24, 25, 26]:

- statistical anomaly detection
- predictive pattern-based anomaly detection
- threshold-based anomaly detection
- machine learning-based anomaly detection
- payload-based anomaly detection
- protocol-based anomaly detection
- graph-based anomaly detection

Statistical anomaly detection method

In the statistical method anomalies are identified based on statistics. This means that the system's measurable characteristics are monitored for a period by statistical means, in order to understand what their typical values are. A model is then created based on the results. Some of these measurable characteristics include: network traffic characteristics, values inside packet header fields, resource usage (CPU, memory, I/O), values inside protocol message fields, a list of processes running on a computer, and a list of ports typically used for communication. After a model is created, the current condition of the system is compared to that model and any deviations from it are considered as anomalous. Once deviation occurs, its severity is evaluated and graded.

The higher the severity, the higher the grade [22]. For example, if a user in a company normally sends around fifty to seventy e-mails per day, going over this number by one or two is anomalous but the severity is low. On the other hand, if the same user goes over the typical number of e-mails which he sends per day by one hundred, then the severity of this anomaly is going to be higher, and its grade too. Of course, every case is different and requires its own customization.

Predictive pattern-based anomaly detection method

In the predictive pattern-based method, occurrences are predicted based on events that have already occurred. There are two main advantages of using this method. First, systems based on this method are highly adaptive to changes, and second, once properly trained, the system can detect and report anomalies within seconds [2].

Threshold-based anomaly detection method

The concept behind the threshold-based method is that predefined thresholds are set for data deviation monitoring, and once this threshold is crossed, the occurrence is marked as an anomaly [2]. To some extent the threshold-based method is a combination of the predictive and statistical methods. The threshold as such is based on previous and statistical observations. As an example, a network administrator knows how much the average bandwidth in a network element (NE) is. Thus, he can set the threshold for this average value, and once it is crossed, an alarm will inform him of this deviation. To conclude the above, the crucial element in deploying an effective anomaly-based IDS using the threshold method and creating as few as possible false alarms, is the proper selection of thresholds levels.

Machine learning-based anomaly detection method

In this method, anomaly detection models are created based again on past behavior, as in the previous three methods. But this time different machine-learning algorithms are used. As an example, the learning algorithm analyses previously captured data sets, which contain network traffic, and after the analysis, a model of the network behavior is created. After this calibration phase, the monitoring sensor looks for deviations from that model. One major advantage in this method is that it can automatically adapt to changes in the network traffic. For example, if an operating system (OS) security update is distributed to all machines within a local area network (LAN) at same time, it will generate previously unknown traffic, but an NIDS which uses a machine learning-based anomaly detection engine, will sense that this traffic is not anomalous [23].

Payload-based anomaly detection method

In a payload-based method, anomaly detection models are based on the payload of a network packet which is targeting to host's Internet Protocol (IP) address and specific port. Additionally, deviation from the model is also calculated based on the payload size. Once the model is created, all incoming traffic and its payload length (designated

to a specific port of the target host) is analyzed and matched against the model's average payload size. If there is a significant difference between them, an alarm is raised [24].

Protocol-based anomaly detection method

In this method, anomalies are detected if the protocol in an incoming communication deviates from the protocol's standard specification. Since all connection oriented protocols have state, most of the protocol-based anomaly detectors are built as state-machines. Therefore, these detectors monitor protocols' transitions from one state to another, and if there is unanticipated transition, an alarm is raised [25].

Graph-based anomaly detection method

In this anomaly detection method, graphs of hosts and network activity are created. These graphs visualize how an activity is spreading in a network environment. As an example, if the graph of an activity becomes huge and tree-like, this activity is considered as anomalous [26].

All of the above methods have their advantages and disadvantages depending on the specifics of the target requiring monitoring. In modern IDSes, a combination of various methods might be employed. But in order to choose the right methods, a prior analysis of the environment, its features and desired goals, must be undertaken.

2.3 Importance of security anomaly visualization

This section is dedicated to establishing the research scope of this thesis. In order for the reader to better understand the research problem, one must have a basic idea of the application environment as well as the issues related to it. In the case of this research, the application environment is telecommunications networks. One might assume that this environment is similar to enterprise networks, however this is not quite true. Though they have some similarities, they also differ in many ways.

2.3.1 Enterprise Networks vs. Telecommunication Environment

Let us imagine a typical enterprise network with its entire sub networks interconnected together via switches and routers, as a fort. The idea of the fort is to keep the attackers out. In most cases, there is one entrance point, several guard towers to monitor the wall of the fort, and a firm control of what goes through the main gate. This is the so called *perimeter defense*. In addition, there may be more security layers inside the fort premises, for example patrolling guards. Similarly in enterprise networks a perimeter defense regarding the Internet is established, which guards, regulates and secures the perimeter of the network, like a fort. Most importantly, only those who are authorized

and approved are allowed inside the fort. Those already inside the fort have specific permissions and places to be, and they can be monitored, controlled and regulated. Hence, most of the attacks can be anticipated, detected and deterred directly at the fort's wall.

Now let us try to picture the telecommunications network as a nuclear missile silo. A nuclear missile silo should withstand any attack, no matter where it originates from. The same applies to a mobile operator's network. It can be attacked from any direction, any network element, any protocol etc. Most importantly, it should survive and endure direct attacks to the core components of its business. In telecommunications, the infrastructure size is typically huge, and in addition to the elements which are similar to those in an enterprise network, there are also various Radio Access Networks (RAN). These RANs create additional attack surface and provide the intruders with new ways to compromise the infrastructure. Telecommunications networks as such, do not control how RANs are used. There are various points of contact which are not under the control of a single entity. This requires security policies with wide ranges. In some cases, packets flow without restraints, thus the "deny everything" approach is not applicable here [7, 8]. Figure 2.6 shows the general architecture of a contemporary telecommunications network.

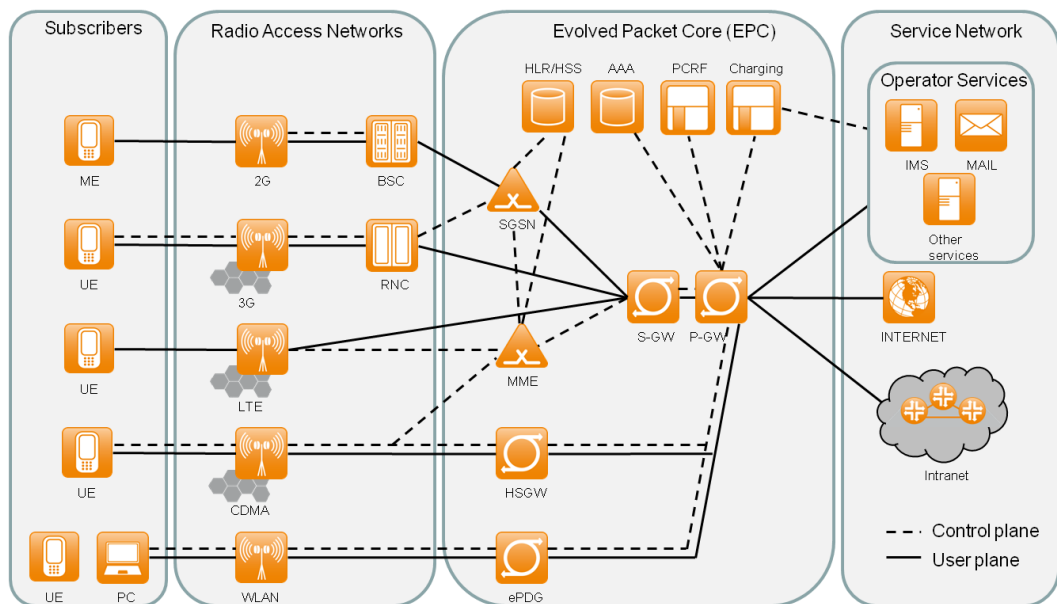


Figure 2.6 Architecture of a contemporary telecommunications network [45]

2.3.2 Main Problem with Security Visualization

Two questions will arise immediately from looking at Figure 2.6. The first is "What to Monitor?" And the second "How to visualize it?" Discussion about these questions is provided in the following two sections.

What to monitor?

Users of enterprise networks are employees, thus the employer has control over his employees' communications. Furthermore, the employer owns the network. In telecommunications networks the majority of users are subscribers, thus their communications are protected by legislation. Consequently, in enterprise networks, security administrators have more rights to observe and inspect the ingoing and outgoing traffic on a packet level, than in telecommunications networks.

Another issue for security administrators in enterprise networks, is firewall configuration. An administrator can set firewall, for example in a way that in the beginning he closes all available ports and then opens only the ones which are needed in order to ensure communication between parties. In telecommunications networks is the other way around. There are domains in which security administrator should leave all available communications towards subscribers open, and will close only the ones which are previously known to be used for malicious activities (See Section 2.3.1). This is done in order to ensure the interoperability with other mobile operators and the resources which are outside the premises of the network.

Although the administrators in mobile operator networks have to monitor much more traffic than their colleagues in enterprise networks, their task is easier in one sense, because they are not allowed to look at the payload. But the hard part is that in mobile operator networks, there are more connections, more complexities and more alarms. Thus, the administrators in mobile operator networks have to process much more information than the security officers in enterprise networks.

In addition to this, given that nowadays mobile telecommunications networks are converging with fixed networks, telecommunications customers are no longer secure behind "walled gardens." This means that the attackers no longer need to be physically close to their victims. Intruders can attack their victims remotely, for example straight from their homes. One can conclude from the above that security solutions, such as IDSeS, for telecommunications networks are immature in comparison with those of enterprise networks.

As it already has been discussed in Section 2.1.2, telecommunications networks are valuable and profitable. It is obvious that somebody with enough motivation will attack this infrastructure, for various reasons, such as fame and glory. Yet at the same time, it is obvious that the owner will want to defend their profitable infrastructure. But another issue is that of where monitoring devices should be placed in order to capture the whole ingoing and outgoing traffic. In Figure 2.7, a proposition for key points where IDSeS should be deployed is presented.

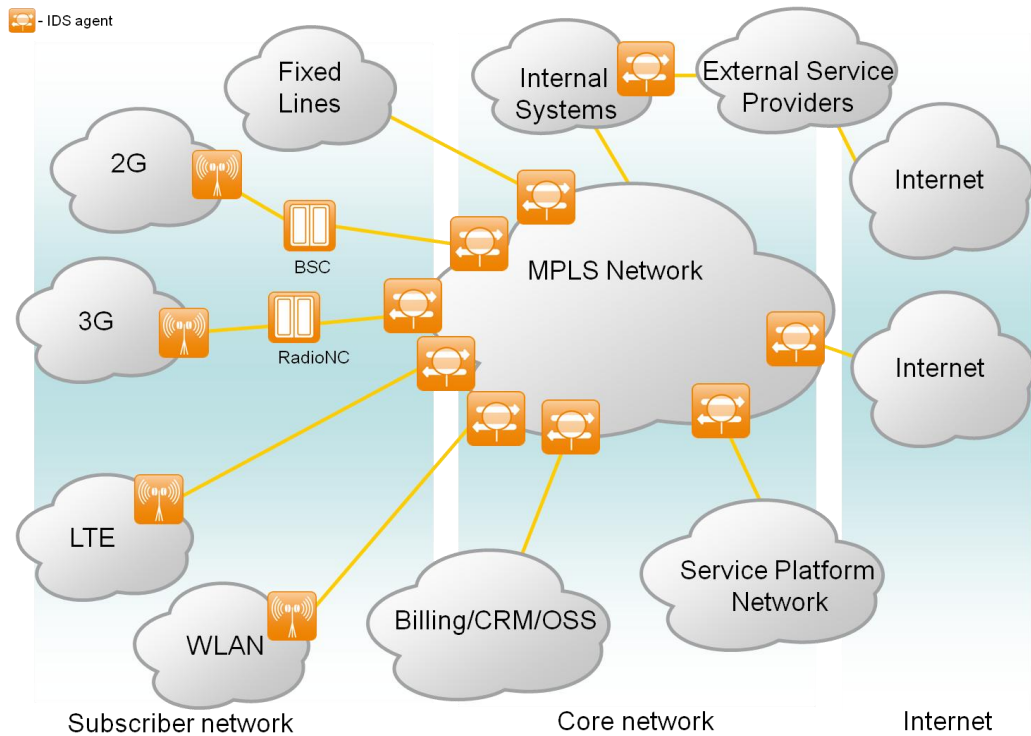


Figure 2.7 High-level overview of a telecommunications network with deployed IDSes

How to visualize it?

Here we come to our second question, how to visualize the malicious activities? More particularly, how to visualize anomalies regardless of whether they are malicious or not, in the environment of telecommunications networks.

The constantly increasing amount of network attacks and ever growing complexity of networks, require better network traffic monitoring tools for security administrators. These tools should be able to determine and assess attacks and anomalies as quickly as possible. As we can see in Figure 2.7, network can be monitored from various places. In addition, there is too much data to be handled manually [48]. One possible solution to this is to employ data visualization.

Visualization is usually not associated with network security, but it is a good way to summarize and understand a large amount of network data. Different visualization methods can be employed to form an overall image, in order for the security management to understand the current situation quickly and easily. Benefits from this are that it saves time and money. Money is saved, by requiring fewer security administrators to browse through the logs manually.

To narrow down the problem, let us take the following example. From a security network administrator's point of view, there is an enormous amount of data generated even by small networks and sub-networks. It is almost impossible to check all reports, and log files, perform monitoring in real time, set configuration files, and maintain a

holistic situational awareness all at the same time. Even if there is an appropriate distribution of well trained security staff dedicated to each one of these purposes, this is still a demanding and time consuming task. Going deeper into the problem, in addition to the normal logs and reports, IDSeS tend to generate massive amounts of false alarms (See Section 2.2.2), and all of them deserve attention from the security team. In Figure 2.8, we can see a practical example from an alarm log file.

```

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:13.418247 192.168.1.205:50572 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:53753 IpLen:20 DgmLen:60 DF
*****S* Seq: 0x981194EF Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294919446 0 NOP WS: 6

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:15.263311 192.168.1.205:50573 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:52677 IpLen:20 DgmLen:60 DF
*****S* Seq: 0x99908B21 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294919908 0 NOP WS: 6

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:16.811569 192.168.1.205:50574 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:38291 IpLen:20 DgmLen:60 DF
*****S* Seq: 0x9AA5DDF4 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294920295 0 NOP WS: 6

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:20.190464 192.168.1.205:50575 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:23226 IpLen:20 DgmLen:60 DF
*****S* Seq: 0x9E9F1109 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294921139 0 NOP WS: 6

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:24.923343 192.168.1.205:50576 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:22996 IpLen:20 DgmLen:60 DF
*****S* Seq: 0xA2680986 Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294922322 0 NOP WS: 6

[**] Snort Alert! [**]
[Priority: 0]
11/25-22:00:29.738580 192.168.1.205:50577 -> 192.168.1.102:22
TCP TTL:64 TOS:0x0 ID:63656 IpLen:20 DgmLen:60 DF
*****S* Seq: 0xA704ADAF Ack: 0x0 Win: 0x16D0 TcpLen: 40
TCP Options (5) => MSS: 1460 SackOK TS: 4294923527 0 NOP WS: 6

```



Figure 2.8 Screen shot of an alarm log file from Snort

The example in Figure 2.8 is a screen shot from the alarm report of one of the most popular IDSeS on the market—Snort. Although it is a very detailed report and it provides enough information in order to investigate and solve the problem, it is a difficult and slow process to browse through it manually. The conclusion from this is that this information needs to be shown in a different way. In a way that obviously highlights the problem, and it's causes. Following the scope of this thesis, an innovative Graphical User Interface (GUI) for an anomaly-based NIDS operating in the mobile telecommunications environment is needed (Figure 2.9).

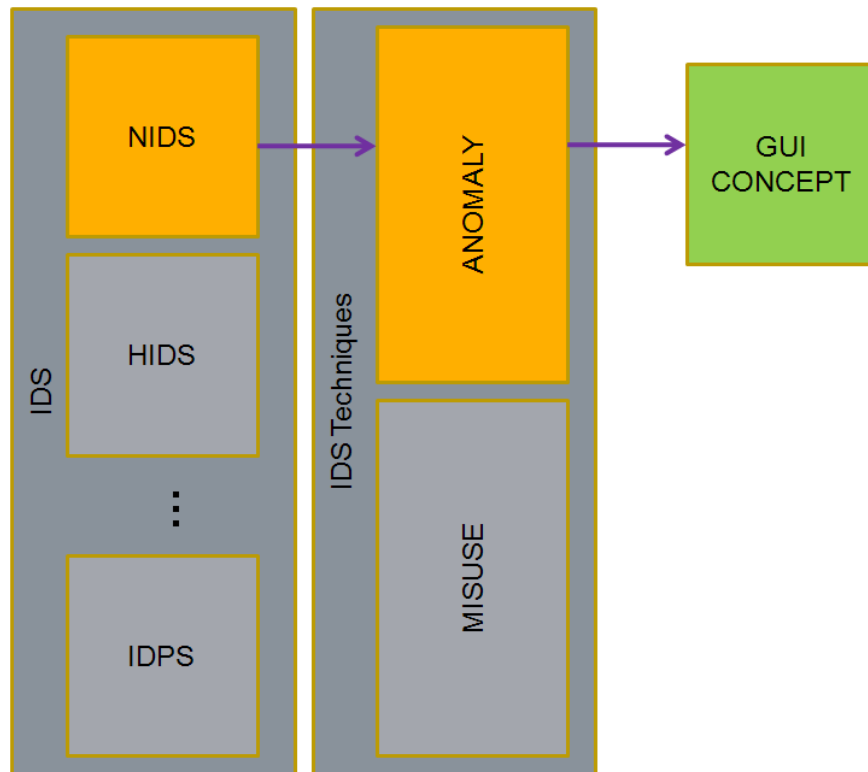


Figure 2.9 Visual representation of the research problem

2.4 Prior art

In this section literature overview and prior art of what has been done so far in security visualization is going to be provided to the reader. Although lot of research has been done in intrusion detection systems for enterprise networks, the situation is not the same with telecommunication networks. This is because until recently, the mobile subscribers were secure inside “walled gardens”, and almost nothing could happen to them caused by malicious outsiders. But now, when IP has taken over the traditional way of connecting in telecommunication environment, the need for different security mechanisms has increased. Nevertheless, some attempts have been done in this area as well.

One of the pioneers in data visualization area is Edward Tufte. In his book, “The Visual Display of Quantitative Information”, he proves that data visualization can do much more than just substitute small statistical tables. The book explains how to display data for precise, effective and quick analysis [9].

Another work in the area of information visualization is the book “Information Visualization – Perception for Design” by Colin Ware. In this book, the key principles for the creation of information visualizations are presented. These include cognitive principles and a guide to the human visual perception. The book justifies that human

brain tends to perceive better and faster combination of colors, images and shapes than digits and strings [11].

One of the most comprehensive overview of network visualization is provided in [15] by Martin Dodge and Rob Kitchin. In the book, the authors summarize hundreds of different network visualization projects and innovative visualization techniques, which help to better understand the cyber world.

In [17] visual symbols are proposed by Hilary Hosmer, in order to facilitate report of security incident to the security personnel and management. The author discusses that visual attack scenarios help defenders to see the ambiguities, vulnerabilities and imprecision in a system, thus speeding up the risk analysis, selection of the security solutions and information security trainings of the personnel.

Going deeper in the area of information and network visualizations, we move towards security visualization. One of the most active persons in this area is Raffael Marty, author of the book: “Applied Security Visualization” [12]. In the book, he introduces an information visualization process that describes, how network data from different sources, should be transformed into a visual representation. This is done in order to make it easier and faster for the network administrators to detect anomalies, vulnerabilities and other security incidents. This information visualization process consists of six steps. These steps are: define the problem, assess available data, process information, visual transformation, view transformation, interpret and decide.

In [16] a prototype design tool from the Harris Corporation is presented by Ronda Henning and Kevin Fox. The name of the tool is Network Vulnerability Tool (NVT). It visually depicts a network topology and generates a vulnerability evaluation window with results from proactive scans and vulnerability database.

In [18] Philip Varner and John Knight prove that visualization should be the next focus of intrusion detection systems. This is because with graphical monitoring it is easier to react to undesirable events, thus increasing survivability of a system.

Bearavolu et al. [13] present in their study a novel visualization tool that provides holistic multi-level security monitoring of an entire IP address space on one screen. Different tests conducted by them show that by visualizing simultaneously traffic activity at different levels, they are able to discover new relationships and patterns in this traffic, that otherwise would have been unnoticed by the total volume of unprocessed information and difficulty of gathering and analyzing this data.

In [19] Soon Tee Teoh et al. demonstrate how with using visualization methods it is easier to detect and analyze anomalies as well as to characterize Internet routing

behavior and insecurity. In addition to this, authors are proving that, with non-visual security methods it is more difficult to correlate events and to distinguish between different types of autonomous systems (AS) route changes.

In [14] Fabian Fisher et al. present a system called NFlowVis which is able to analyze intrusion detection and flow data. Their user interface follows a drill-down technique, in which administrator is guided from the abstract overview of the overall network activity to aggregated views of IDS data.

3 DATA AND VISUALIZATIONS IN SECURITY

In this chapter various data types and sources, as well as some common problems related to this area are discussed. In an addition, an introduction to some visualization methods is also offered.

3.1 Security Data

Before exploring the topic of security visualization, the reader must know and understand the data which needs to be visualized. He needs to know the answer of the questions “What is security data?” and “Where can data be extracted from?” There is no separate class, which is labeled *security data*. Every record, which is useful for finding and solving security issues, regardless of its source (network elements, transaction records, OS log files etc.), can be considered *security data*.

Here are the definitions of some of the terms which are going to be used in the following sections of this thesis. Definitions are extracted from Common Event Expression (CEE) White Paper published by The Mitre Corporation [27].

Event

An event is an observable situation or modification within an environment that occurs over a period of time. An event may be a specific state or a state change of a system [12, 27].

Each event can be described or recorded. In a typical situation an individual record is frequently called log entry.

Log Entry

A log entry is a single record involving details from one or more events. A log entry is sometimes referred to as an event log, event record, alert, alarm, log message, log record, or audit record [12, 27].

Log

A log is the collection of one or more log entries typically written to a local log file, a database, or sent across the network to a server. A log may also be referred to as a log file, an audit log, or an audit trail [12, 27].

3.1.1 Data types

Security data can be divided in two main categories. These are *time-series* data and *static data* or configuration data [12, Chapter 2].

Time series data

Any data which can be attributed to a specific moment in time is called *time-series data*. The perfect example of *time-series data* is a log record. In a typical case every log record has a timestamp coupled with it, which marks the time when an activity has been logged. The timestamp does not identify the exact time when the event has occurred; instead the timestamp identifies when the log record was generated and stored [12, Chapter 2].

Static data

The other major type of data is *static data*. This means that any data which is not associated with a specific moment in time is *static data*. Files and documents are typical examples of such data. In addition, any information about the users or machines in an environment is considered also as *static data*. In some specific cases, *static data* can also be observed as *time-series data*. Such a case could be, a system administrator checking when the last modification was made to configuration file from a data base server [12, Chapter 2].

3.1.2 Data sources

There are various sources from which security data can be extracted. The following descriptions of *data sources* are ordered, according to the Network Stack as depicted in Figure 3.1 [3, Chapter 2], starting with Packet Capturers and moving towards Applications.

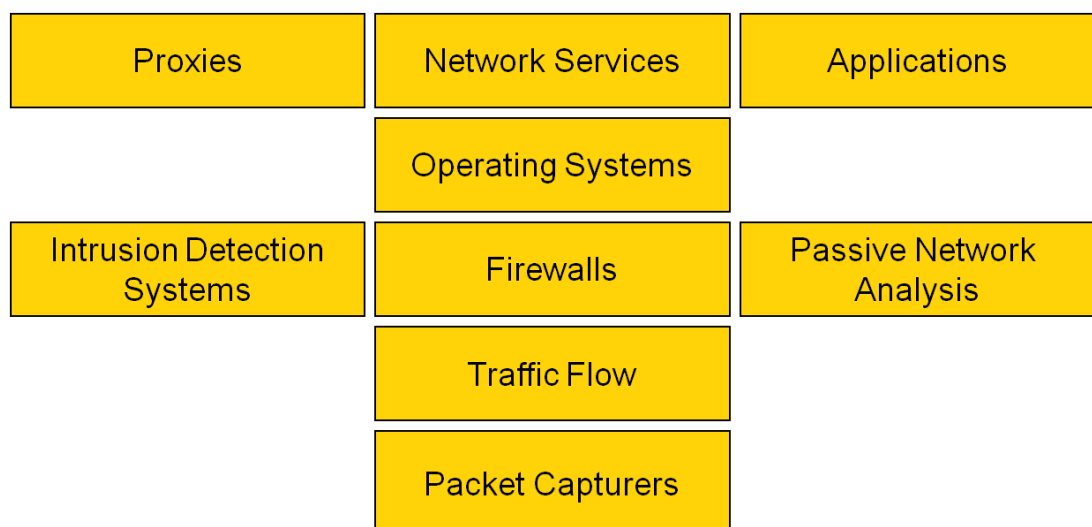


Figure 3.1 Data sources

Packet captures

The first possible sources of information according to the Network Stack are network *packet capture*s. Network packets are physically received by network interface. From this point, packets are passed to the operating system, and the network driver is responsible for extracting the information encoded in the packets. After this, the packets are analyzed layer by layer, and passed up to the corresponding network protocol in the stack. The moment in which a packet is recorded in a log is immediately after a network packet is passed from the network element to the operating system.

The biggest advantage in recording the packets at this point, is that all the data, including the payload, is present for recording. This means that no higher level protocol from the network stack is used to filter or remove information from the packet. Contrary to this, the biggest disadvantage, in recording packets at this point, is that no higher level protocol intelligence from the network stack can be applied. One can only guess at what information the packet is carrying, based on previous experience, but it is unknown how the packet is going to be interpreted by the application to which is designated.

There are various applications for capturing packets available on the Internet, but undoubtedly, the most popular one is Wireshark [28, 29, 30]. It works by listening to a network interface, taking the raw traffic, then displaying the captured traffic and analyzing the individual protocol headers [28].

Traffic flows

Moving up the network stack, one level above *packet capture*s is *traffic flows*. It is important to mention at this point, that by moving up the network stack, some information is dropped away which is available at the lower levels. *Traffic flows* are captured on routers and switches and they operate in the transport layer of the network stack. This means that the information from the layers above transport layer, such as the application layer, is still not available. Almost all of the major network equipment vendors are supporting their own protocol or format to record the *traffic flows*. For example, Cisco has NetFlow [31], Internet Engineering Task Force (IETF) standardizes IPFIX [32] and Juniper supports cFlowd [33]. Nevertheless, all of these formats are more or less similar, and the only major difference between them is the transport methods which use, to collect the flows from the network element.

In addition, it is also possible to collect traffic flows on a host, instead of from a router. One possible solution is ARGUS. ARGUS works by collecting the captured packets and then translating them into traffic flow [34].

Firewall logs

Typically firewalls operate in the transport layer of the network stack. Some newer generation firewalls operate in the application layer and employ various techniques like application inspection, deep packet inspection and protocol analysis. In this thesis traditional functionality of firewalls is discussed.

Log entries generated by firewalls are very similar to the ones generated by traffic flows. The major difference between them is the addition of information in the firewall log, namely, whether a packet was passed through or blocked.

Intrusion detection and prevention logs

Nowadays it is almost impossible to implement a secure IT infrastructure without employing IDSes. At this point in the discussion, the reader already knows how a typical IDS works (See Chapter 2), and he is aware that IDS does not log every activity, but only the ones which are violating a specific rule. Thus, use cases in which an IDS log can be used are reduced. This means that in most cases, logs from other resources are needed in addition to the logs from IDSes, in order to see the whole picture of how an attack was conducted and assess its impact to the system.

Passive network analysis

Operating on the same level of the network stack as firewalls and IDSes, are passive network analysis (PNA) tools. These applications passively listen on the network and capture traffic. Their purpose is to extract various meta-information about the communicating hosts. Using different heuristic methods they are able to determine the hosts' operating systems, as well as various services and applications used by them. One of the most popular open source tools for passive analysis is called p0f [35].

Operating system logs

In some cases, it is more useful to log and analyze information about the end systems rather than network-based information. One of the biggest advantages, is that operating system log files consists of information recorded by the kernel itself. Two kinds of data can be categorized in a typical operating system environment. These are *near real-time information* and *state information*.

Near real-time operating system information records status changes which occurred to the system. Typical information enclosed in this kind of data consists of: operating system logins, file auditing (file creation, changes to the file, file deletion), system restarts or shutdowns, actions executed as a different user, as well as resource errors, like hard disk failure. Of course each case is different and under specific circumstances, an operating system can log many more other activities, but this heavily depends on the system's configuration and the purpose for which it is used [12, Chapter 2].

Additional information which can be monitored is the state of an operating system. Typically every operating system is shipped with its own tools to monitor and extract such information. Examples of this kind of information are: network status (including all related network information, like list of available network interfaces), input/output statistics for hard drives, CPU and memory, list of the constantly running processes as well as their ID number, memory utilization and initiator of the process. In practice security administrators utilize such information with the usage of the embedded tools shipped with the OS. Collection for Windows OS tools is available in [36]. Monitoring of state information is also possible remotely and each OS has its own tools and methods to retrieve this information.

Application log files

Traditionally, the area of log analysis is reserved for the lower levels of the network stack. Nonetheless, in some cases only logs from the application layer are suitable to provide enough information in order to solve a security issue. This is because information about the username, web address or database query are stored only on a data source from the application layer. There are three most typical data sources from which one can extract application layer log info [12, Chapter 2]. These are logs from:

- web proxies
- mail server logs
- databases

Web proxies are interesting because in addition to logging just the network connection, they also log the exact web requests associated with this particular connection. Another benefit is that if the proxy server is set in a way that will allow only legitimate users after authentication, it will also log the corresponding user name and password for the person who tries to access this web resource.

Mail server logs deserve attention as well. This is because after analyzing the relationship between the corresponding parties in a mail exchange, a security officer can derive the social circle of malicious insiders and use the information for information leak protection.

And third, there are also benefits from analyzing database server log files. This is because by default, a database log contains info about startups, shutdowns, errors and, if the authentication is turned on, logins as well. This type of information is especially important for diagnostic purposes.

3.1.3 Common Problems

During the quest for extracting, visualizing and analyzing data, three major problems are inevitable. These are

- incomplete information
- parsing information
- non-synchronized time

Incomplete data

One of the challenges of working with data, is incomplete information. Some data is always missing which would be needed to put the complete story together when investigating a pre-recorded set of log files. Incomplete data can be either a missing log record, or an entire missing log file. Solution for this problem is proper log management. System administrators should make sure that all log files are consistently collected. In addition, they need to be prepared with possible use-cases for investigating problems. This is because use-cases dictate how to deploy proper logging architecture, as well as how, and from where data should be collected and aggregated.

Additionally, the case maybe that a security administrator wants to log a specific feature, but there is no way to log it, due to the fact that there is no such embedded function to log this specific information in the product. The CEE standard developers are currently working at preparing a set of logging recommendations that vendors should implement in their appliances. However, published standard exists at the time of writing this thesis.

Non-synchronized time

Another common problem is with recording the timestamp in the log files. In case collection of the data is from sources in different time zones, the time zone should be indicated in the log, otherwise there will be confusion in the order of the occurred events. In addition to this, the granularity of recording and collecting of the occurrences should be adequate to the event. For example, financial transactions are very time sensitive and even small differences of a milliseconds can be crucial in analyzing them later on.

Besides the problem with the timestamp in log files, the clock synchronization of the nodes itself also poses a problem. It is possible that the internal clocks of several distributed machines differ. This is because, even when these clocks are initially set accurately, real clocks will differ after some amount of time due to clock drift. Clock drift is caused by clocks counting time at somewhat different rates. In a distributed environment this is especially important, because distributed nodes need to realize the same global time. One possible solution is to employ usage of network time protocol (NTP).

Parsing information

One of the most common problems working with log files is translating data from one format to another. This process is also known as *parsing*, and tools which automate this process are known as parsers. The definition of a parser provided in [12] is:

“A computer program that breaks down text into organized strings of characters for further analysis.”

This means that a text file, a log record for example, is selected. Individual parts in it are identified, and then transformed into another data format. This is especially important when working with visualization tools. First, because almost every tool has its own data format, which is taken as an input, in order to generate visualizations. Second, usually tools do not contain built-in parsers which can be used to directly read or extract information from a log file. In addition, as yet is no standard form in which events should be recorded, and there is no one parser that could be used to analyze all of the various existing logging formats.

3.2 Visualization methods

Once it is clear what data needs to be analyzed and what information is encoded in it, a decision how to visually represent this data needs to be made. However, this is not a straightforward process. This is because various visualization methods exist, and each one has its own advantages and disadvantages [12, Chapter 3]. Some of these visualization methods are: pie charts, bar charts, line charts, link graphs, scatter plots and the usage of colors in graphs.

Pie charts

One of the most popular and widespread methods to visually represent data is through pie charts. Pie charts are circular charts, divided into sectors, and they typically present the percentage of a whole. They are good for visualizing static data. One of their biggest disadvantages is that they are able to visualize only a small amount of different values. This is because if too many values are used, the chart becomes illegible. If many different values need to be represented, another visualizing method should be considered instead. In Figure 3.2, examples of legible and illegible pie charts are presented.

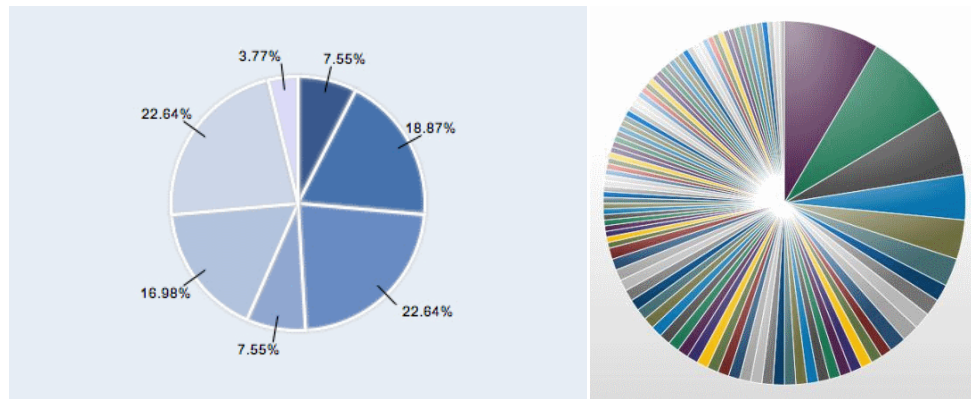


Figure 3.2 Examples of legible (left) and illegible (right) pie charts

Bar charts

This is a type of chart with rectangular bars, whose length is proportional to the values of the represented data. In a typical case the bars are plotted vertically, but horizontal plotting is also possible. Bar charts are primarily used to represent static data. It is preferable, whenever possible, that the y-axis of a bar chart start from zero. This is in order to make the comparison between individual graphs easier. In addition, if the bars do not start from zero, it is misleading and may cause confusion. An example of how the same data can be represented in different ways using bar charts is shown in Figure 3.3.

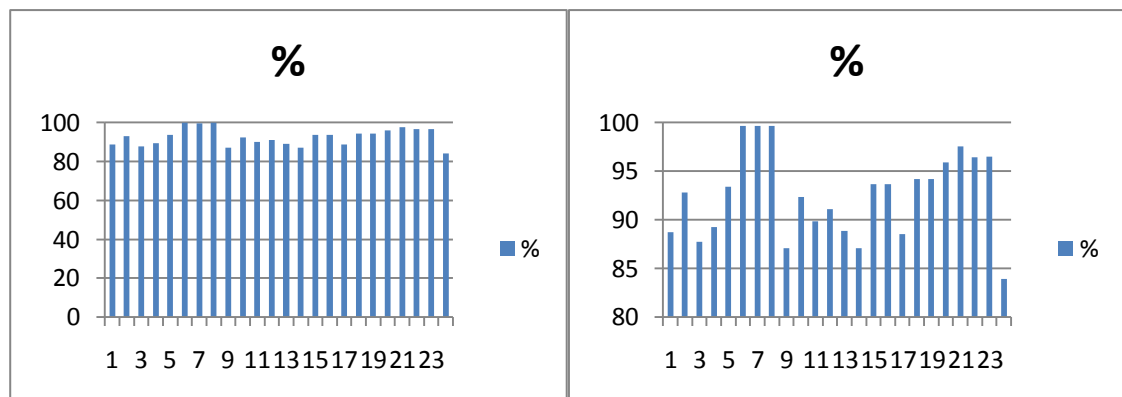


Figure 3.3 Examples of proper (left) and misleading (right) bar charts

Line charts

This is a type of chart which is created by connecting a series of data points together with a line. Line charts are best at representing continuous data and are especially useful for showing trends. It is very easy and fast to identify whether continuous data is showing an upwards or downwards trend. It is important to point out at this point, that occasionally neither is the case. This is just because the data can be random by nature. An example of a line chart is shown in Figure 3.4.

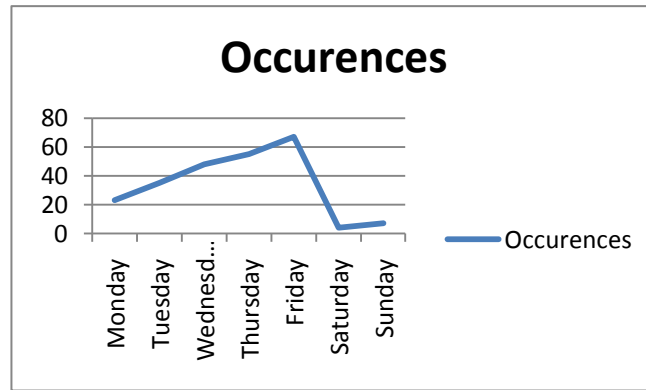


Figure 3.4 Example of a line chart

Link graphs

Link graphs, also known as semantic graphs, event graphs, network maps or link maps, are a kind of visualization method which consists of nodes and edges connecting the nodes. In directed link graphs there are arrows in-between, which represent the direction of the participating nodes. The perfect example here is representing the source and destination parties which participate in a network connection. Example of this can be seen in Figure 3.5.



Figure 3.5 Example of a link graph

Scatter plots

Scatter plots, also known as scatter charts, scatter graphs and scatter diagrams are another very popular and widespread method to visualize data. In this method, data is presented as a collection of points on a two dimensional coordinate system, and their purpose is to display the relationship between two variables, if any. They are best utilized to detect clusters and trends in data. Often in computer security this reveals if a given host is under port-scan attack. This is done when a set of destination IP addresses from a firewall log file are plotted against the ports of these destination addresses. An example of a scatter plot, which shows that a host is under port-scan attack, can be seen in Figure 3.6.

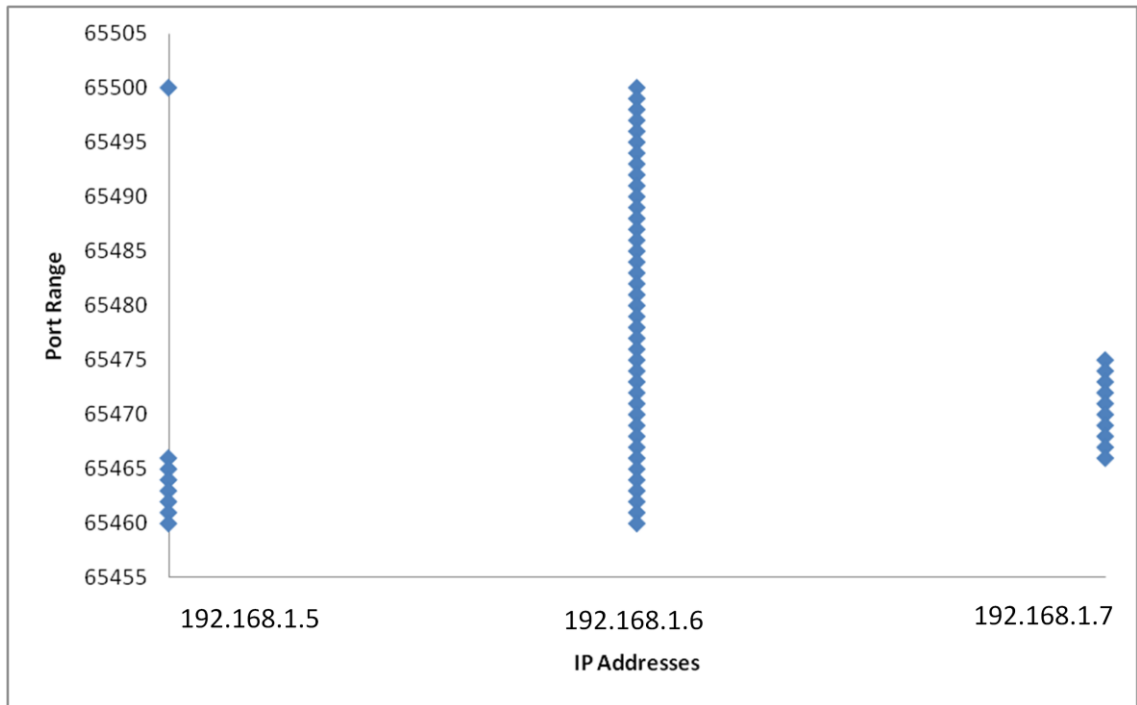


Figure 3.6 Example of a scatter plot which visualizes a port scan attack

Usage of colors in graphs

The usage of colors in graphs is not only to improve the aesthetics of the image, but can also represent an additional data feature in the graphs. It is important to emphasize at this point, that the usage of similar colors in a graph can cause difficulty in distinguishing differing data, so this should be avoided. This is the reason why the usage of colors is employed only in visualizing a small set of data values. In addition, it is a good practice to use colors which are culturally well accepted color assignments, like the colors of traffic lights. An example of how colors can support a report file from an IDS is shown in Figure 3.7.

Signature Name	Source Addr...	Destination A...	Sensor Nam...	Highest Seve...	Total Alarm ...
Too Many Frags	2	2	1	Informational	11373
IDS Evasive Encoding	1	1	1	Informational	4894
WWW Directory Traversal ..	1	1	1	Medium	3730
Oracle 9iAS Web Cache Buffer Overflow	1	1	1	High	3644
Lotus Domino database DoS	1	1	1	Low	3643
Tivoli Storage Manager Client Acceptor Overflow	1	1	1	Medium	3572
ICMP Echo Rply	3	2	1	Informational	3500
ICMP Echo Req	1	1	1	Informational	3374
Long SMTP Command	1	1	1	Medium	3084
Dot Dot Slash in HTTP Arguments	1	1	1	Medium	562
TCP SYN Port Sweep	2	2	1	Low	567
Unix Password File Access Attempt	1	1	1	Medium	383
mstream DDOS control traffic	1	1	1	Medium	204
WWW msadcs.dll access	1	1	1	Medium	164
WWW php view file Bug	1	1	1	Medium	147
WebSite uploader	1	1	1	Low	146
WWW finger attempt	1	1	1	Low	146
IOS Udp Bomb	1	1	1	Medium	142
WWW .bat file	1	1	1	Medium	138

Figure 3.7 Example of a color usage in a report from IDS

3.3 Information visualization process

Now that the reader is acquainted with the various types of data, data sources and visualization methods, it is appropriate to describe the whole information visualization process. This process consists of six main steps, which are [37, 12, Chapter 4]:

- define the problem
- access available data
- process information
- visual transformation
- view transformation
- interpret and decide

Define the problem

Information visualization can be a very challenging and demanding task, and it is not a straightforward process. It is very important that the goal or objective of the visualization is clearly defined at the very beginning. Information visualization should not be data driven, instead it should be use-case driven. The main questions for which the user must be prepared are:

- What is the user is interested in?
- What questions need to be answered by the graph which is about to be generated?
- What is it that the user wants to understand, communicate and explore?
- What is the user expecting to see?
- What would the use like to see?

This is the reason why the very first step of the visualization process should be defining the problem and establishing a goal [12, Chapter 4].

Access available data

Going further into the process, the next logical step after the definition of the problem is ascertaining what type of data is needed, and where it can be extracted in order to solve the defined problem. Of course there is no guarantee that this info is always available or that it will solve the issue, but it is worthy to try. Let us take the following example. A system administrator needs to understand what kinds of attacks are performed against the company in which he is working for, and also determine the geographical locations of the places from which attacks originate. In this case he needs the IDS logs from the NIDS which is deployed outside of the firewall border. This is because the log file generated by the firewall does not include such information (see Section 3.1.2).

Process information

Once the problem is defined and the source of data is available, the next step is to process the information. It happens very often in practice that the available data is not in

the right format. Thus, this data need to be transformed (parsed) into a format suitable for the security administrator, so that he can visualize and process it further (See Section 3.1.3). In addition to this it may be that there is too much redundant initial information, which is not relevant for the defined goals and tasks. This is the right place for the administrator to choose what to include and what to exclude from the selected data.

Visual transformation

The output from the previous steps should be data which is already selected, translated (parsed) and ready to be visualized. The next step in order for the security administrator to understand where the problem originates from is to choose which visualization method to use. This is another crucial point, because every visualization method is suitable for different purposes (See Section 3.2). Two very useful tools for this task are enclosed as appendices to this thesis. The first tool (Appendix A) is a table which summarizes different visualization methods. For each of the visualization methods, the graph summarizes the following features:

- number of data dimensions that can be visualized
- maximum number of data amount that can reasonably be displayed for each visualization method
- type of data which suits best the visualization method
- basic use-case scenario for which to use each graph
- sample security related application
- example to demonstrate the use of the method

The second tool (Appendix B) is a flow chart, which helps to decide what visualization method to be used in order to visualize the available data. Both tools are extracted from [12, Chapter 3].

View transformation

In addition to the previous step, often in practice it happens that the chosen visualization methods are not necessarily the ones which will help to solve the problem. This means the system administrator needs to test several methods to check which one will suit him best. However, it may be that there is nothing wrong with the chosen visualization methods, but that the selected data is wrong or insufficient (See Section 3.1.3). In this case the security administrator needs to go a few steps back, repeat some of the steps and fetch the required data.

Interpret and decide

In an ideal scenario, after going through all of the previous steps, the security administrator will end up with many graphs and possibly one of them will either show the solution to the problem, or will meet the required goal. However, very often the achieved result is not satisfying and established goals are not met. In this case the security administrator has to repeat the whole process again. Nevertheless, this is still a

very valuable experience, because the produced graphs may give him an insight to what is still missing or may show him another problem of was previously unaware. In Figure 3.8 all of these six visualization steps are visually represented.

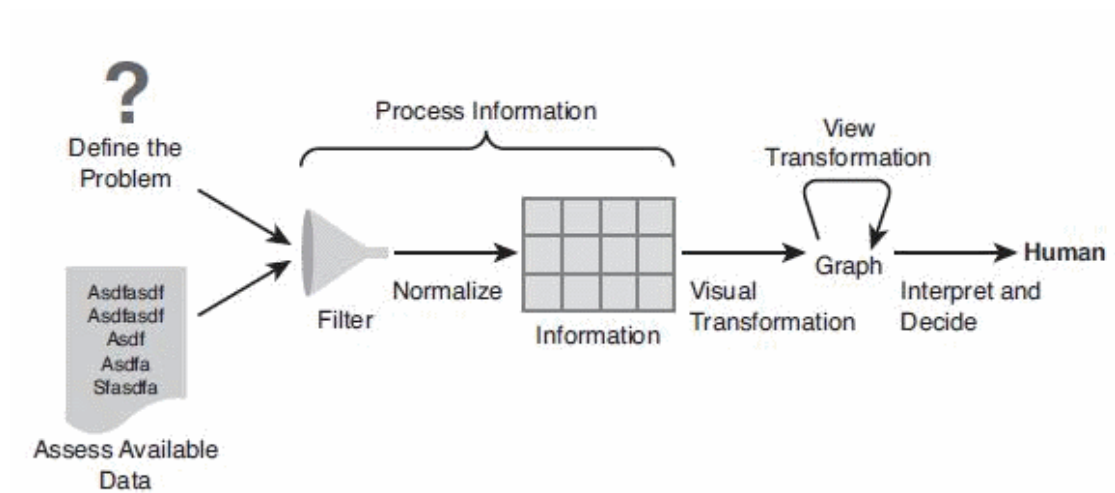


Figure 3.8 The six steps of information visualization process [12]

4 COMPARISON OF MONITORING USER INTERFACES

In keeping with the scope of this thesis, the following approach has been used in order to develop a GUI concept for an anomaly-based NIDS, designed for the mobile operator network environment. Firstly, a description of various security and network monitoring tools currently available on the market is given. Subsequently, a comparison and analysis of these tools is presented, in order to highlight useful features and practices and at the same time to exclude those that are irrelevant.

4.1 Description of the tools

Tools are evaluated based on the features of their GUIs, such as the usage of different colors, sizes, shapes and other visualization techniques. The evaluation of their core functionality, for instance their algorithms and methods is out of the scope of this comparison.

For the evaluation specific tools were chosen. An IPS tool, an IDS tool, a management tool for IDS and IPS solutions, a log monitoring and reporting tool, a vulnerability scanner and a network monitoring tool. These are:

- Sourcefire IPS - IPS
- Cisco IDS Event Viewer - IDS
- StoneGate Management Center – management tool
- Splunk - log monitoring and reporting tool
- Nessus - vulnerability scanner
- Cacti - network monitoring tool

Sourcefire IPS

Sourcefire IPS is a network security system based on Snort, which is an open-source intrusion detection engine and is considered as a de facto standard in the industry [43]. Sourcefire's GUI utilizes several different visualizations methods, such as:

- usage of colors
- bar charts
- line charts

In addition, it allows customizable and adjustable way of ordering the main screen. An example of a customized overview can be seen in Figure 4.1. In this case, the administrator has built his own event summary view. It shows that for monitoring purposes, the security administrator is interested in changes in the network events and if any rules or policies have been violated in the past one hour.



Figure 4.1 GUI of Sourcefire IPS [44]

Cisco IDS Event Viewer

Cisco IDS Event viewer is an application written in Java, which enables managing and monitoring alarm reports from a limited amount of sensors. At the time of writing this thesis, the limited amount of sensors is five [41]. This tool supports monitoring of alarms in real-time and/or performs investigation from an imported log file.

Alarms are visualized in a table and are separated into four categories. These are Low, Medium, High and Informational, and are colored correspondingly with yellow, orange, red and blue. A user can order them according their categories for a better overview. In Figure 4.2, a screenshot from Cisco's IDS Event Viewer is shown.

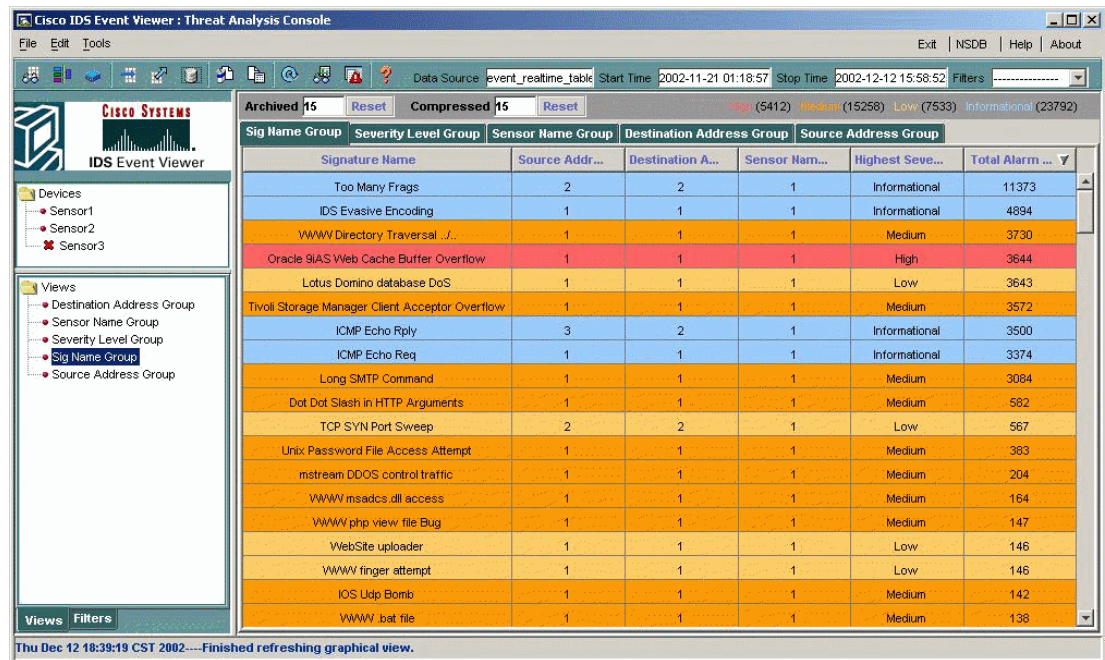


Figure 4.2 GUI of Cisco IDS event viewer [41]

StoneGate Management Center

StoneGate Management Center is a solution from Stonesoft, which allows users to control all of Stonesoft's other products, as well as manage products from other manufacturers in real-time from a single point. This system provides a single user interface which allows unified configuration, as well as monitoring and reporting of used products via a singular tool in the same user session. Stonesoft has decided to employ many visualization techniques in their user interface. Some of them are:

- colors
- link graphs
- pie charts
- line charts
- bar charts

Examples of the above mentioned methods can be seen in Figure 4.3. In addition, the Management Center's user interface is customizable and each user can include or exclude various modules and put them in an order which suits him best. A security administrator can choose to create his own overview from an empty template or can use one of the default overviews. An example of a sample overview configuration can be seen in Figure 4.4. In this example, the system administrator has chosen to monitor general system status characteristics, traffic flow, statistical charts of systems and network operations, as well as the geographical location of the monitored systems [42].

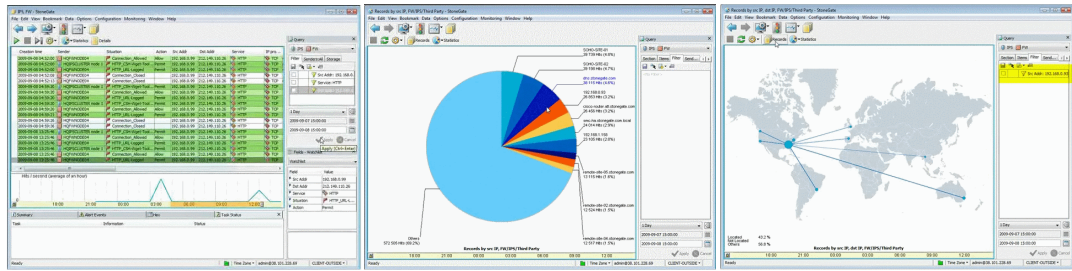


Figure 4.3 Some of the visualization methods used in Stonesoft's Management Center [42]

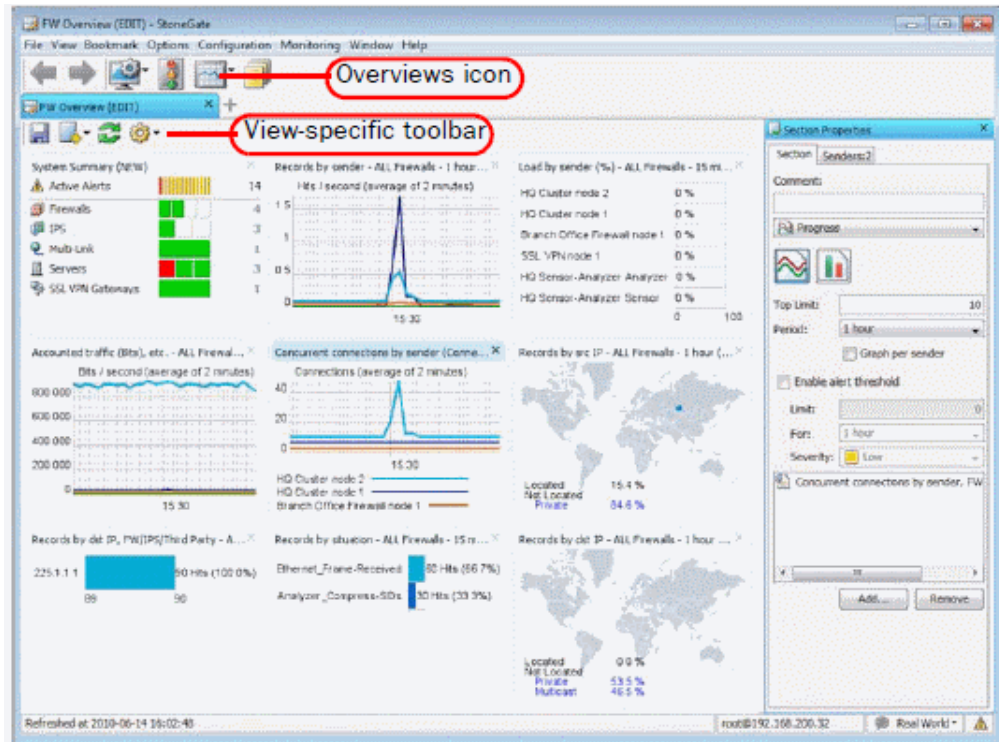


Figure 4.4 Sample overview of configuration in Stonesoft's Management Center [42]

Splunk

Splunk is a monitoring and reporting tool designed for system administrators. That which distinguishes this tool from other solutions in this area is its search capabilities and usability. Splunk is able to crawl in-depth through any kind of data sources given to it as a feed, and indexes this data in a searchable repository. After the data is stored, Splunk is able to generate graphs, reports and alerts based on this data. The main purpose of Splunk is to help system administrators with audit trailing in order to diagnose, identify and narrow the focus of investigating the problem with the aim of finding its causes.

As seen in Figure 4.5, a screen in Splunk's GUI is divided into four major parts. These are:

- search bar
- time line
- log record view

- field selector

In Splunk, searching interacts with imported content in a similar way to that of a search in Google. The content is filtered and highlighted, based on the user's query. In addition, the search language is easy and it supports auto-complete functionality, which greatly boosts the usability of the product.

The time line in Splunk visualizes the number of occurrences with a bar chart. The selection of the occurrences is done in three ways. First, are the results which match the queries from the search bar. Second, they can be selected from a calendar by pressing the corresponding button on the time line. And third, a user can browse by dragging the time line itself to the left and right. All occurrences are equally important and their intensity is shown with green bars.

Once the preferred time frame is selected and the desired occurrence is found, on the lower part of the screen is shown the actual log record or records corresponding to this event. Besides that, from the field selector on the left part of the screen, a user can include or exclude more fields from the imported data, which are associated with this event. If the operator is aware of what he is searching for, he can directly use this field selector instead of typing a query into the search bar.

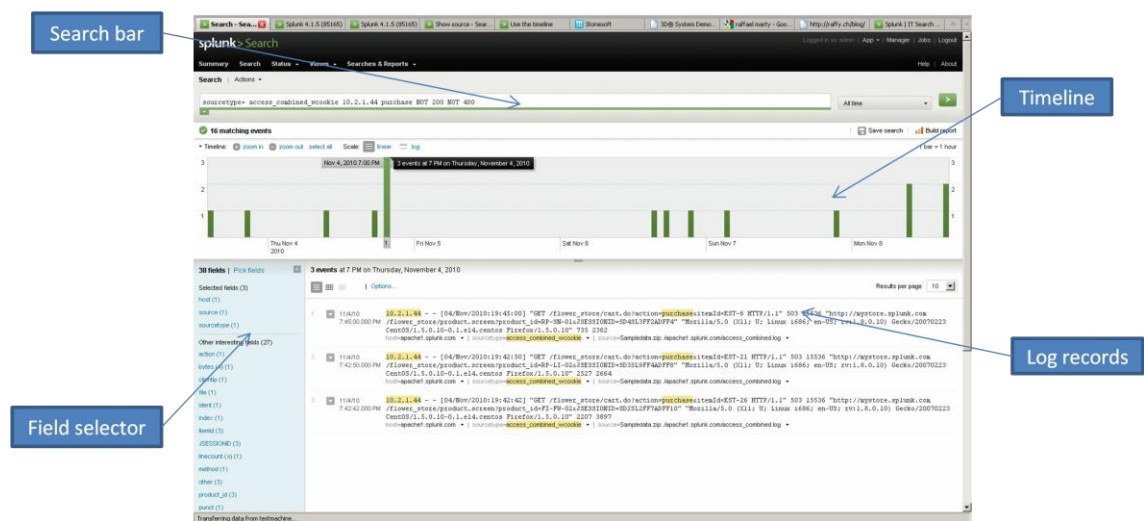


Figure 4.5 GUI of Splunk

Nessus

Nessus is the most popular vulnerability scanning and analysis tool [30], and it is designed to detect known security problems. One of the biggest advantages of Nessus is its client-server architecture. Server parts can be deployed at various strategic points in the network, and multiple distributed clients or a single central client can perform tests remotely. The server side is that which actually performs the checks. In a typical operation, these checks are actually port scans. After the port scanning is done, it tries

various exploits on the open ports. In addition to these checks, Nessus can use operating systems credentials, in order to examine existing patches and updates, as well as perform password auditing by using dictionary attacks and brute force methods. In Nessus' client side, configuration and reporting functionality are provided.

Nessus supports various types of security audits, such as [40]:

- credentialed and un-credentialed port scanning
- network-based vulnerability scanning
- credential-based patch audits for Windows and most Unix platforms
- credentialed configuration auditing of most Windows, Unix platforms
- robust and comprehensive credentialed security testing of 3rd party applications such as iTunes, JAVA, Skype and Firefox
- custom and embedded web application vulnerability testing
- SQL database configuration auditing
- Cisco Router configuration auditing
- software enumeration on Unix and Windows
- testing anti-virus installations for out-of date signatures and configuration errors

In Figure 4.6, a screenshot from Nessus' GUI is available. This is a screenshot from its client part which is accessible through a web browser. As seen in the screenshot, reports of different issues are divided into three groups. These are high, medium, low and are colored with red, yellow and green accordingly.



The screenshot shows the Nessus GUI with a report titled 'My Network Scan' for host '192.168.1.103'. The report displays a table of open ports and their associated services, categorized by severity (High, Medium, Low) and total count. The table has 9 columns: Port, Protocol, SVC Name, Total, High, Medium, Low, and Open Port. The data is as follows:

Port	Protocol	SVC Name	Total	High	Medium	Low	Open Port
0	tcp	general	6	0	0	6	0
80	tcp	www	6	0	0	5	1
123	udp	ntp	1	0	0	1	0
135	tcp	epmap	1	0	0	0	1
137	udp	netbios-ns	1	0	0	1	0
139	tcp	smb	2	0	0	1	1
443	tcp	www	6	0	0	5	1
445	tcp	cifs	11	0	1	9	1
34103	tcp	www	6	0	0	5	1

Figure 4.6 GUI of Nessus' client part

Cacti

Cacti is an open source, web-based tool for local or remote monitoring of different equipment and devices. The operation of Cacti can be divided into three main steps. These are:

- retrieving of data
- store data
- present data

The first of Cacti's main purposes is to poll data from the source at predetermined intervals of time. Cacti provides multiple data acquisitions methods [38], but the most common one for remote retrieving uses the Simple Network Management Protocol (SNMP). This means that any appliance which supports SNMP is eligible to provide information as a feed for Cacti and can be monitored by it.

The second step is to store the data. Often in such cases, the storing of data is done in a relational database or in a file. Cacti uses a different approach and stores its data in a Round Robin Database (RRD) by using the RRD Tool. The RRD is a system to store and display time series data, like network bandwidth, CPU load and hard disk utilization. It allows storing of the data in compact way and its primary advantage is that this saves space [38, 39].

The third step is to visually represent the fetched data by using different visualization methods, for example graphs. Cacti offers many predefined templates for creating graphs. The most commonly used ones are line charts. This is because Cacti's primary function is to visualize time-series data, and line charts are best at representing such type of data. In addition to this, Cacti's visualizing methods employ the usage of colors to represent more than one feature in the same graph. A screenshot of Cacti's GUI is shown in Figure 4.7.

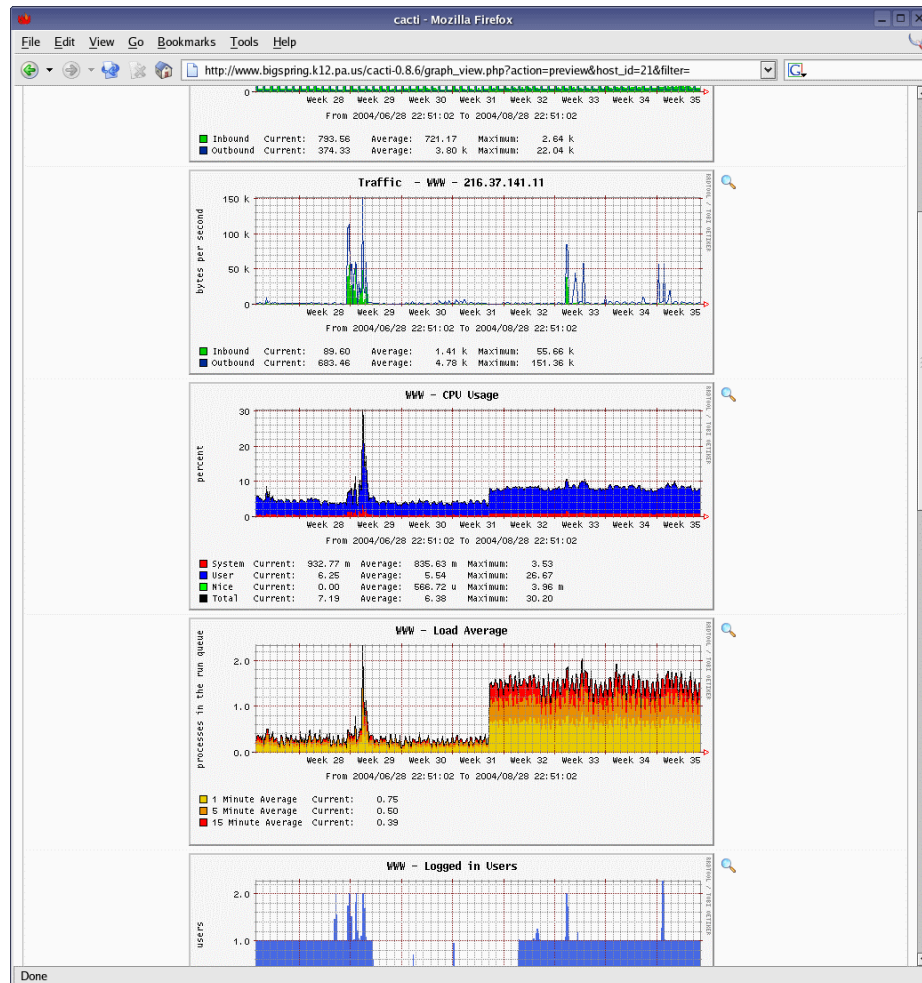


Figure 4.7 GUI of Cacti

4.2 Comparison and analysis

Compared to the old fashioned way of browsing manually through log files, a lot of work has been done in the past few years on security visualizations. This is true both in commercial and open-source products. Although some complex issues can be solved only by manually browsing through logs, this method is slow and it is not applicable in large scale networks. This is especially valid in a mobile operator network environment in which traffic volume is much bigger than that of enterprise networks. Comparison of the GUIs and their features from the above selected tools are summarized in Table 1.

It is interesting to note several trends in contemporary network and security tools which employ visualization methods. One of the trends is the usage of colors. All of the reviewed tools utilize colors as a method to represent some hierarchy or severity of an issue or event. The possibility for remote access is utilized by almost all of these solutions. Another trend is that almost all of these products employ line charts in order to visualize continuous data, as well as pie charts to visualize the percentage of a whole. The exception here is the tool from Cisco as it uses colors but not graphs. Going further,

it appears that the input information given to the tools as a feed is in the form of various types of log files. This is not applicable for Nessus solution. Nessus performs its own active scan of the desired target system and takes a snapshot of it as an input for further analysis. Another interesting trend in this review is that almost all of the tested solutions provide the user with functionality to customize their user interface. This is because different administrators have different working styles. This improves the usability of the products and the issues' resolution time.

One of the few features in which these tools differentiate, is with their reporting mechanisms. It should be taken into account that each of these tools has different functionalities and they are developed for different purposes, so this should not be considered as a drawback in their implementation. For example, Stonesoft and Sourcefire's solutions provide customization in their reporting functionalities. In addition, the solution from Sourcefire provides a default quick summary report (Figure 4.8). On the other hand Cisco's IDS and Nessus provide their reports in a table (Figure 4.2 and Figure 4.6). Cacti's report consists of a list of predefined graphs (Figure 4.7) and the report from Splunk shows the raw log file associated with the selected event (Figure 4.5). There is one more distinguishable characteristic in Splunk. This is its modular architecture design. While other implementations are using a monolith architecture type, in which all functionalities are already defined and implemented, Splunk provides the user with the opportunity to import more applications, even those written by third parties. This is in order to take advantage of Splunk's GUI implementation. This agrees with the idea presented in Section 3.3, namely that the information visualization process should be use-case driven, not data driven. This functionality enables the solution from Splunk to import modules for different problem scenarios and use-cases which need to be solved.

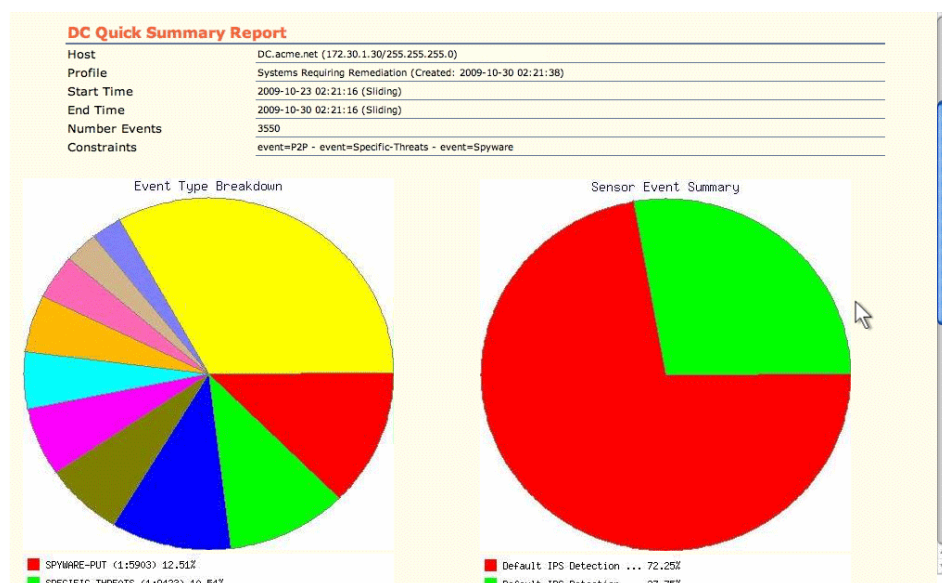


Figure 4.8 Screenshot of a report from Sourcefire system [44]

Table 1 Summary of GUI features comparison

Features	Sourcefire System	StoneSoft Management Center	Cisco IDS Event Viewer	Splunk	Nessus	Cacti
Usage of colors	yes	yes	yes	yes	yes	yes
Remote access	yes	yes	N/A	yes	yes	yes
Visualization Methods	line chart, pie chart	line chart, pie chart	table/ none	line chart, pie chart, bar chart	line chart	line chart, pie chart
Type of input data	log	log	log	log	snapshot /report from a scan	log via SNMP
Resolving problem starting point	customized screen	customized screen	summary in a colored table	customized screen	summary in a colored table	customized list of graphs
Error information	summary report; customized report	customized report	list of triggered alarms in a table	raw log(s) associated with the selected event	list of ports in a table as well as vulnerabilities and their severity to the system related to them	peaks and downfalls in a graphs

5 SECURITY VISUALIZATION DEMONSTRATION

In this chapter I will present my GUI concept of an anomaly-based NIDS for a mobile operator network environment. First, a short description about the system for which this interface is designed is provided. Then, a screenshot from the default view of this concept is presented. After this, each individual element is introduced and its functionalities are explained. Henceforward, a section with outlining the GUI requirements when implementing a prototype based on this GUI is offered. The chapter ends with the descriptions of two imaginary use-cases in which this interface can facilitate the work of a network administrator, in order to solve complex environment problems and perform root cause analysis.

5.1 GUI concept

Keeping the general workflow of system and network administrators in mind, I have developed a GUI concept, with the goal to enable quick visual insight into communication patterns and network behavior. A partly functional mockup of the GUI was developed, in order to experiment with the concept design. This concept is designated for an NIDS with modular architecture. This means that, functionalities which are developed in the future can be easily added to the GUI as modules. Figure 5.1 depicts the general architecture of such a system and a screenshot of this interface is available in Figure 5.2.

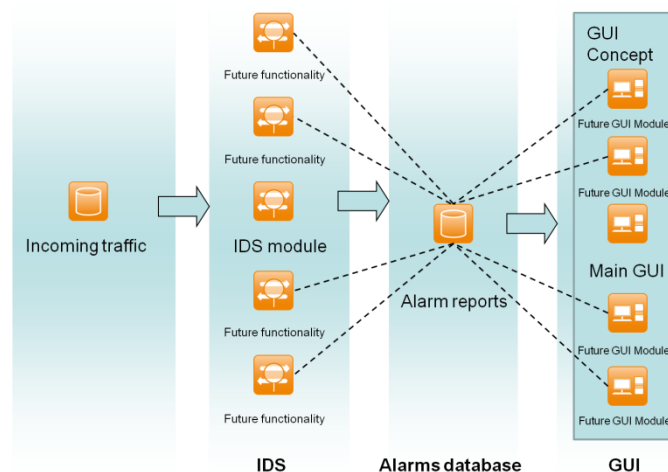


Figure 5.1 NIDS with a modular architecture

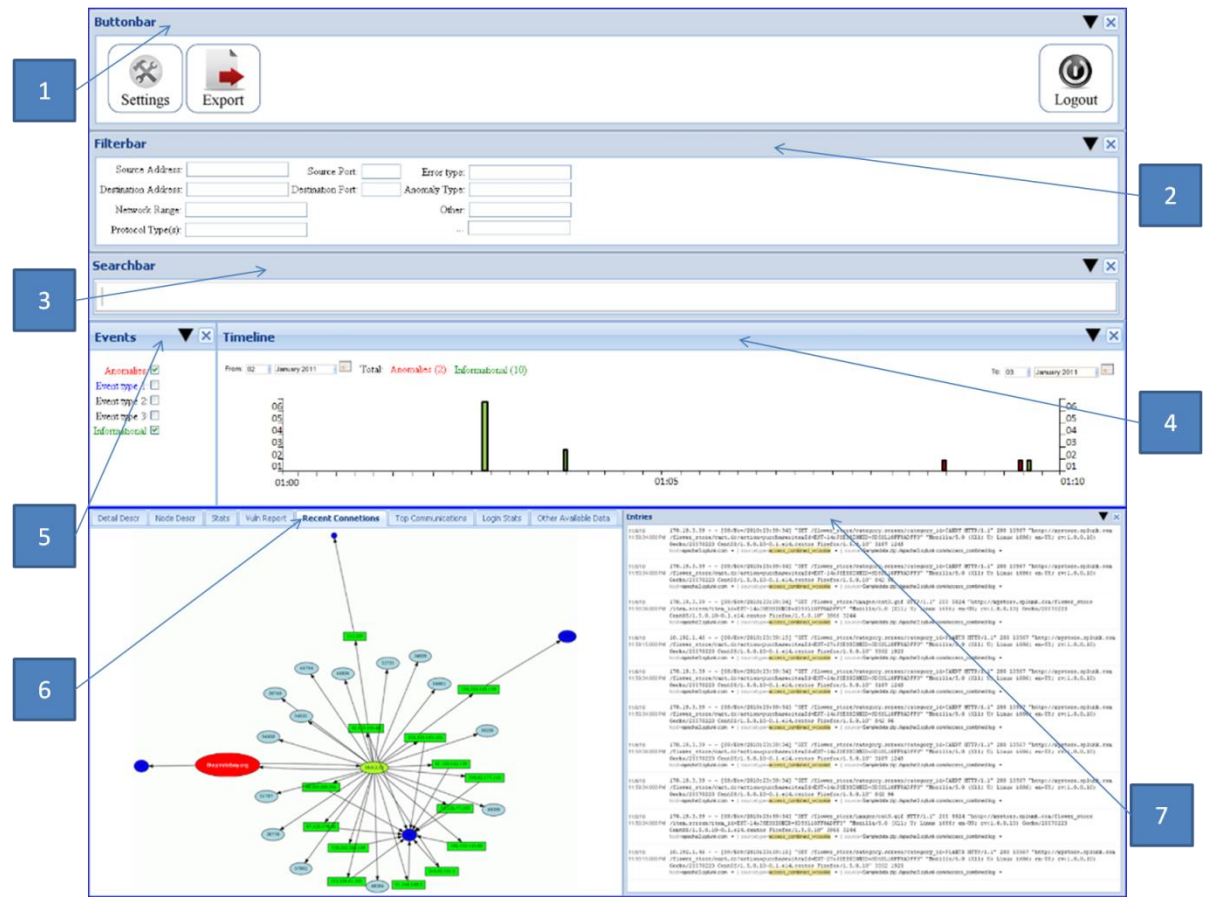


Figure 5.2 GUI concept

5.1.1 Components of UI

As seen in Figure 5.2, the default view of the main screen for this GUI concept is divided into seven panels. These are:

1. buttonbar
2. filterbar
3. searchbar
4. timeline
5. events
6. modules panel
7. entries

1. Buttonbar

A screenshot of the “Buttonbar” panel is available in Figure 5.3. This is a standard button bar, which contains a predefined set of buttons. Such buttons could include settings, export, and logout. It is important to emphasize that the selection and ordering of the buttons should be customizable.

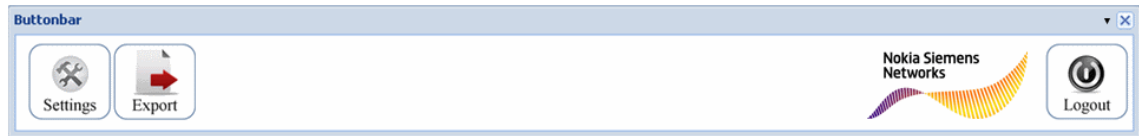


Figure 5.3 Buttonbar

2. Filterbar

A screenshot of the “Filterbar” panel is available in Figure 5.4. This panel should contain input cells, which provide the administrator with the functionality to perform fast selections of the recorded events in the system. The order, size and types of the cells should be customizable. Imagine a case in which a junior administrator with no previous knowledge of the system or SQL-like language, wants to select specific kinds of anomalies from a given sub-network. In this case, he will need to fill only two cells and he will receive a summary report about them. Further, he could also add additional filtering by selecting all of those anomalies in a specific sub-network for the last twenty four hours. Administrators can adjust the content of this panel in a way which suits them best.



Figure 5.4 Filterbar

3. Searchbar

A screenshot of the “Searchbar” panel is available in Figure 5.5. This panel is targeted to more experienced administrators. In this bar more complex selections from the database with the recorded events can be performed. The selection language should be relatively easy and close to human language. This language should be SQL-like and should be able to support regular expressions. This panel should be used in addition or instead of the “Filterbar”, but it is optional. For example, the administrator can have both panels open, with a predefined selection of cells in the “Filterbar” for trivial and routine cases. He can then to use the “Searchbar” when more complex situations occur.



Figure 5.5 Searchbar

4. Timeline

A screenshot of the “Timeline” panel is available in Figure 5.6. The “Timeline” is the main reporting and summarizing tool in my concept. All events in this concept, regardless of their type and severity to the system are depicted with bars. The height of the bar represents how many occurrences of these type happened within the selected timeframe, and the color of the bar determines the type of the occurrence. It should be

taken into account, that in order to best optimize the functionality of the “Timeline”, no more than five types of events should be presented at the same time. This is for two main reasons. First, there is a limitation from the color representation point of view (see Section 3.2), and second, to avoid redundancy on the screen. This means that some aggregation of events must be done. For example, successful and failed logins to the system as well as system restarts and schedule maintenance could be aggregated as “Informational” events.

Selection of the time frame can be done in multiple ways. First it can be done by selecting the days from the calendar buttons on the “Timeline”. Calendar buttons are intentionally placed on the furthest left and right sides of the “Timeline”. The logic here is that all occurrences will be displayed between the “from” and “to” calendar buttons. Another way to find occurrences is to browse by dragging the “Timeline” to the left and right or to select the time frame from the corresponding input cells on the “Filterbar”.

A very useful feature in the “Timeline” according to my vision, should be its drill-down functionality. This means that the “Timeline” should support time frames from one year to one second. A detailed explanation about how this feature works and about the behavior of the “Timeline” will be provided further in this thesis in the use-case section (Section 5.2).

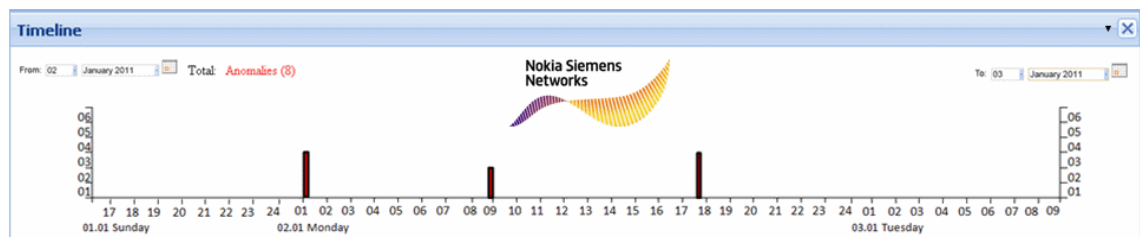


Figure 5.6 Timeline

5. Events

A screenshot of the “Events” panel is available in Figure 5.7. The “Events” panel is to be used in addition with the “Timeline”. From it, an administrator can select the type of occurrences which he wants to include or exclude from the “Timeline”. This can be done simply by clicking on the corresponding checkbox. More detailed information how this bar facilitates an administrator’s work will be provided further in this thesis in the use case section (Section 5.2).

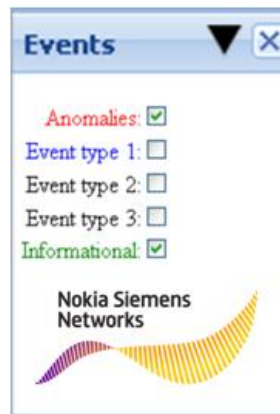


Figure 5.7 Events panel

6. Modules panel

A screenshot of the “Modules” panel with a selected tab on it is available in Figure 5.8. This is the panel which will provide more detailed information about the selected occurrences from the “Timeline”. On it, there should be as many tabs as necessary which contain additional information, functionalities and visualizations about the selected event, its source and other relevant information. This is in order to facilitate the root cause analysis for the selected event. In addition, any future functionality of the system can be represented as an additional tab on this panel. The order of the tabs should be customizable similar to tabs in contemporary web browsers, like Mozilla Firefox. In case the amount of tabs is too large to be visualized in one row, they can be visualized in two or more rows. Additionally, there should be a functionality which supports excluding or including them. More detailed information of how this panel facilitates an administrator’s work will be provided further in this thesis in the use-case section (Section 5.2).

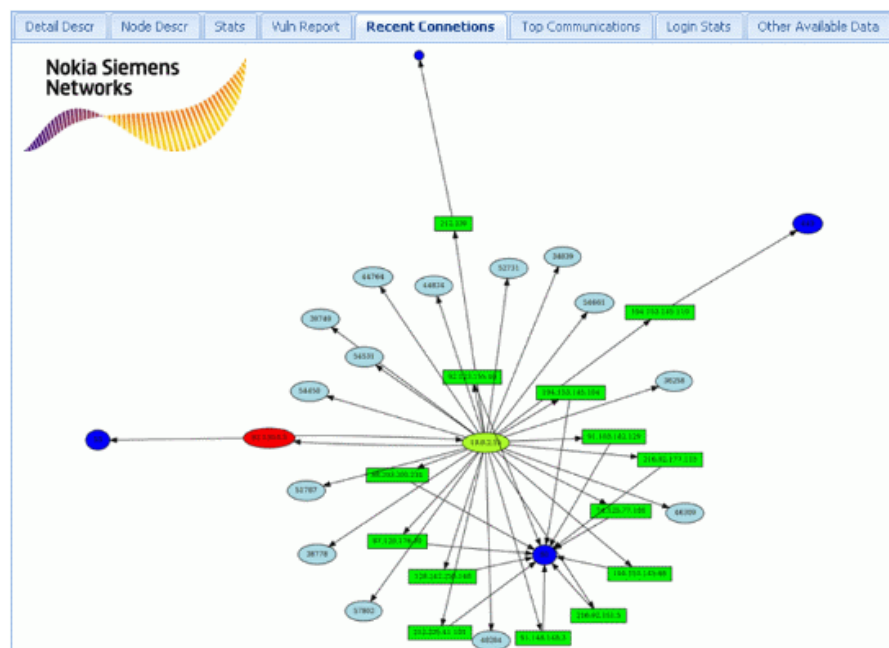
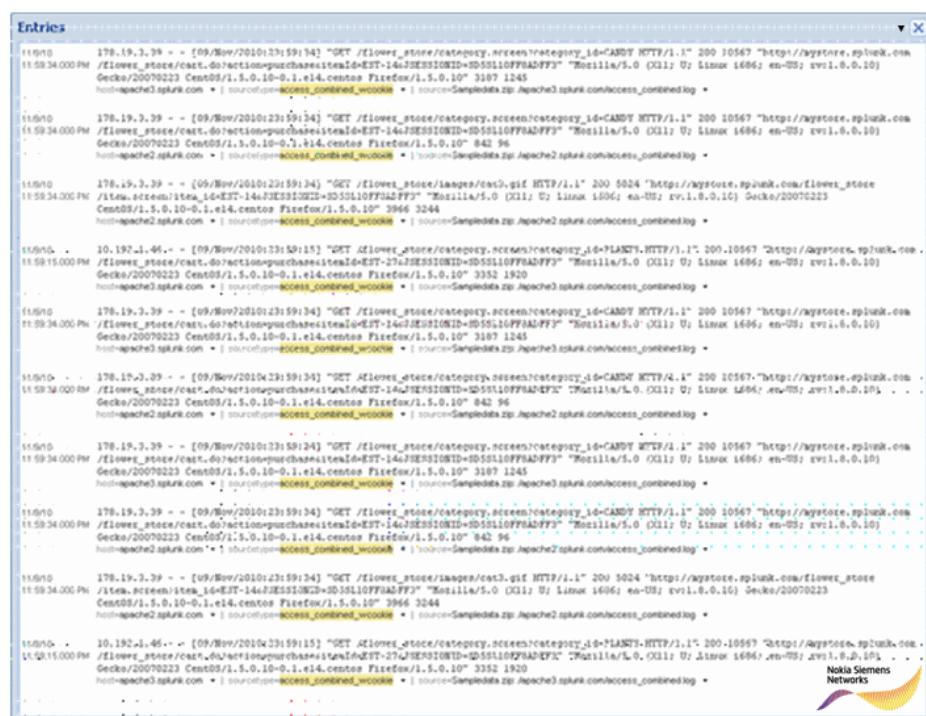


Figure 5.8 Modules panel

7. Entries

A screenshot of the “Entries” bar can be seen in Figure 5.9. The purpose of the “Entries” panel is to show all the data related to the selected event in its raw format. This data could be pure log data from a server. This panel is designated for the more experienced administrators or for those who are aware of the format of the represented data. As in any other panel in my concept, this one should be also optionally displayed. Its main purpose is to present the raw data in order to determine if the selected event is properly categorized. This is especially useful for double checking, if the module responsible for anomaly detection has properly categorized the selected event, or if it is just a false alarm.



```

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3187 1245
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 842 96
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966 3244
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 10.192.1.46 - [09/May/2010:23:59:15] "GET /flower_store/category.screen?category_id=PLANET HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:15:00 PM /flower_store/cat.do?action=purchase&itemId=EST-27&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3352 1920
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3187 1245
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 842 96
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3187 1245
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 178.19.3.39 - [09/May/2010:23:59:34] "GET /flower_store/category.screen?category_id=CARD HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:34:00 PM /flower_store/cat.do?action=purchase&itemId=EST-14&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3966 3244
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

11:59:00 10.192.1.46 - [09/May/2010:23:59:15] "GET /flower_store/category.screen?category_id=PLANET HTTP/1.1" 200 10567 "http://mystore.sp1nuk.com
11:59:15:00 PM /flower_store/cat.do?action=purchase&itemId=EST-27&SESSIONID=3D5510FF9A2FF3" Mozilla/5.0 (X11; U; Linux i686; en-US; rv:1.8.0.10)
Gecko/20070223 CentOS/1.5.0.10-0.1.el4.centos Firefox/1.5.0.10" 3352 1920
host=apache2.splunk.com * | sourceType=access_combined_wcookie * | source=Sampledata.zip:Apache2.splunk.com/access_combined.log *

```

Figure 5.9 Entries

5.1.2 GUI development requirements

According to my vision, there are several requirements which should be taken into account when implementing a prototype based on this concept.

First, such a system should allow remote access to it. The main benefit of this is that the system administrators are not required to be at the same physical location as the monitoring system. The advantages for a system being accessed remotely are many. Imagine a case in which there are several distributed monitoring systems for backup or availability assurance purposes, but only one or a few system administrators who are

able to monitor them or have the rights to access them. It is far more convenient for them to access these systems remotely.

Second, this solution should support customizable report and summary functionality. The main argument for this is the modular architecture of such a monitoring system. As it has been pointed out, visual representations should be use-case driven, rather than data driven. This means that each use-case should be visually represented in the best way suitable for its particular case. Thus, each use-case can have a special module for it, and a special summary report or default view which best depicts its current state. In addition, modular architecture supports future modules for various use cases to be easily embedded in the system. Finally, each security administrator may have his own working practices and preferences, so customization of his working environment is crucial part in order to ensure that his time is best utilized, as well as to reduce the time he spends resolving different issues.

5.2 Use cases

Start of a security administrator's work shift

As it was said earlier, this concept has been designed for an anomaly-based NIDS and it is intended to be used by network administrators within a mobile operator. Let us imagine a situation where a network administrator comes to work in the morning. Most probably he would like to check first what was happened during the past 24 hours or since his last login to the system. The administrator authenticates himself to the system. With this GUI concept, he needs to select the time frame in which he is interested. The system can be set in such a way that it automatically shows a summary since his last login or of the past 24 hours. We will take the case in which the summary of the events is shown since his last login (Figure 5.10).

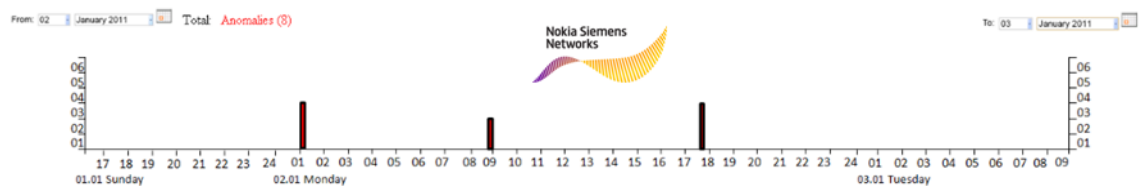


Figure 5.10 Selection from the “Timeline” – step 1

In this simulation, the administrator can see summarized eight anomalies within the given time range. Following the same logic of software development, specifically that the first error should always be corrected first (because the others may be consecutive to it), investigation should start from the first reported anomaly. This assumption is discussed in detail further in this thesis in Chapter 6. Now, the administrator has to click on the first bar, where three anomalies are recorded in the system, slightly after midnight at one o'clock (Figure 5.11).

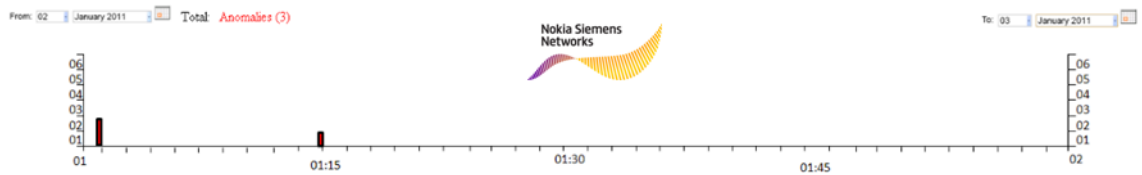


Figure 5.11 Selection from the “Timeline” – step 2

Once he has clicked on this bar, the user can see the drill-down functionality of the “Timeline”. What is now shown on the “Timeline” is a more fine-grained time frame within which these three anomalies occurred. The first two of these three anomalies occurred at almost the same time, around one o’clock, and the third one occurred close to 01:15. Now the administrator is interested in the first two occurrences and would like to know more about them. So after this, he clicks on the bar which represents the first two anomalies and now he is able to see an even more detailed time frame, in which these two anomalies are clearly distinguishable events (Figure 5.12).

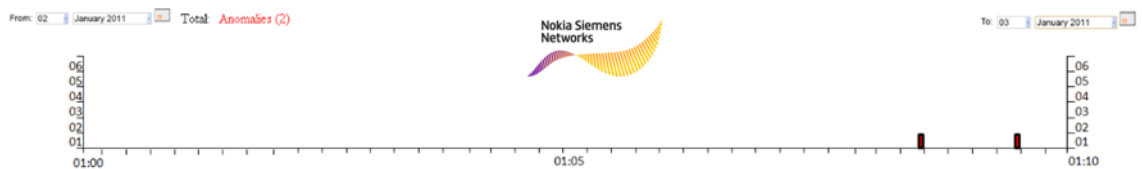


Figure 5.12 Selection from the “Timeline” – step 3

In this simulation he may want to see if there are other types of events which have occurred within this time frame. This is just for the administrator’s information, and it is shown in this case just to inform the reader how the “Events” bar correlates with the “Timeline” and that such a functionality exists. Now the administrator wants to include other types of events into the summary on the “Timeline”. This is done by clicking on the corresponding check box from the “Events” panel, and in this case, the administrator includes “Informational” events to the “Timeline”, which have occurred in this time frame (Figure 5.13).

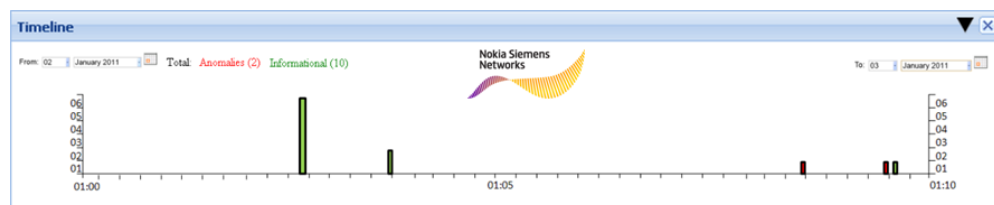


Figure 5.13 Selection from the “Timeline” – step 4

Returning to the anomaly in which the administrator is interested, it is time to introduce the “Module” panel. In its first tab, detailed information is displayed about the selected anomaly. Such information may be the type of anomaly, which node generated it, why

this is considered as an anomaly, and which rule it violates. In this case let us assume that this particular anomaly is correctly identified and it is a known behavioral pattern.

The first tab is labeled “Detail Descr” (Figure 5.14). In this particular case it shows that the anomaly is an open port (445) on a host. It also shows the anomaly’s severity to the system, a solution for this issue and other relevant information related to this occurrence.

Detail Descr Node Descr Stats Vuln Report Recent Connctions Top Communications Login Stats Other Available Data

Plugin ID: 26919 Port / Service: cifs (445/tcp) Severity: **Medium**

Plugin Name: SMB Guest Account Local User Access

Synopsis
It is possible to log into the remote host.

Description
The remote host is running one of the Microsoft Windows operating systems. It was possible to log into it as a guest user using a random account.

Solution
In the group policy change the setting for 'Network access: Sharing and security model for local accounts' from 'Guest only - local users authenticate as Guest' to 'Classic - local users authenticate as themselves'.

Risk Factor
Medium

CVSS Base Score
5.0 (CVSS2#AV:N/AC:L/Au:N/C:P/I:N/A:N)

CVE
CVE-1999-0505

Plugin Publication Date: 2007/10/04

Nokia Siemens Networks

Figure 5.14 Module bar – “Detail Descr” tab. Picture is based on [40]

To investigate further, the administrator wants to know more about this particular node, and to do so, he selects the “Node Descr” tab (Figure 5.15). In it, all available information about this node is shown, such as: its IP address, MAC address, hardware specifications, operating system, geographical location, contact information, the sub-network to which it belongs, and the person responsible for this node.

The screenshot displays a network management interface with several tabs: Detail Descr, Node Descr, Stats, Vuln Report, Recent Connctions, Top Communications, Login Stats, and Other Available Data. The 'Node Descr' tab is active, showing a detailed description of a network element.

Hardware		Contact person	
CPU	Intel E6600	Name	Alexander Zahariev
MotherBoard	Asus P5B	Position	Tech Support
HDD	Seagate Baraccuda 7200 100GB	Location	Finland, Espoo, Saterinportti
Video	GeForce 7800 GT	E-mail	a.zahariev@abv.bg
RAM	Kingston 2x2GB	Phone	+358449983423
LAN	Gigabit Ethernet		
WLAN	intel wireless 3945abg		

Network Information	Geo Location
Mac Address 00:18:DE:4C:B4:F9	
IP Address 192.168.1.101	

Network MAP

The network map shows a central 'Network Switch' connected to several components: 'Corporate DB Server', 'Corporate Web Server', 'Some Modem', 'Some Fax', 'Network Printer', 'Work Station 2', 'Work Station 1', and three 'Some [Small/Big] Network Group' nodes.

Nokia Siemens Networks

Figure 5.15 Detailed description of a network element

Then the administrator is interested in the latest vulnerability scan for this particular node (Figure 5.16). This information is available on the “Vuln Report” tab. This could be a similar functionality to that which the Nessus Vulnerability Scanner provides [40]. In this tab, this information is available and it is summarized in a table. In this case, it shows the scanned ports, issues related to them and severity to the system, if any. In this case there are several reported issues for this node.

The screenshot displays the 'Vuln Report' tab for a network scan of IP address 192.168.1.103. The table shows 11 results.

Port	Protocol	SVC Name	Total	High	Medium	Low	Open Port
0	tcp	general	7	0	0	7	0
80	tcp	www	6	0	0	5	1
123	udp	ntp	1	0	0	1	0
135	tcp	epmap	1	0	0	0	1
137	udp	netbios-ns	1	0	0	1	0
139	tcp	smb	2	0	0	1	1
443	tcp	www	6	0	0	5	1
445	tcp	cifs	12	1	1	9	1
1241	tcp	nessus	6	0	0	5	1
8834	tcp	www	10	0	0	9	1
34103	tcp	www	6	0	0	5	1

Nokia Siemens Networks

Figure 5.16 Vulnerability report tab. Picture is based on [40]

To conclude this use-case, the only thing which the administrator has to do, is to contact the responsible person for this network element, and to remind him to apply the latest

hardening practices available for this node. In the case that he is authorized to do it on his own, the administrator can remotely connect to it, and do the necessary procedures in order to avoid future triggering of anomalies in this monitoring system.

Investigation of a denial of service situation

The second use case will be related to an anomaly associated with a node which is not accessible and I will call it DoS. In order to avoid redundancy, the case will start from the position which the administrator has already found and selected via the “Timeline”– the anomaly entry in which he is interested. The investigation begins from the point in which the administrator reads the detailed information about the chosen anomaly in the “Detail Descr” tab from the “Modules” panel.

Once the administrator has selected the anomaly in which he is interested, he has to read the detailed description available for it from the “Detail Descr” tab on the “Modules” panel (Figure 5.17). This time the anomaly description says that a node in a subnetwork is not responding to automated ping requests. This could be a similar functionality like the one provided by Nagios [46], which sends a ping request at predefined time intervals to a network element in order to ensure this element is active. By doing so, Nagios monitors node status. In our case, the violated rule says that a node didn’t respond to three of these requests, so an anomaly entry has been recorded in the system.

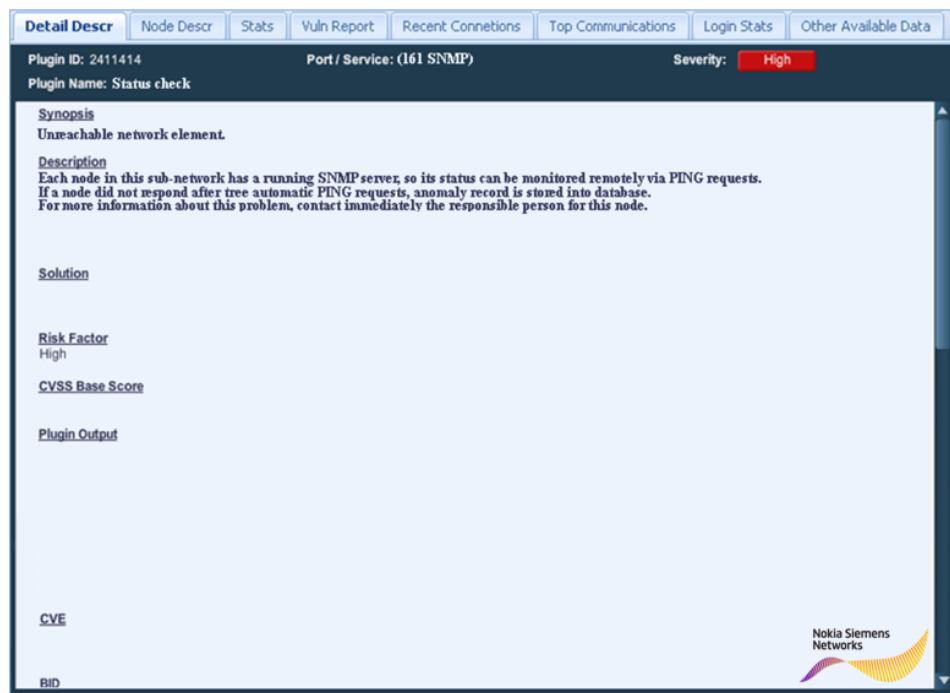


Figure 5.17 Module bar – Detailed description tab. Picture is based on [40]

The next logical step which the administrator can do is to manually check if this node is responding to ping commands (Figure 5.18). This is in order to double check if the system which is responsible for performing automated checks is working. To do so, we

assume that in this system there is a module with a similar functionality to the “ping” command. This is visualized in a separate tab, so the administrator can check this manually from there.

The screenshot displays a network monitoring application with a table of host status and a terminal window. The table has columns for Host, Status, Last Check, Duration, and Status Information. The 'Status' column uses color-coded balloons: green for 'UP' and red for 'DOWN'. The 'Status Information' column provides details like 'PING OK - Packet loss = 0%, RTA = 72.88 ms' or 'CRITICAL - Host Unreachable (10.0.0.238)'. Below the table, a terminal window titled 'C:\WINDOWS\system32\cmd.exe' shows the execution of a ping command to 192.168.101.254, resulting in four 'Request timed out' messages and a summary: 'Ping statistics for 192.168.101.254: Packets: Sent = 4, Received = 0, Lost = 4 (100% loss)'. The Nokia Siemens Networks logo is visible in the bottom left corner.

Host	Status	Last Check	Duration	Status Information
corne	UP	11-02-2005 11:10:08	3d 2h 22m 26s	(Host assumed to be up)
shewood	UP	11-03-2005 13:32:45	4d 11h 59m 1s	PING OK - Packet loss = 0%, RTA = 72.88 ms
zhan	UP	11-02-2005 11:13:49	7d 2h 13m 3s	(Host assumed to be up)
lin	UP	11-02-2005 11:10:34	7d 2h 29m 52s	(Host assumed to be up)
huoqerv	UP	11-02-2005 11:10:47	7d 2h 29m 39s	(Host assumed to be up)
isa	UP	11-02-2005 11:14:15	7d 2h 12m 37s	(Host assumed to be up)
lun	DOWN	11-03-2005 13:51:33	0d 4h 13m 26s	CRITICAL - Host Unreachable (10.0.0.238)
overledge	UP	11-03-2005 13:33:35	4d 3h 43m 40s	PING OK - Packet loss = 0%, RTA = 59.19 ms
lusa	UP	11-02-2005 11:14:41	2d 1h 57m 56s	(Host assumed to be up)
span	UP	11-02-2005 11:11:26	7d 2h 15m 26s	(Host assumed to be up)
luc	UP	11-03-2005 05:51:55	4d 3h 43m 41s	PING OK - Packet loss = 50%, RTA = 1391.42 ms

```

C:\>ping 192.168.101.254

Pinging 192.168.101.254 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.101.254:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
C:\>

```

Figure 5.18 Module for various functionalities

What comes next depends entirely on the working practices of each individual administrator. He may check, as in previous use case, the “Node Descr” tab and/or its latest “Vuln Report” tab.

For example, the administrator’s next steps may be to check the “Recent Connection” tab for this node (Figure 5.19). What he sees in it, is that normal web-browsing has been conducted from this host, but one of its recent connections seems more suspicious. It is depicted with a balloon which has a larger size and its color is red. The size of the balloon represents the amount of the connections which originate from this host. Color is used in order to ensure that balloons with similar sizes will be noticed, but this entirely depends on the configuration file, which is used to create this visualization. This functionality is provided by AfterGlow [47], and when implementing a prototype based on this concept, similar functionality could be used.

Returning to this particular connection, it represents a connection to “The Pirate Bay” torrent tracker web site. One can conclude from this, that the whole network bandwidth is taken by torrents and there is no space left for the legitimate network operations. Such operations could be automated ping requests.

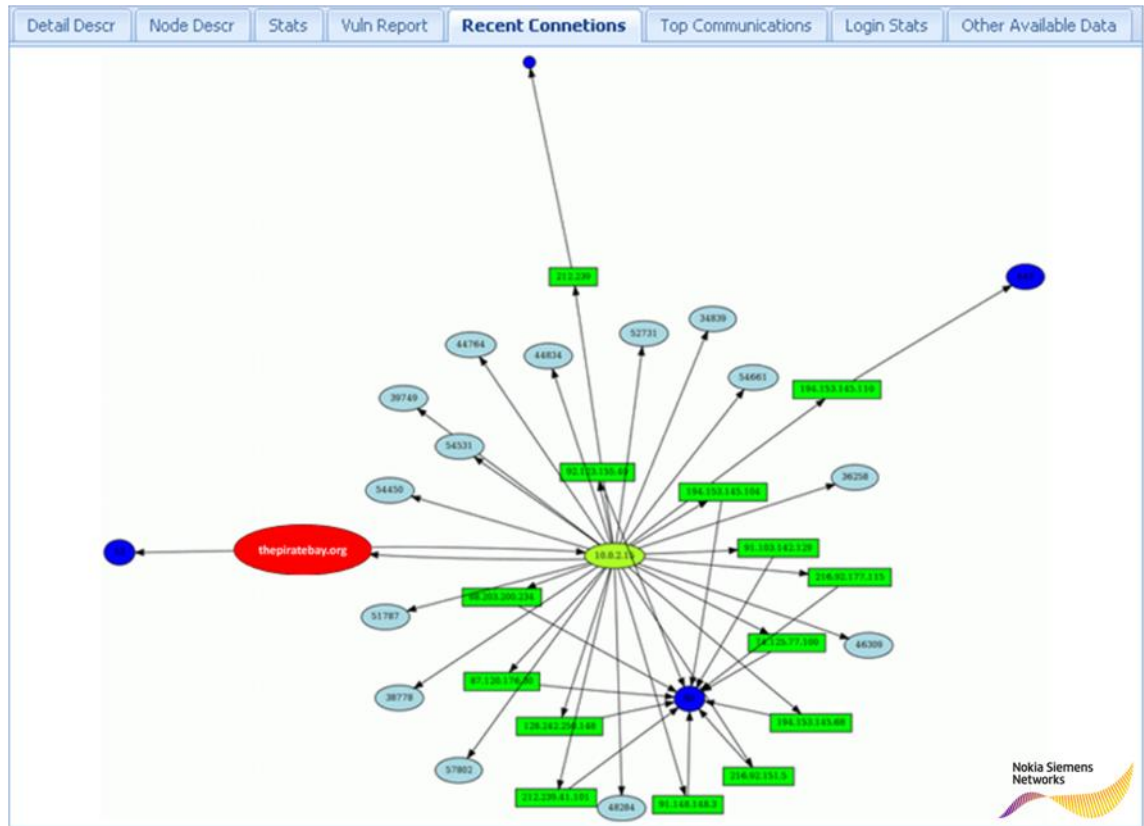


Figure 5.19 Recent connections tab

The next logical step for the administrator could be to check the node’s login statistics (Figure 5.20). This can be done from the “Login Stats” tab. From this tab the administrator can see the successful and failed login attempts to this node as well as information about which user is currently logged-in to the system, or who was the most recently logged in user before this node became unreachable.

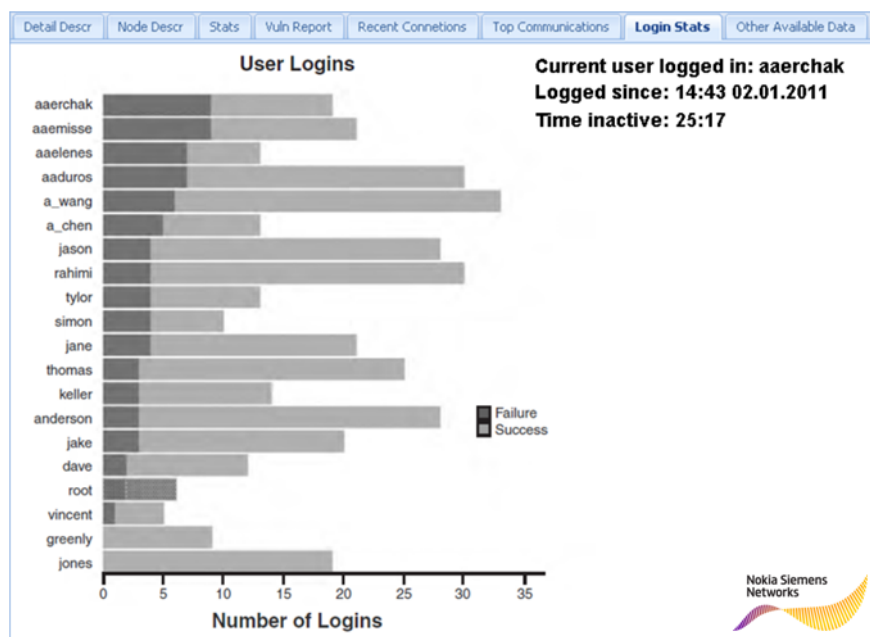


Figure 5.20 Login statistics tab

To conclude this second use-case, the last thing which the administrator could do in this situation is to select the “Node Descr” tab (Figure 5.15) and to check who is responsible for this node. Then, he can inform the responsible person about this case, and provide more detailed information. Such information could include the fact that this particular node is not responding, as well as which user was last logged-in to it and what were his latest activities with it. In this case, a connection to a torrent tracker web site.

These component descriptions and use-cases provide an example of how this GUI concept might be employed in a working environment. The way in which the elements intrinsic to my design and features included in the GUI concept facilitate a system administrator’s daily tasks is further discussed in the following chapter.

6 DISCUSSION

The purpose of this thesis was to create an overview of the current situation with network monitoring and reporting tools, to extract the best practices from them and to produce a GUI concept for an anomaly-based NIDS designed for a mobile operator network environment. The reason behind this is that, although a significant amount of work has been done for the enterprise environment, at the time of writing this thesis, there is no customized solution for the telecommunications environment.

Keeping the general workflow of the system and network administrators in mind, a GUI concept was developed with the goal of enabling quick visual insight into network behavior. When I was a system administrator of a medium size company, I had to deal with more than two hundred user machines, multiple servers and various appliances which were connected to the network. The main challenge in administrating all of this equipment was that I had to use different solutions to manage them. In some cases detecting similar or even the same problem on different machines was possible only by using proprietary tools, designed for each product. No single solution was available on the market which could allow me to control products shipped by various suppliers from a single point.

Based on the above discussion, I believe that a product with a modular design is the key to coping with the challenge of managing various appliances. This is because any new functionalities developed in the future can then be imported to the product as a plug-in. But, in order for this to be possible, the GUI should support visualization of modules and functionalities developed in the future. For example, let us imagine a case in which an anomaly is reported to the system and is related to the network traffic. In order to double-check if this anomaly is properly classified, we need to capture and observe the traffic. In this case we can use Wireshark to capture and investigate the traffic, but we may not need the whole set of functionalities it provides. Instead, we could have a specially designed module for our monitoring system, based on the same library on which Wireshark relies—pcap. Namely, the pcap library provides us with the functionality to capture the traffic.

My point here is that each individual administrator has his own set of tools and way of working to discover and solve problems. With my GUI everyone can have only one window open, instead of several. In addition, since the tabs in the GUI are customizable, each administrator can order them the way he prefers and have his own specific set of

open tabs in the “Modules” panel. This is because each issue requires a different set of tools and methods to be discovered and solved.

Furthermore, once an anomaly in the network is detected, every tab opened in the “Modules” panel will provide functionalities specifically related to the same moment of time. This is not the case when you have multiple tabs in different windows opened. This is because you have to select the same time frame manually in every opened tool. Another advantage is that, the administrator can still have a visual overview of the problem he is currently resolving and when it has occurred. This also provides him with the possibility to dynamically include and exclude other events from the timeline view, and build a bigger picture of how different events have occurred in time.

One important assumption was made while this user interface was designed: the investigation should always start from the first reported event or anomaly, because others might follow consecutively to it. However, in reality, reports about secondary problems may be recorded in time before the initial problem. An example for this could be a record in the system about several hundred unreachable nodes without a previous record for a base station which is down and which is responsible for serving these nodes. This problem has been briefly discussed in Section 3.1.3.

One way how this could be avoided is with SNMP trap messages. An SNMP trap message is initiated by a network element and is sent to the management system in case of sudden or unexpected changes in this element or in some of its services [49]. As an example here, we will take a network element which is managed remotely via SNMP. The way SNMP works is to send information about the network element when it is requested. But in addition, it can monitor specific processes in the element and once a process is down, it can send information about its state, even when this information is not requested by the management system.

However, the above described method is not a remedy for every problem which may occur. Let us imagine a case in which many elements rely on another network element which is not equipped with a redundant power supply unit. In case its power suddenly goes down and the last poll from the management system was done recently with a successful status, this serving element will be off but the management system will not be aware of it before the next status check. This could be the case when many subscribers are currently connected via a base station, for which the power supply suddenly goes down. In this situation, the monitoring system may receive reports of a thousand elements which are not reachable without having a previous report for a failure in the base station which serves them.

One possible solution here could be to employ some heuristics about the hierarchy and dependency of the services. For example, in the above described scenario, the

management system can automatically check if all of these unreachable subscribers' elements rely on the same base station and automatically check if it is alive by sending a ping command. In case the base station is not responding, the system can record and report first a problem for the serving base station and store reports about the unreachable elements after. Another way could be when storing alarm or anomaly reports in the system, to also display which other elements and services rely on this node, so the network administrator can check their status as well.

Two more issues may appear during the development process of a monitoring system which is using this GUI. First is how to visualize on the "Timeline" occurrences with same timestamp, and second, how to distinguish one or a few anomalies, amongst several thousand other events reported on the "Timeline".

The way I approached the first problem is depicted in Figure 6.1. Here, it is visible that when a single bar is clicked on which represents more than one event recorded at the same time, it can be transformed into multiple bars. This is done in order to ensure that data records from various sources can be visualized, because theoretically it is possible that two records from two different locations are recorded with the same timestamp in the management system.

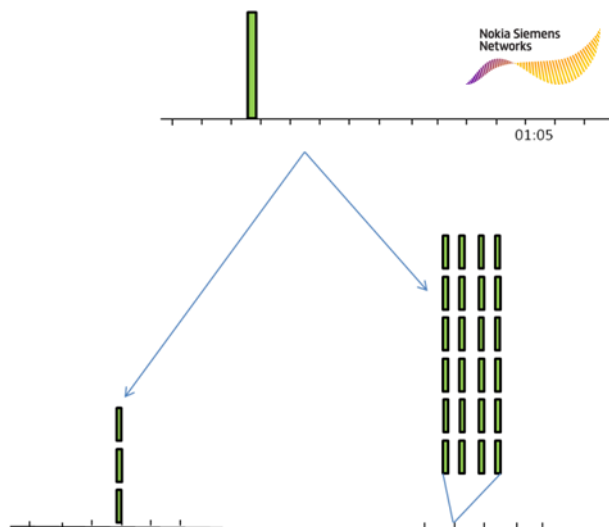


Figure 6.1 Visualizing multiple events with same timestamp

The solution to the second issue is visualized in Figure 6.2. It is visible from the figure that the "Timeline" provides a summary of the visualized events between the "From" and "To" calendar buttons. To distinguish the anomalies, it is sufficient just to press on the "Anomalies" label in the summarized events. This will cause the "Timeline" to exclude everything else and will show only this particular category of events, even if there is only a single one.

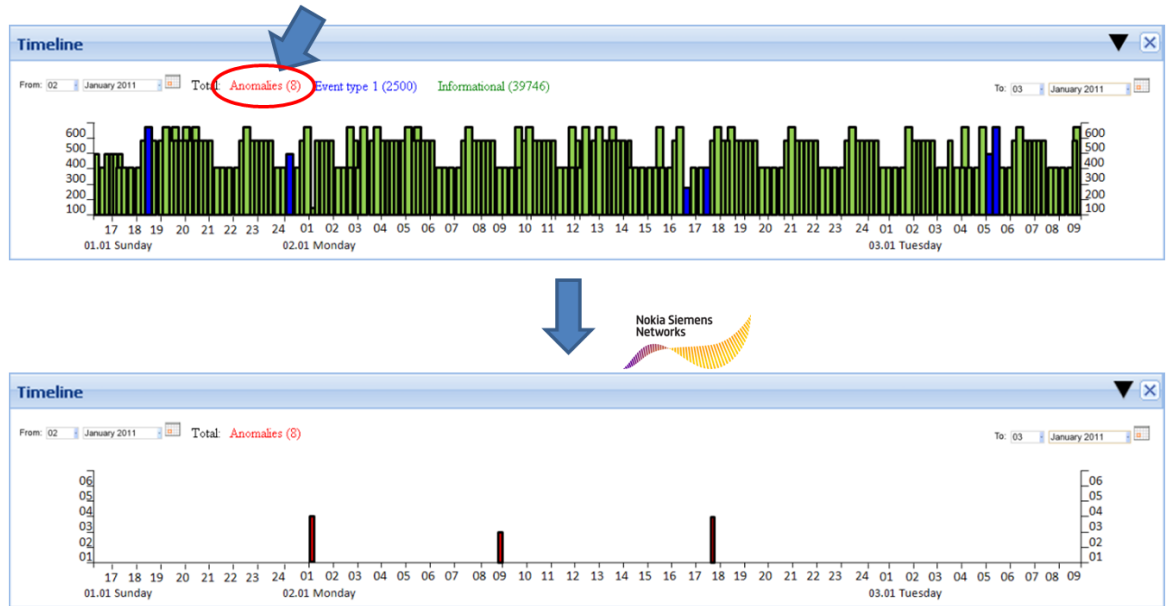


Figure 6.2 Filtering events on the timeline

In order to evaluate the GUI's usability, a real test use-case should be conducted. First, the whole system needs to be implemented or a similar one should be used instead. After this, two different GUIs can be used with the same system. For example, 10 persons can try to perform a root cause analysis and audit trail with my GUI, and 10 more could try to perform the problem solving activity with the usage of the same recorded data employing another interface. Afterwards, the output of both teams can be compared. Metrics which can be used to evaluate their performance can be their ability to discover the problem and time taken to resolve it.

7 CONCLUSIONS

In the scope of this thesis I have presented my vision for a GUI concept for anomaly-based NIDS in a mobile network operator environment. The user interface follows a drill-down technique, guiding the network administrator from a general overview of the overall network activity, to an aggregated view of NIDS data and a thorough analysis of the causes of the anomalies, alarms and problematic network elements. In particular, this thesis is focused on designing a front-end interface for a system which, in addition to its main scope of visually representing anomalies produced by an NIDS, can be used for general network monitoring purposes.

Two short imaginary use cases with synthetic data were described in this thesis. The goals were to demonstrate the tool's applicability for exploring records from a monitoring system, performing a root cause analysis for anomalies with the aim to verify their proper classification, and for analysis of service usage within a network environment.

Based on my observations and personal experience as a system and network administrator, my conclusion is that modular architecture design should be used when implementing the whole NIDS prototype which will use this GUI. This is because the possibility to test various detection methods implemented as modules will allow us to evaluate the most useful ones. In addition, any functionality implemented in the future could be easily integrated and visualized as a plug-in.

A decision has been made to implement a prototype which will use this GUI. The prototype will have a modular architecture design and will use several detection methods to classify anomalies. Another thing that should be done is to evaluate the GUI and to conduct a real test use-case.

REFERENCES

- [1] SANS FAQ, SANS Institute web page. [Accessed on 21.10.2010]. Available at: http://www.sans.org/security-resources/idfaq/what_is_id.php.
- [2] A. Sundaram, An introduction to intrusion detection, Crossroads, Volume 2, Issue 4, pp.3–7, April 1996.
- [3] R. Marty's blog, Raffael Marty's web page. [Accessed on 05.11.2010]. Available at: <http://raffy.ch/>.
- [4] S. Andres, B. Kenyon, Hardening the network infrastructure. Rockland, MA: Syngress Publishing; 2004.
- [5] SANS Institute, Intrusion Detection Systems: Definition, Need and Challenges. 2001.
- [6] E. Maiwald, Network Security: A Beginner's Guide, McGraw-Hill Professionals Publishing, 2002.
- [7] LinkedIn, LinkedIn Answers, What are the differences between Telecom Security vs. Enterprise Security? [Accessed on 15.11.2010]. Available at: <http://www.linkedin.com/answers/>
- [8] Rysavy Research, Security Requirements for Wireless Networking, 2007. Available at: <http://www.rysavvy.com/papers.html>
- [9] E. Tufte, The Visual Display of Quantitative Information, Second Edition, Graphics Press LLC, 2007.
- [10] D. Gullet, Snort 2.9.0 and Snort Report 1.3.1 on Ubuntu 10.04 LTS Installation Guide, version 1.01, October 8, 2010. Available at: <http://www.symmetrixtech.com/articles/008-snortinstallguide290.pdf>
- [11] C. Ware, Information Visualization – Perception for Design, Second Edition, Elsevier, 2004.
- [12] R. Marty, Applied Security Visualization, Pearson Education, Inc., 2009.
- [13] R. Bearavolu, K. Lakkaraju, W. Yurcik & H. Raje. A Visualization tool For Situational Awareness Of Tactical And Strategic Security Events On Large And Complex Computer Networks.. IEEE. 2003.

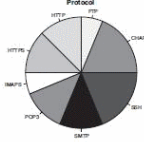
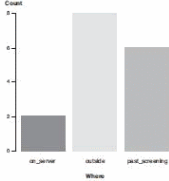
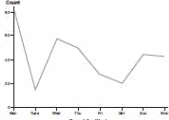
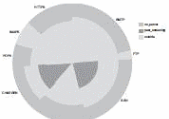

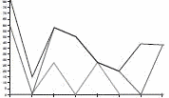
- [14] F. Fischer, F. Mansmann, Daniel A. Keim, S. Pietzko and M. Waldvogel. Large-Scale Network Monitoring for Visual Analysis of Attacks. Visualization for Computer Security. In Proc. of the 5th International Workshop, VizSec 2008, Cambridge, MA, USA, Spetember 2008.
- [15] M. Dodge, R. Kitchin, Atlas of Cyberspace, Addison-Wesley, 2001.
- [16] R. Henning and H. Fox, The Network Vulnerability Tool (NVT) – A System Vulnerability Visualization Architecture, National Information Systems Security Conference (NISSC), 1999.
- [17] H. Hosmer, Visualizing Risks: Icons for Information Attack Scenarios, National Information Systems Security Conference (NISSC), 2000.
- [18] P. Varner and J. Knight, Security Monitoring, Visualization and System Survivability Workshop (ISW), 2001.
- [19] S. Teoh, et al., Case study: Interactive Visualization For Internet Security, IEEE Visualization, 2002.
- [20] D. Denning, An intrusion-detection model, (IEEE) Transactions on Software Engineering, volume 13, number 2, pp. 222–232, 1987
- [21] C. Gates and C. Taylor, Challenging the anomaly detection paradigm: a provocative discussion. In Proc. of ACM Workshop on New Security Paradigms 2006, Schloss Dagstuhl, Germany, September 2006.
- [22] H. Javitz and A., Valdes, The SRI IDES Statistical Anomaly Detector, In Proc. of the IEEE Symposium on Security and Privacy, May 1991.
- [23] P. Chan, M. Mahoney and M. Arshad, A Machine Learning Approach to Anomaly Detection, Department of Computer Sciences, Florida Institute of Technology, Melbourne, 2003.
- [24] K, Wang and S, Stolfo, Anomalous Payload-based Intrusion Detection, Computer Science Department, Columbia University, New York, 2004.
- [25] K, Das, Protocol Anomaly Detection for Network-based Intrusion Detection, SANS Institute, GSEC Practical Assignment Version 1.2f, 2001.

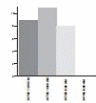
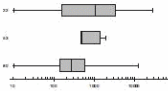

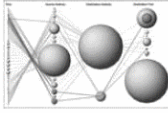
- [26] S, Staniford-Chen, et al., GrIDS—A graph based intrusion detection system for large networks, Department of Computer Science, University of California, Davis. Information and Communications Security. In Proc. of Third International Conference, ICICS 2001, Xian, China, November 2001.
- [27] The Mitre Corporation, Common Event Expression white paper, 2008. Available at:
http://cee.mitre.org/docs/Common_Event_Expression_White_Paper_June_2008.pdf.
- [28] Wireshark, Wireshark home page. [Accessed on 13.12.2010]. Available at:
<http://www.wireshark.org/>
- [29] Security Tools home page. Top 11 packet sniffers. [Accessed on 13.12.2010]. Available at: <http://sectools.org/sniffers.html>
- [30] Security Tools home page. Top 100 network security tools. [Accessed on 13.12.2010]. Available at: <http://sectools.org/>
- [31] B. Nickless. Combining Cisco NetFlow Exports with Relational Database Technology for Usage Statistics, Intrusion Detection, and Network Forensics, In Proc. of the 14th USENIX conference on System administration, CA, USA, 2000.
- [32] Internet Engineering Task Force (IETF). IP Flow Information Export home page. [Accessed on 13.12.2010]. Available at: www.ietf.org/html.charters/ipfix-charter.html
- [33] P. Barford, A signal analysis of network traffic anomalies, Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement, NY, USA, 2002.
- [34] Auditing Network Activity. Audit Record Generation and Usage System (ARGUS). [Accessed on 13.12.2010]. Available at:
<http://www.qosient.com/argus/>
- [35] Michal Zalewski. p0f home page. [Accessed on 13.12.2010]. Available at:
<http://lcamtuf.coredump.cx/p0f.shtml>
- [36] Microsoft. Windows Sysinternals Documentation. [Accessed on 14.12.2010]. Available at: <http://technet.microsoft.com/en-us/sysinternals/default.aspx>

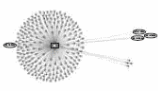

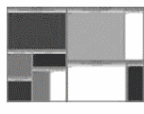
- [37] J., Dürsteler, Inf@Vis! – The digital magazine of InfoVis.net. Message No 187, February 2007. Available at:
<http://www.infovis.net/printMag.php?num=187&lang=2>
- [38] The Cacti Group, Cacti home page. [Accessed on 22.12.2010]. Available at:
<http://www.cacti.net/>
- [39] T. Oetiker. RRDTool home page. [Accessed on 22.12.2010]. Available at:
<http://www.mrtg.org/rrdtool/index.en.html>
- [40] J. Beale, Nessus Network Auditing, Syngress Publishing, Inc., 2004.
- [41] Cisco, IDS Event Viewer Documentation. [Accessed on 23.12.2010]. Available at:
http://www.cisco.com/en/US/docs/security/ips/4.0/configuration/guide/idm/swc_hap6.html
- [42] Stonesoft, Management Center Administrator's Guide vrsion 5.2.2, 2010.
Available at:
http://www.stonesoft.com/export/download/sg_man/StoneGate_Administrators_Guide_v5-2.pdf
- [43] Sourcefire, Snort home page. [Accessed on 28.12.2010]. Available at:
<http://www.snort.org/>
- [44] Sourcefire, Sourcefire Cybersecurity home page – 3D System Demo. [Accessed on 28.12.2010]. Available at:
<http://www.sourcefire.com/resources/3d-system-demo>
- [45] J. Duffy, Network World, Cisco outlines LTE strategy, September 2009.
Available at: <http://www.networkworld.com/community/node/45226>
- [46] Nagios Enterprises, Nagios home page. [Accessed on 03.01.2011]. Available at:
<http://www.nagios.org/>
- [47] AfterGlow Project, AfterGlow Project home page. [Accessed on 10.12.2010].
Available at: <http://afterglow.sourceforge.net/index.html>
- [48] R. Meyer, Challenges of Managing IDS in the Enterprise, SANS Institute, 2008.
- [49] CISCO, Understanding Simple Network Management Protocol (SNMP) Traps, October 2006. Available at:

http://www.cisco.com/en/US/tech/tk648/tk362/technologies_tech_note09186a0080094aa5.shtml

Appendix A

Visualization Technique	Data Dimensions	Maximum Number of Data Values	Data Type	Use-Case	Example Application	Example Chart
Pie chart	1	~10	Categorical	Use to compare values of a dimension as proportions or percentages of the whole.	Proportion of application protocols.	
Bar chart	1	~50	Categorical	Use to show the frequency of the values of a dimension or the output of an aggregation function. Each bar represents a value. The height of the bar represents the frequency count of the value.	Number of bytes transferred per machine.	
Line chart	1	~50	Ordinal, interval	Use to show the frequency of the values of a dimension or the output of an aggregation function. The height of data points in the chart indicates the counts. The data points are connected by lines to help display patterns or trends.	Number of blocked connections per day.	
Visualization Technique	Data Dimensions	Maximum Number of Data Values	Data Type	Use-Case	Example Application	Example Chart
Stacked pie	2	~10 times 5	Categorical	Use to compare values of two dimension as proportions or percentages of each whole.	Based on the role of machines, identify the percentage of protocols used to connect to the machines.	
Stacked bar	2	~50 times 5	Categorical	Use to show the frequency of values or the output of an aggregation function for two dimensions. The chart represents one dimension as the bars. The second dimension is represented as subdivisions in the bars.	For each destination port, identify the role of the machines involved in the traffic. The role is determined by the protocols the machine was using.	
Stacked line	2	~50 times 10	Ordinal or interval for each of the data series	Use to show the frequency of values or the output of an aggregation function for multiple dimensions.	Number of attacks per day across multiple locations.	

Visualization Technique	Data Dimensions	Maximum Number of Data Values	Data Type	Use-Case	Example Application	Example Chart
Histogram	1	~50	Ordinal or continuous	Use to indicate the shape of the distribution of values.	Distribution of number of logins over period of a day.	
Box plot	2	~10	Continuous, categorical	Use to show distribution of values. The categorical dimension can be used to split into multiple box plots for comparison.	Distribution of packet size in traffic.	
Scatter plot	2 or 3	Thousands for each dimension.	Continuous, continuous	Use to examine how two data dimensions relate or to detect clusters and trends in the data.	Show communication patterns of machines by plotting the endpoints along with the destination ports they accessed.	
Parallel coordinates	<i>n</i>	Thousands for each dimension. Up to 20 dimensions.	Any	Use for visualizing multidimensional data in a single plot.	Analyzing firewall rulesets to show for each rule what traffic is affected.	

Visualization Technique	Data Dimensions	Maximum Number of Data Values	Data Type	Use-Case	Example Application	Example Chart
Link graph	2 or 3	Without aggregation: 1000	Any, any	Use for visualizing relationships among values of one dimension and across multiple dimensions.	Identify the impact and extent of a compromise by visualizing communications of the compromised machine after the attack.	
Map	1	100	Coordinates, any	Use to display data relative to a physical location.	Number of trouble tickets per state.	
Treemap	<i>n</i>	10,000	Categorical, any	Use to visualize hierarchical structures in data. Enable comparison of multiple dimensions at once.	Assess risk by visualizing severities and criticalities of vulnerabilities per machine.	

Appendix B

